

実用化が進む 3D 形状圧縮技術

(3D Geometry Compression on the Practical Stage)

増田 宏

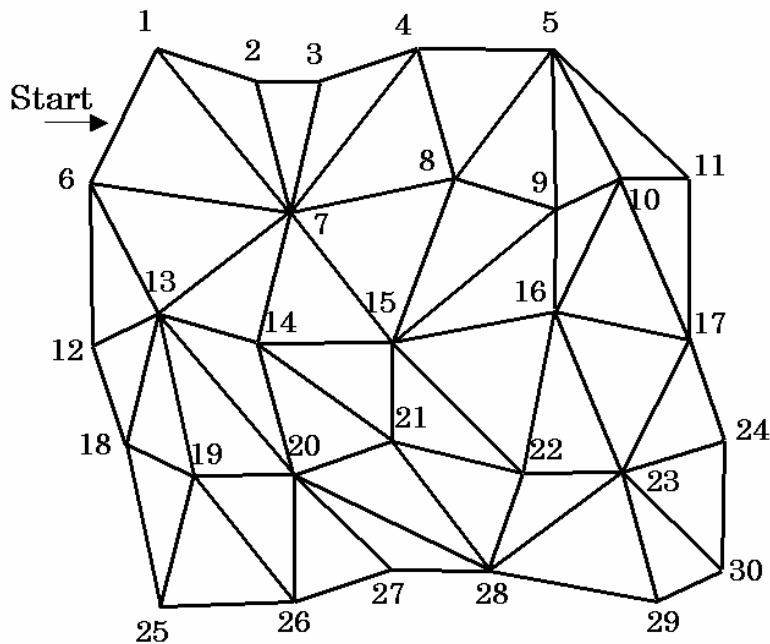
1. はじめに

3D CAD が広く普及し、設計などの業務で 3D モデルを作成することが一般的になってきた。それにつれて、設計以外の部門でも、設計評価や情報伝達の目的で 3D モデルを利用しようとする要求が強まっている。最近では、複数の事業所で協同的に製品設計を行ったり、設計生産の一部をアウトソースしたりということも広く行われている。そのような場合には、3D モデルをネットワーク経由で転送する必要がある。3D モデルの転送において、3D モデリングで利用させるデータをそのまま送るとは、ネットワーク負荷とセキュリティの点で好ましくない。形状モデリングに要求される形状の精度は閲覧やデータ参照に必要な精度に比べて遥かに高い。また、3D CAD ではその機能が向上するに従って、フィードバックや履歴など、形状以外的大量の情報を含むようになつた。最近では 3D モデルに設計知識を埋め込まれることもあり、データ量はさらに増大する傾向にある。形状モデル単独で見ても、最近の CAD は複雑な位相が扱えるようになっていて、データ量の増大や企業ノウハウの流出の問題が顕在化するようになってきた。このようない付加データや精度はモデリング作業や高い精度を要求するアプリケーションでは必須であるが、形状の情報共有だけが目的であれば絶対必要なものではない。

このような事情から、付加情報を含まず、また形状データ量の小さい軽量モデルが注目されるようになってきている。大手の CAD ベンダでも、精度を要求しないうえに閲覧などの目的に特化した軽量フォーマットを提供するようになっており、また、軽量データの提供に特化した曲面表示の二通りが使われている。大手の CAD ベンダが提供する軽量フォーマットは、その前身がコンピュータグラフィックス用途の表現形式であったこともあり、三角メッシュデータを基本としたものが多いようである。また、XVL に代表されるパラメトリック曲面圧縮技術は

少ないデータ量で高精度な表現が可能なことをその特徴としている。

形状圧縮の研究は、1995年頃から非常に多くの研究がされてきた。形状データは、位相データと幾何データに依って記述されるので、これらの幾何と位相をどれだけ少ないビット数で表現できるかが主要な課題となる。形状圧縮では、三角形メッシュ、多角形メッシュ、曲面モデルなどの表現形式の違い、また、形状として穴や多様体以外を許すか等によっても、手法が異なっている。形状圧縮手法のバリエーションは非常に多いので¹⁾²⁾、目的に応じた使い分けも必要である。本稿では、主要な形状圧縮技術に関して実用化された手法を中心に紹介していくことにする。



