

量子化法線ベクトルによる三次元高速ブラウジング

長澤 幹夫

超高速ネットワーク・コンピュータ技術研究所 (UNCL)

〒185 東京都国分寺市東恋が窪 1-280

m-nagasa@crl.hitachi.co.jp

三次元サーフェスデータは通常、ポリゴン集合として表現される。このポリゴンレンダリングにともなうエリアシング処理のような高品質描画計算は多大の負荷となるため、VRMLに代表されるネットワーク環境での大規模三次元データの高速レンダリングには、効率的なデータ削減方法やデータ表現形式が重要となる。そこで、我々は、三次元サーフェスを法線ベクトル集合として符号化する量子化法線ベクトル表現を提案する。各法線ベクトルは離散値として、その基となる三角パッチの面積に比例した長さと、テクスチャマップとして使われる不透明度分布の情報を持つ。この不透明度分布を視線積分することで、少ないデータ量での高速高品質レンダリングが可能となる。

FAST BROWSING OF 3-D SURFACE DATA USING QUANTIZED NORMAL VECTORS

Mikio NAGASAWA

Ultra-high Speed Network and Computer Technology Laboratories (UNCL)

1-280, Higashi-Koigakubo, Kokubunji, Tokyo 185 Japan

m-nagasa@crl.hitachi.co.jp

Three-dimensional surface structures are usually represented by the set of polygon patches. The accompanying anti-aliasing calculation of polygon rendering is heavy for the realistic massive 3-D data. Thus, especially in the network environment of VRML, an effective reduction algorithm and an alternative representation of surface data are important for the fast rendering. We propose the Quantized Normal Vector representation, which encodes the 3-D surface as a set of normal vectors. Each normal vector is discretized to have the length proportional to its sampling area and the opacity profile to be used in texture mapping. Since the pixel intensity is calculated by superposing the distributed vector opacities, the fast and high-quality rendering becomes possible.

1. 三次元遠隔画像通信

1.1 分散グラフィックス環境の広域化

現在、三次元データ通信の標準として、VRML (Virtual Reality Modeling Language) が提唱されているが、二次元画像における JPEG や MPEG のような圧縮技術の研究は、近年、著についたばかりである。

仮想現実の実現や、遠隔協調科学シミュレーションの実現には、三次元以上の高次元多変量データ通信が必要となってくる。特に、ATMに代表されるギガビットクラス的高速ネットワークが整備されると、遺伝子設計や分子設計の三次元構造データや、遠隔医療診断にMRIのボクセルデータを参照したりする三次元データアクセスが頻繁に行われるようになる。しかしながら、現状の科学シミュレーションが扱うような数百 MBytes の三次元データを最大解像度で転送可視化するには、スーパーコン的高速バス並の通信帯域や専用グラフィックス・マシンが要求される。そこで、一般のユーザーからも高効率アクセスを可能にするために、可視化目的だけに三次元データを転送するという大前提を活かした圧縮方式の開発が求められてきている。

1.2 ポリゴンの高効率アクセス方式

三次元ビデオゲームなどにおいて、幾何学的オブジェクトの多くは、ポリゴン集合によって表現されている。そのため、グラフィック・ワークステーションにおいては、このポリゴン処理を最適化するハードウェア設計がなされている。既に我々は、このポリゴンを対象とした圧縮方式について、隣り合うポリゴンにある閾値のもとに法線方向でグルーピングし、粗視化していく方式を開発提案している。^[1]

任意のスケラブルな圧縮レベルでデータを表現できることは、ネットワーク環境にあって巨大データをアーカイブしたり、与えられた通信帯域でリアルタイムにデータ転送処理するために重要なことである。このような Level-of-Detail の圧縮パラメーターを導入することによって、通信帯域を有効に利用したデータアクセスを可能としようとする技術開発は、すでに標準化の動きが始まっている。^[2] また最近ではポリゴンの LoD 圧縮に、ウェーブレット変換も用いられるようになってきている。^[3]

1.3 サーフェス・データ表現

サーフェスデータを表現するマクロスコピックなモデルは数多い。NURBS による関数表現や、メタボールによる曲面モデリングがあるが、やはり現在の主流はポリゴン表現である。実際、ポリゴンレンダリングは、グラフィック・ワークステーション上のハードウェアによる高速処理の研究が最も進んでいる。しかし、アプリケーションから求められるのは、ポリゴン集合そのものではなく、等値面などのサーフェス情報である。科学計算や医学分野での大規模データも、もし Marching Cube 法^[4]のような輪郭決定アルゴリズムを省略できるサーフェスデータ表現と効率的な圧縮アルゴリズムがあれば、ネットワーク環境のなかでも効率的に利用できると期待される。そこで本論文では、その三次元データ表現と三次元データ圧縮の目的のために、量子化法線ベクトル表現とそのレンダリング方法を提案する。これは、法線ベクトルと要素面積の情報からなる表現であり、通常のポリゴンデータからも容易に変換できる。ベクトル表現の特徴となるポイントは、ポリゴンエッジを無視したことである。この大胆な仮定により、三次元データ転送と遠隔可視化は、高い効率と耐故障性を備えることになるのである。

