6AD－5

3次元形状モデルの位相データの圧縮

増田 宏, 青野 雅樹, 大渕 竜太郎
日本アイ・ビー・エム株式会社 東京基礎研究所

1 はじめに

最近, 製造業では設計データの3次元モデル化が急速に進行している。それに伴い, 自動車や家電業界では3Dモデルのデータ量の膨大さが問題になり始めている。一つにはエンジン部品のような複雑な形状を3次元化する必要があるからであり, もう一つには非常に多数の部品を同時に扱う必要性からである。今後, 3Dモデルが設計のマスターモデルになっていくと, データ量の大きい3次元データを多数保持したり, ネットワークを通じて高速にデータをやりとりすることが必要になってくる。しかし, 現状では3次元モデルのデータ量が非常に大きいため, 膨大なデータ容量を必要としたり, データ転送に非常に時間がかかるという問題が生じる。

このような問題点を解決する一つの方法は, 3Dモデルデータを圧縮することである。しかし, CADモデルに適した圧縮方法の提案はほとんどなく, また従来のテキストベースの圧縮法を流用しただけでは圧縮率は低い。本稿では, 3次元形状モデルの位相データについて圧縮方法を考え, 実験を行なったので報告する。

2 位相データの圧縮手法

3次元形状モデルのうち, 三角形メッシュモデルに関しては, 構造が単純で最適化処理が行ないやすいため, lossless なデータ圧縮法が提案されている [1, 2]。しかし, 通常, サーフェスモデルやソリッドモデルでは, 面の穴や1角形や2角形も許しており, 三角形メッシュの手法は適用できない。そこでここでは, ワイヤフレームとサーフェスの混在も許した任意の位相の3Dモデルデータを圧縮, 解凍するために以下の手法を用いる。

1. 3Dモデル M に可逆な位相変形操作 (オイラー操作) を施し, 多様体サーフェスモデル $\{S_i\}$ と, 連結なワイヤフレームモデル $\{W_i\}$ に分離する。さらに S_i の各面を分割し、三角形メッシュモデル T_i に変換する。
2. $\{T_i\}$ と $\{W_i\}$ を spanning tree 等を用いて encode する。さらに、これらから元の位相構造 M を復元する位相操作のためのデータを encode する。
3. encode されたデータに辞書ベースの圧縮手法を適用し, データの圧縮を行なう。

解凍は, この手順を逆に辿り、辞書ベース圧縮データの解凍、三角形メッシュとワイヤフレームの復元、サーフェスの復元 ($T_i \rightarrow S_i$), 元の位相構造の復元 ($\{S_i\} + \{W_i\} \rightarrow M$) という手順で行なう。

2.1 位相操作

まず, 可逆であるオイラー操作を用いて位相データを圧縮しやすい形態に変形する。ここでは座標値情報を使わず位相データのみを用いるので, 幾何データの誤差に影響されない確実な圧縮が実現できる。手順は以下の通り。

(A) 最初に3Dモデルをワイヤフレーム部分とサーフェス部分に分離する。サーフェス部分に非多様体稜線が含まれていたら, 多様体の集合になるようにさらにサーフェスを分離する。分離する操作を行なったときには, 同一であった位相要素の ID のペアを table に保持しておく。

(B) 図1のような手順で,face を三角形に分割する。追加した vertex, edge はそれぞれ ID を table に保持する。loop の連結 (a) を行なったときは, 外郭 loop 上の一つの edge (どれでもよい) の ID と, loop を反時計周りに辿る向きの始点の ID を table に保持する。

2.2 Spanning Tree の encode

図2は、すべての vertex を通る spanning tree を示している。ワイヤフレーム、サーフェスのいずれの場合も、まず、spanning tree を作成する。vertex は辿られる順に番号付けされる。図2の spanning tree は (1 2 3 (4 (5 6 7 8 9 10) 11) 12) と記述できるが index が連続しているので, 連続する頂点の個数を用いて (3(1(7)1)1) のように encode できる。

2.3 ワイヤフレームの encode

ワイヤフレームモデルは, spanning tree とそれに含まれない edge の両端点 の index (sv_i, ev_j) $(sv_i < ev_j)$ で表現できる。後者は sv_i でソートし, $(sv_i - sv_{i-1}, ev_j - sv_i)$ $(i > 1)$ を encode されたデータとする。

Compression of 3D Geometric Models
Hiroshi Masuda, Masaki Aono, Ryuraro Ohbuchi
IBM Research, Tokyo Research Laboratory

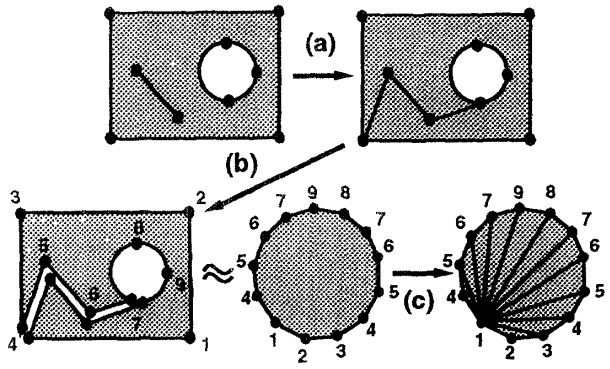


図 1: 位相操作の手順.