R6p, O1, O7p, O2, O3, M4, M8p, O5, O9p, O10, M11, O17p, M16p, M-3, O15p, O-5, O6, M14p, O13p, M-9, O12, M18p, M19p, M20p, M-5, O21p, O-7, O22p, O-9, O23, O-10, O-7, M30, M29, M28, M-1, O-2, M-3, M27, O26, M-4, O25, O-5

図 1 . 一般化三角形メッシュ

文献 3) より転載

2. メッシュモデルの圧縮

ここでは、メッシュモデルの圧縮法について代表的なものを紹介する。現時点で実用化された圧縮法のほとんどは三角形メッシュを対象としたものである。

2.1 一般化三角形メッシュ

Deering³⁾ は形状圧縮 (Geometry Compression) という考え方を導入し、従来の三角形ストリップを拡張した一般化三角形メッシュを提案した。また、座標や法線を量子化してデータを圧縮する方法も示した。この方法は、その後、Sun Microsystems 社の JAVA3D に実装された。この方法は、ハードでの扱いに適した手法であり、データ交換の目的で利用する場合には圧縮率は高くない。三角形メッシュの頂点は複数の三角形に共有されているので、三角形ごとに座標を転送すると、同じ座標を何回も送らなければならない。そこで、隣り合った三角形を連続的に送ることで重複を避ける方法がしばしば用いられてきた。Deering の提案した一般化三角形メッシュは、この考え方を広範囲のメッシュモデルに適用できるように一般化したものである。

図 1 に考え方を示す。メッシュの記述法を図の下に示す。この記法は、(R):開始、(O):転送順に 3 点を取って三角形を生成、(M):扇状に三角形を生成の意味である。また、16 個の頂点バッファを用意し、再利用され得るものを明示的に push 記号 (p) をつけて示し、過去の座標を再利用するときには、このバッファの位置をオフセット値で指定する。バッファは 16 個なので、オフセット値は 4 ビットで記述される。図では、オフセットは、マイナス記号をつけて示されている。

各頂点の座標・法線・色の数値も頂点と一緒に送られるが、それらはビット数を減らして表現する。メッシュモデルに使われる座標や法線などのデータは浮動小数点 (float:32 ビット) で記述することが多いが、この精度 (2^{-46}) はメッシュモデルが元々が近似表現であることを考えれば、著しく過剰である。Deering は、通常の閲覧目的であれば、座標 x, y, z それぞれに対して 16 ビット、法線を 9 ビット、色を RGB それぞれに対して 5 ビットに落としても品質がそれほど劣化しないことを示した。

2.2. Topological Surgery

Taubin ら⁴⁾ の提案した Topological Surgery と呼ばれる圧縮手法は MPEG4 の 3D モデル圧縮で用いられた。この方法では今日では必ずしも最適な手法とはいえないが、

その後の圧縮手法の基礎となる重要な考え方を示した。この手で通過する全域木をこの全木は図2(b)に示すように深さを優先で辿るものとし、図2(c)のように全域木をこのストリップと図2(b)の全域木を四角のマークで示したように対応つけて記述しておけば、受け取り側では、全域木を参照しながら三角形ストリップを張り合わせる。圧縮データとして転送するのは、座標列と相情報(全域木とストリップ)である。ストリップは、出発点の三角形の底辺からみて、連結する次の三角形が右(1)、左(2)、分岐(3)、終端(0)であるかによって符号化する。次の三角形についても、前の三角形と共有する辺を底辺として、同様に符号化する。したがって、三角形1個につき、2ビットで符号化されることになる。この例題では、図2(c)の破線で示した一番長いストリップの連結情報は、開始三角形の頂点番号と、[221222221]の18ビットの相情報で記述できる。この方法では、全域木によって決定される三角形のストリップが長くなるほど圧縮率は向上する。逆に、短いストリップが多数存在する場合には、開始三角形の頂点番号を多数保持しなければならぬため、圧縮率はあまなり高くなる。ストリップの個数を最小にする全域木を求める問題はNP完全問題であるため、この方法で形状圧縮においては、座標の圧縮に差を予測符号化を用いることが多い。典型的な方法は、既に転送された座標から次の座標を予測し、その残差のみをエントロピー符号化を用いて圧縮する。よい予測を用いれば残差の桁数が少なくなるので圧縮率が高くなる。予測手法は、圧縮方法に応じて決められる。