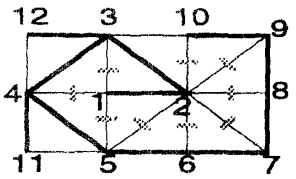


図 2: Spanning Tree.

2.4 サーフェスデータの圧縮

本手法では, サーフェスモデルは三角形メッシュモデルに変換されているので, 既存の三角形メッシュ圧縮手法を利用することができる。ここでは, Taubin の提案した方法 [1] を採用した。この手法は, 大雑把にいうと, 図 2 で spanning tree にはさまれ, 辺を共有して連なる三角形を encode していくもので, 始点番号 (図の番号とは異なる), 三角形の個数, 三角形 1 個につき 1 ビットのフラグで三角形の列を表現できる。

ただし, 三辺すべてが spanning tree 上にない三角形の場合には例外処理が必要である。Taubin は, jump edge を導入したが, ここではもっと単純に三角形の三頂点の index で面を表現するものとする。頂点の index は, ワイヤフレームモデルと同様に, ソートして差分を取ることでデータ量を小さくする。

2.5 位相要素の naming

三角形メッシュモデルから元の位相構造を復元するには,2.1 で示した位相要素のテーブルが必要である。ここで, 位相要素を示す ID は, 解凍後も同じ名前で参照できていなければならない。そこでこの ID として, 解凍するときに生成される vertex, edge, face の順番を位相要素の index となるように table を記述しなおす。もし,edge や face が曲線/曲面の式を持つ場合には, この index で記述されたポインタを持つ。

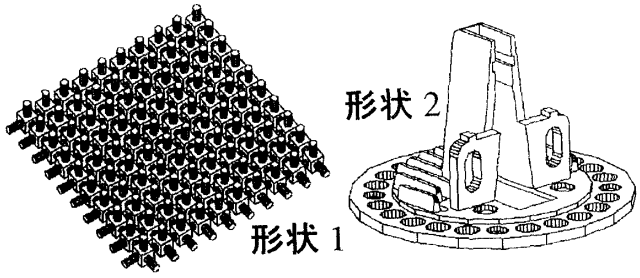
なお,table の encode では, index を列挙した場合と, すべての vertex, edge に関して, table に含まれるか否かを示す 1 ビットのフラグを付加した場合とを比較し, データ量が小さい方の encode 法を用いる。index を列挙する場合には, ソートした後に差分をとることでデータ量を小さくする。

2.6 辞書ベースの圧縮

上記の方法で encode されたデータを, 辞書ベースの圧縮法を施したものを最終的な圧縮データとする。本手法では, 位相的に近傍のデータには近い index が割り振られるために, encode されたデータには 0,1,2 に大きな偏りができ, 圧縮しやすくなる。また, 類似の位相構造を encode すると, 同一の数字列が割り当てられる傾向があるため, 大幅なデータ圧縮が可能になる。

3 結果とまとめ

図 3 の形状 1, 2 のワイヤフレーム表現とサーフェス表現において, edge, face を頂点 index で表現した ASCII データ, それを gzip で圧縮したデータ, 本手法で圧縮したデータとを比較した。圧縮率の欄は, 上は ASCII データ, 下は gzip 圧縮データの何%まで圧縮されたかを示している。これにより, 本手法により大幅なデータ圧縮が実現できることがわかる。今後は, 幾何データの圧縮方法と組み合わせて実際の CAD モデルに適用していく予定である。



形状番号	要素数	ASCII データ量	Gzip データ量	本手法 データ量	圧縮率
形状 1 ワイヤフレーム	15,600 EDGES	153.89 KB	54.75 KB	0.23 KB	0.15 % (0.42 %)
形状 2 ワイヤフレーム	3,304 EDGES	29.72 KB	12.67 KB	0.66 KB	2.23 % (5.24 %)
形状 1 サーフェス	5,520 FACES	495.52 KB	186.59 KB	11.43 KB	2.31 % (6.12 %)
形状 2 サーフェス	1,116 FACES	93.35 KB	38.80 KB	4.21 KB	4.51 % (10.84 %)

図 3: 圧縮率 (上は ASCII, 下は gzip との比較)

参考文献

[1] G.Taubin 'Geometric Compression Through Topological Surgery,' *IBM Research Report RC-20340*, Lan. 1996.

[2] M.Deering, 'Geometry Compression,' *SIGGRAPH'95*, 1995.