Taubin らが用いたのは、線形予測符号化である。全域木を辿った順番に座標を格納していき、数値の並びの規則性を利用することで必要なデータ量を削減する。頂点を全域木の深さ優先順に格納した場合、頂点は螺旋状に並ぶ傾向がある。そのため、頂点の予測には線形予測符号化が有効である。線形予測符号化は、頂点列 $\{v_i\}$ ($i = 1, 2, \dots, n$) に対して、

$$v_i = \varepsilon_i + \sum_{j=1}^K \lambda_j v_{i-j}$$

を用いて次のデータを算出する。K は予測に用いる頂点

数, ε_i は残差, λ_j は線形予測係数である. 係数 λ_j は残差 ε_i の 2 乗和が最小になるように最小自乗法で算出される. 復元の際には, 頂点列ごとに K 個の λ_j 値を転送する.

予測値との残差は整数に量子化され, Huffman 符号化などのエントロピー符号化が施される. 近似精度は, 量子化の幅によってきまる. 予測が十分適切なものであれば, 残差は小さい値に偏るので, エントロピー符号化が有効に機能する.

Taubin らの示した実験では, メッシュモデルの精度が低くてよいときは非常に大きな圧縮率を実現できることを示している. これは, 大きな桁ほど規則性があるので圧縮が効き易いが, 小さい桁は微小な変動で乱雑性が高いため圧縮が効きにくいからである.

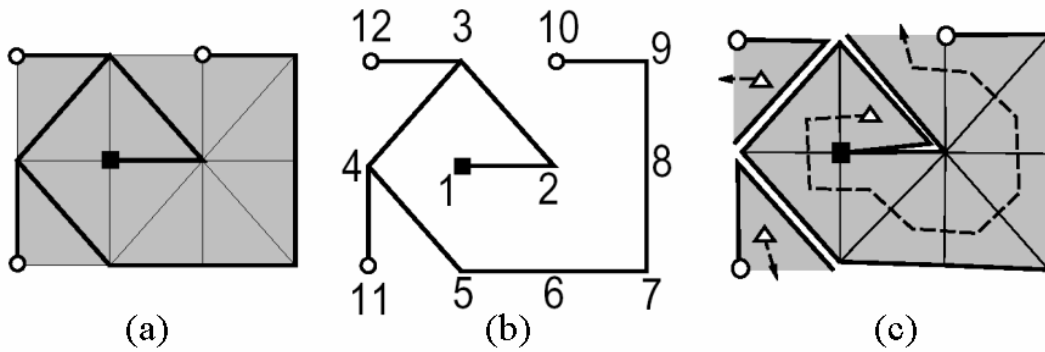


図 2 . Topological Surgery による圧縮

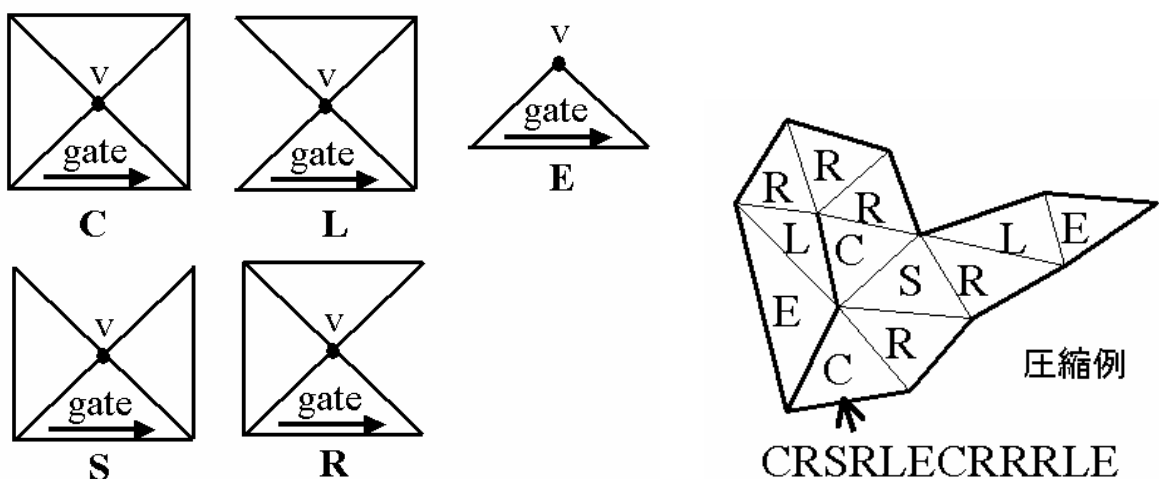


図 3 . Edgebreaker による圧縮

文献 5) より転載

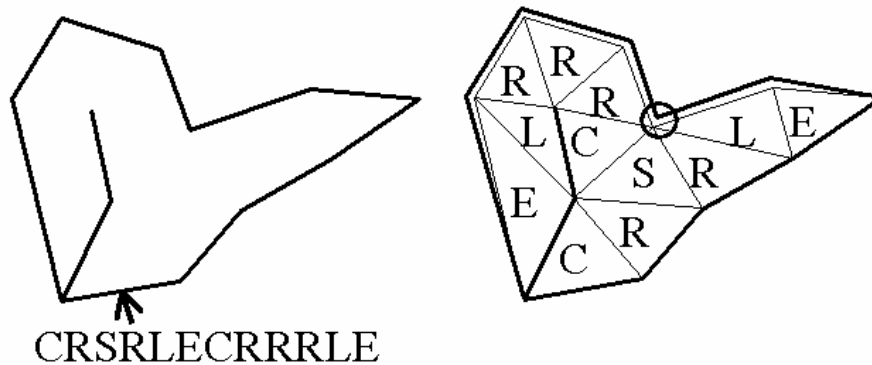


図 4 . Edgebreaker による圧縮データの解凍

文献 5) より転載

2.3. Edgebreaker

Edgebreaker⁵⁾ は Rossignac が提案した手法で , 商用システムなどで広く採用されている . この方法も , メッシュを三角形ストリップに切断するが , 分岐を含むストリップをより効率的に符号化することができる . 最悪のケースでも頂点あたり 4 ビットで位相を記述することができる . また , Taubin らの手法に比べて , 少ないメモリ量で処理が可能である .

考え方を図 3 に示す . メッシュを辿る際の起点となる三角形の一边を gate と呼ぶ . gate を底辺とする三角形の頂点 (黒丸で示す) に連結する三角形に応じて , 図に示すように C, L, S, R, E の 5 通りに分類できる . この図において , 既に辿られた三角形は除去されていると考える . これらは , C : 頂点の回りに境界がない , L : 境界が gate から見て左側にある , R : 右側にある , E : 左右にある , S : それ以外 , によって分類されている .

図 3 の右に , Edgebreaker による符号化の例を示す . 分岐 S においてはストリップが二股に分かれるので , 左側の gate をスタックに入れ , 左側のストリップを処理する . 終端 E に達すると , スタックから gate を取り出して左側のストリップの処理を続ける .

図 4 は , 解凍して元のメッシュを復元する様子を示している . 最初は閉領域とコード列 , 出発点の gate 位置が与えられる . 解凍においては , 出発点からコードを解釈して該当する三角形を復元していく . 分岐 S が現れると , 頂点の境界上の位置はその時点ではわからないので , 図の丸で囲んだ部分のように , 境界に沿わない形で三角形を生成していく . すべての三角形が復元されたら , 終端

コード E の頂点から辿り，境界と三角形の辺を連結して形状の復元を完了する．

2.4. TG 圧縮法

Tauma と Gotsman⁶⁾ は，向き付け可能な境界のない多様体に対して，頂点の次数（頂点回りのエッジの個数）に基づいた符号化手法を提案した．現時点では，最も効率のよい手法の一つと考えられる．ここでは，この圧縮法を，提案者の名前をとって TG 圧縮法と呼んでおく．

正則な三角形メッシュでは次数が 6 となるので，次数を列挙する符号化では正則に近いほど同じコードが連続して現れる．そのため，冗長性を排除するエントロピー圧縮と組み合わせると，非常に高い圧縮率を達成するこ

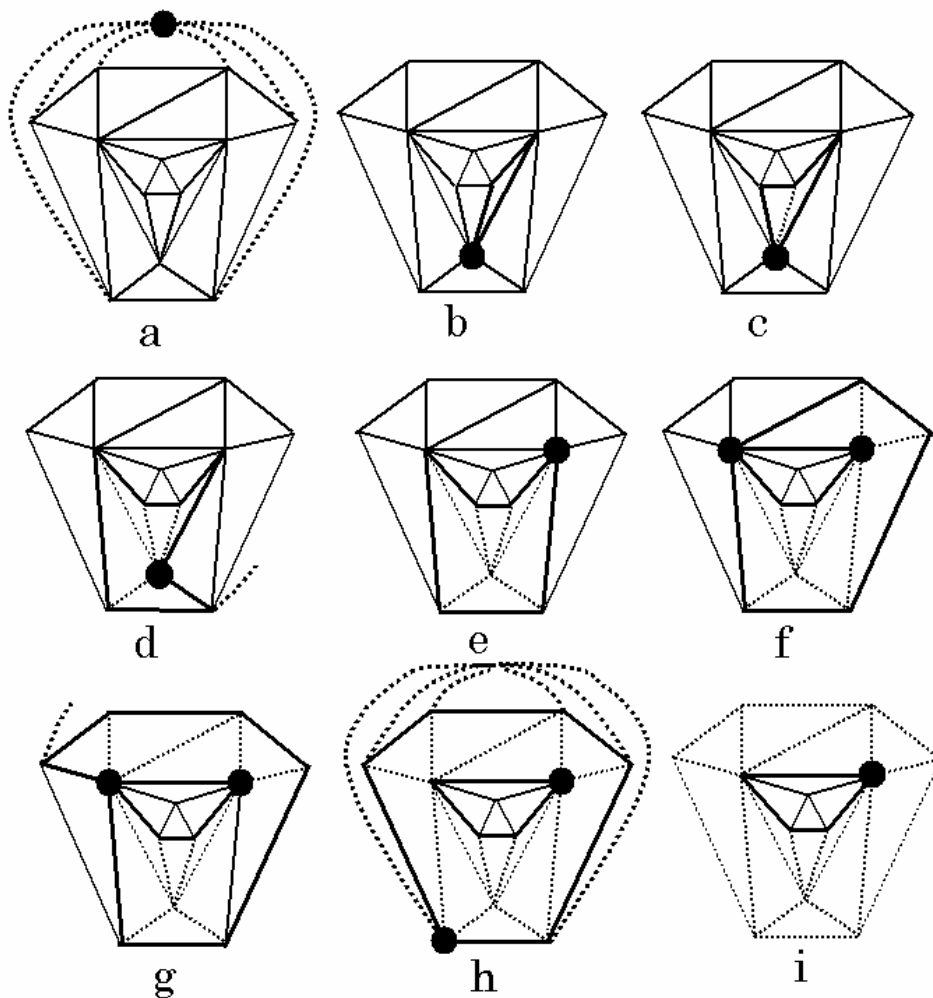


図 5 . T G 圧縮法

文献 6) より転載

とができる。正則なメッシュの場合には頂点あたりのデータ量が0.2ビット以下，それ以外では3~5ビットとなると報告されている。

図5の例を用いて圧縮手法の概略を示す。メッシュが境界を持つ場合には図5(a)の破線で示したように仮の点を作成して境界上の頂点と接続し，境界のない多様体に変換する。次に，図5(b)の太線で示した三角形の各頂点を注視点（黒丸で示す）から辿り，その次数を $add(6)$ ， $add(7)$ ， $add(4)$ と記述する。 add は頂点をリストに加える操作である。次に，注視点から順次，(c)(d)のように反時計周りにエッジを除去していき（除去されたエッジは破線で示す），端点の次数を $add(4)$ ， $add(8)$ ， $add(5)$ ， $add(5)$ と記述する。(d)では注視点周りの6頂点を辿ったので，頂点リストの次の頂点到注視点移動し，同様にして， $add(4)$ ， $add(5)$ と符号化する。(f)では，注視点の次数7に達する前に既に辿った頂点が見れている。この場合にはメッシュが二つに分割されたことを意味している。切断線は，注視点から5個前の頂点を結んだエッジなのでこれを $split(5)$ と記述する。分割された一方のメッシュはスタックに保持する。次に(g)のように注視点を移動させ， $add(4)$ ， $add(4)$ と符号化する。(h)では仮の頂点を $add_dummy(4)$ と記述する。これで分割した一方のメッシュは(i)に示すようにすべて符号化されたので，残ったメッシュの内部の頂点を $add(4)$ と記述する。これですべての頂点が符号化されたので，処理が終了する。結局，このメッシュの符号は， $add(6)$ ， $add(7)$ ， $add(4)$ ， $add(4)$ ， $add(8)$ ， $add(5)$ ， $add(5)$ ， $add(4)$ ， $add(5)$ ， $split(5)$ ， $add(4)$ ， $add(4)$ ， $add_dummy(4)$ ， $add(4)$ となる。解凍するときには，この符号に基づいて順次頂点と次数を決定していくことで，元のメッシュが復元される。

座標は，頂点を辿る順に格納しておく。この手法では，頂点が一個転送される度に三角形が一個決まる。そこで，辺を共有する二つの三角形を考え，一方の三角形の頂点を他方から予測する。この予測を平行四辺形線形予測と呼ぶ。隣接三角形の4頂点が平行四辺形の頂点であると仮定すると，三角形の頂点が (u, v, w) のとき，隣り合った三角形の頂点の予測値は $v + u - w$ となる。

2.5. プログレッシブメッシュ

プログレッシブ圧縮転送は，最初に粗いメッシュを転送し，その後送られるデータによって徐々に3Dモデルを詳細化していく方法である。データの受け取り側では，位相と幾何データを受信中に再構成していく。この方法により，データの転送中でもモデルが表示できる。

Hoppe⁷⁾ は三角形メッシュのプログレッシブ転送の手法を提案した。この手法は，Microsoft社の DirectX に採用された。

プログレッシブメッシュでは，段階的な粗いメッシュモデルを作成することが必要である。形状の簡略化には図6に示す edge collapse を用いる。この操作は，図6の頂点 V_t と V_s を新しい頂点 V_s' に縮退させ，2個の面と3個の稜線を削減する。このとき， V_s' ， V_l ， V_r の頂点番号，座標の差分 $V_t - V_s'$ ， $V_s - V_s'$ を記録しておく。形状の変化が最も小さい edge を選びながらこの操作を繰り返していくと元の形状の概略を保ったままで徐々に簡略化されていく。

解凍するときには，この操作列を逆向きに適用する。このときは，図6に示すように edge collapse の逆操作である vertex split という操作を適用する。簡略化のときに V_s' ， V_l ， V_r の位置が記録されているので，稜線 $V_s'V_l$ ， $V_s'V_r$ を展開して新しい頂点 V_s ， V_t を生成する。また，記録された差分を元の V_s' の座標に加えてこれらの頂点

の座標を復元する。

プログレッシブメッシュでは、 n 個の頂点があるとき、頂点番号の指定に $\log_2 n$ ビット必要なので、全体としては多くのデータ量が必要となる。したがって、この方法をそのまま利用しただけでは大きな圧縮率とはならない。

また、この方法を改良した、より圧縮率の高いプログレッシブ転送方式も提案されている。Pajarola ら⁸⁾は Hoppe らの手法で頂点番号の指定に多くのデータが費やされることに着目し、まとまった操作単位で転送する手法を示した。この方法では、操作対象の頂点をグルーピングした上で、どの頂点に対して vertex split を施すかを 1 ビットでマークしておき、近傍の頂点を連結情報に基づく局所的なインデックスによって記述することで必要なデータ量を減らしている。

また、Taubin ら⁹⁾は、Topological Surgery の手法を応用した Progressive Forest Split と呼ばれる手法を示した。この方法は、図 7 に示すように、メッシュ上に指定された木に沿って切り開き、その隙間に対して、図 7(c) のような三角形ストリップを埋め込んで詳細化する。ストリップには、位置合わせを行うために図で示した位置を記述しておく。埋め込む三角形ストリップは、高い圧縮率で符号化できるので、Hoppe の手法に比べてデータ量を少なくできる。

2.6. その他の圧縮手法

通常利用されているメッシュモデルには、三角形以外の面を含むことも多い。特に、パラメトリック曲面から生成されるメッシュモデルは四辺形となることが多い。

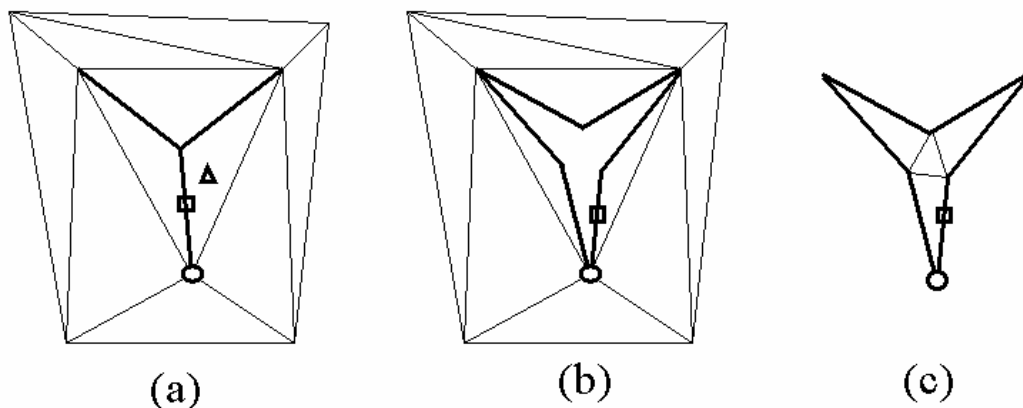


図 7. Progressive Forest Split

文献 9) より転載

四辺形メッシュは三角形メッシュに変換できるので，三角形メッシュの圧縮手法を利用することは可能であるが，面の個数が増えるのであまり効率的とはいえない．

多角形メッシュの圧縮手法についても，数は多くないものの提案されているので簡単に示しておく．大別して，場合分けを多くして多面体に対応させる手法と，分割して三角形メッシュに帰着される方法がある．いずれも三角形メッシュの圧縮に比べて処理が複雑になるという問題がある．

Isenbueg ら¹⁰⁾は，多面体モデルを三角形分割することなしに，直接圧縮できる方法を提案した．この方法は Edgebreaker で用いられた gate による分類を穴やハンドルを含む多面体に拡張したもので 7 通りのコードを用いて符号化を行う．

Masuda ら¹¹⁾は，穴の空いた面や空洞のあるソリッドモデルの圧縮を行う手法を示した．この方法では，位相変形操作であるオイラー操作を用いて，多面体を三角形に分割し，三角形メッシュの圧縮問題に帰着させている．解凍のときは，三角形メッシュに逆オイラー操作を適用して元の位相を復元する．

一方，大容量のモデルに対応した圧縮手法も研究されてきた．Ho ら¹²⁾は，主記憶に入りきらない大規模なメッシュモデルを転送するために，メッシュを分割して転送する手法を示した．個々の分割されたメッシュモデルは Edgebreaker と平行四辺形線形予測を用いて別々に符号化される．さらに，張り合わせるための報を付加することで，解凍後に個々の断片を張り合わせて元のメッシュを復元できるようにする．メッシュ単体で圧縮する場合に比べて 25% 程度データ量が増大すると報告されている．

3．曲面の圧縮

CADではパラメトリック曲面が多く用いられるが，曲面を含んだ形状モデルの圧縮手法も提案されている．

実用的に最も広く用いられているのは，XVL¹³⁾¹⁴⁾として知られている圧縮方法である．この方法では，ラティス構造と呼ばれるポリゴンデータに基づいて近似曲面を生成する．まず，ラティス構造の初期ポリゴンに対して細分割処理を 1 回を行い，再分割されたポリゴンに基づいて個々の曲面パッチの境界となる稜線を 3 次 Bezier 曲線として求める．次に，得られた曲線ネットワークに双 3 次 Gregory パッチを内挿することにより，自由曲面を得る．

この方法だけでは初期ポリゴンに対して曲面が一意に決まるので，形状の自由度は高くない．そこで，初期多面体の頂点と稜線に重みをつけることにより形状を制御し，高精度の近似モデルが生成できるようにしている．詳細なアルゴリズムについては，文献[13,14]を参照されたい．XVLでは複雑な自由曲面を含む製品を高精度に圧縮する用途に適している．

XVLとは異なるアプローチとして，制御点座標を直接エントロピー圧縮する方法も提案されている．増田ら¹⁵⁾が示したように，制御点は規則的に並んでいるので，制御点圧縮を画像圧縮に帰着させることができる．すなわち，離散コサイン変換やWavelet変換などを用いて空間座標を周波数成分に変換し，高周波成分をカットすることでデータ量を削減できる．パラメトリック曲面は元々滑らかになるように作られているので，高周波成分は大きくないという特徴がある．そのため，多くの場合，品質を大きく損なうことなく高い圧縮率を達成できる．また，これとは別の方法として，Santa-Cruz と Ebrahimi¹⁶⁾は，NURBS 曲面の制御点を平行四辺形線形予測によって予測し，差分を圧縮することで曲面圧縮を行う方法を示している．これらの手法は，CADデータの圧縮よりはむしろ汎用のマルチメディアデータ圧縮とみなす立場に立ったものと考えられることができる．

4．まとめ

本稿では実用化されている手法を中心として形状圧縮手法を紹介した．圧縮手法は既に実用化の域に入っており，現在では，商用レベルで圧縮率を高める努力が継続的になされている．ここで紹介したような手法のいくつかは，処理プログラムがWeb上で公開されており，圧縮率の評価を行うことができる．圧縮率は，どのような形状を対象にするかによって異なるので，利用する際には，手法ごとの性質を知った上で選択することが必要である．

参考文献

- 1) G. Taubin and J. Rossignac : 3D Geometry Compression, Course Notes for SIGGRAPH 2000, (2000)

- 2) P. Alliez and C. Gotsman : Recent Advances in Compression of 3D Meshes. Proceedings of the Symposium on Multiresolution in Geometric Modeling, Cambridge, (2003)
- 3) M.Deering : Geometry Compression, SIGGRAPH 95, (1995), 13.
- 4) G.Taubin and J.Rossignac : Geometry Compression through Topological Surgery, ACM Transactions on Graphics, 17, 2 (1998), 84.
- 5) J.Rossignac. Edgebreaker : Connectivity Compression for Triangle Meshes, IEEE Transactions on Visualization and Computer Graphics, 5, 1 (1998), 47.
- 6) C.Touma and C.Gotsman : Triangle Mesh Compression, Graphics Interface 98, (1998)
- 7) H.Hoppe : Progressive meshes, SIGGRAPH 96, (1996), 99.
- 8) R. Pajarola and J. Rossignac : Compressed Progressive Meshes, IEEE Trans. of Visualization and Computer Graphics, 6, 1 (2000), 79.
- 9) G.Taubin, A.Gueziec, W.Horn, and F.Lazarus : Progressive Forest Split Compression, SIGGRAPH 98, (1998), 123.
- 10) M.Isenbueg and J.Snoeyink. Face Fixer : Compressing Polygon Meshes with Properties, SIGGRAPH 2000, (2000), 263.
- 11) H.Masuda and R.Ohbuchi : Coding Topological Structure of 3D CAD Models, Computer-Aided Design, 32, 5-6 (2000), 367.
- 12) J. Ho, K-C. Lee, and D. Kriegman : Compression Large Polygonal Models, Proc. of IEEE Visualization Conference, (2001), 357.

13) A. Wakita, M. Yajima, T. Harada, H. Toriya, and H. Chiyokura : "XVL: A Compact and Qualified 3D Representation with Lattice Mesh and Surface for the Internet", Proceedings of ACM VRML 2000, (2000) 45.

14) 脇田玲, 矢島誠, 原田毅士, 鳥谷浩志, 千代倉弘明 : "ラティス構造に基づく軽量で高品質なWeb3Dデータ表現", 情報処理学会論文誌, 42, 5 (2001), 1170.

15) 増田宏, 大淵竜太郎, 青野正樹 : "周波数領域での曲面データの圧縮と転送", 情報処理学会論文誌, 40, 3 (1999), 1188.

16) Diego Santa-Cruz, Touradj Ebrahimi : Compression of parametric surfaces for efficient 3D model coding. Visual Communications and Image Processing (VCIP) 2002, Proc. of SPIE, 4671, (2002), 280.

増田宏

1987年東京大学大学院工学系研究科精密機械工学専攻修了。同年, 日本アイピーエム東京基礎研究所。1998年東京大学工学系研究科助教授, 1999年同人工物工学研究センター助教授。2003年より東京大学工学系研究科環境海洋工学専攻助教授。工学博士。設計工学, 形状処理工学の研究に従事。