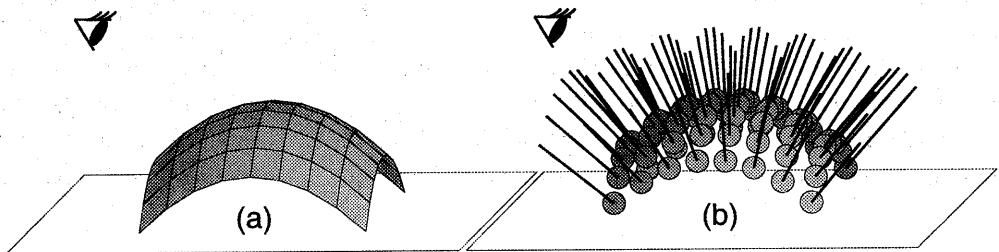


図1. (a) ポリゴン表現と (b) 量子化法線ベクトル表現の模式図。量子化法線ベクトルによるレンダリングは、面積成分に応じた半透明平滑パッチの重ね合わせであり、輪郭の描画処理が不要となる。

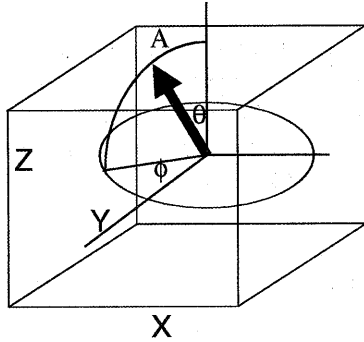


図2. 量子化法線ベクトル表現の変数要素属性。ベクトル $\mathbf{A}(\mathbf{x}_k)$ は、面に垂直で標本パッチの面積に相当するベクトル長を持つ。

X	Y	Z	θ	ϕ	A	c
1024	1024	1024	32	64	32768	256
1024	1024	1024	32	64	32768	256
1024	1024	1024	32	64	32768	256
1024	1024	1024	32	64	32768	256
1024	1024	1024	32	64	32768	256
1024	1024	1024	32	64	32768	256

図3. 量子化法線ベクトル表現のレコード形式。位置成分については (x, y, z) のデカルト座標系，方向成分については (A, θ, ϕ) 極座標で，ATM 通信セルのペイロード (48Bytes) に対応した固定長形式に離散化してある。

2. ポリゴンからベクトルへ

ポリゴンデータから量子化法線ベクトルを得る方法は、いくつか考えられるが、最も容易な方法としては、サーフェスデータを三角形ポリゴン集合に変換してから始める方法がある。与えられた三角形ポリゴンの重心 $\mathbf{x}_k = (\mathbf{x}_k^{(0)} + \mathbf{x}_k^{(1)} + \mathbf{x}_k^{(2)})/3$ をベクトルの空間座標 $\mathbf{x}_k = (x, y, z)$ とする。ここで、 $\mathbf{x}_k^{(0)}, \mathbf{x}_k^{(1)}, \mathbf{x}_k^{(2)}$ は、三角形の頂点座標である。また、三角形の外積ベクトルから、法線方向ベクトルと三角形ポリゴンの面積に対応する長さを決定する。

$$\mathbf{A}(\mathbf{x}_k) = \frac{1}{2}(\mathbf{x}_k^{(1)} - \mathbf{x}_k^{(0)}) \times (\mathbf{x}_k^{(2)} - \mathbf{x}_k^{(0)}) \quad (2.1)$$

ベクトル表現を得る一般のアルゴリズムとしては、与えられたサーフェスデータを分割し微小面積要素として分布させる Dividing Cube 法^[5]が利用可能である。ただし、Dividing Cube 法は正規化法線ベクトルを扱い、面積要素がないため、ピクセル解像度に依存するデータ形式となっている。これは量子化法線ベクトル表現において、高い解像度を要求される領域に分布するパッチサイズを縮小し、より多くのパッチを集めてベクトル化した極限に相当している。

2.1 符号化方式

まず、サーフェスデータは、空間的に広がりを持つ平滑パッチの集合として扱われる。平滑パッチは、分配された仮想面積 A と存在確率の内部分布関数 $W(\mathbf{x}, A)$ を持つ。これらの平滑パッチは、連結性の制約を外して三次元座標系内に任意に配置できる。この平滑パッチは、標本要素あるいは統計的確率分布とも解釈できる。数学的には、平滑パッチの核関数 $W(\mathbf{x}, A)$ の重ね合わせによって、サーフェスデータの存在期待値 $\langle \rho \rangle$ が計算される。

$$\langle \rho(\mathbf{x}) \rangle = \int W(\mathbf{x} - \mathbf{x}', A) \rho(\mathbf{x}') d\mathbf{x}' \quad (2.2)$$

量子化法線ベクトル表現とは、この仮想的な平滑パッチを格納するデータ形式なのである。つまり、面要素から計算される外積ベクトルによって生成されるものである。

したがって一般にベクトル表現は不均一な変形した三次元データの表現にも適しているが、良質のベクトル表現を得る符号化の過程においては、与えられたサーフェスデータ領域に対して、標本点の適切な位置決定をおこなわなければならない場合がある。この配置計算は、三次元ポテンシャル内の標本点として均等化する方法^[6]や、三次元サーフェスデータの均等表現としてパッチをあててポリゴン分布を均一化するアルゴリズム^[7]が利用できる。

さらに、ベクトルデータのデータサイズを圧縮する符号化において、シェーディングに十分であるとの理由から、ベクトルの方向成分は図2のような極座標で約6度の分解能で量子化されている。また、ATM ネットワークでの効率転送をはかるために図3にあるような整数値離散表現を用いている。各量子化法線ベクトルは一本あたり8Bytesの固定長を持つ。空間解像度は各次元につき10Bits ずつとしている。

2.2 復号化方式

量子化法線ベクトル表現からサーフェスデータを復号するには、ベクトルに垂直なパッチ分布から得られる重みを計算することになる。これにはいくつかの方法があり得るが、ここではピクセルの解像度によらない任意の空間座標で期待値の得られる方式として、統計平均手法を採用する。この場合、 i 番目のピクセル位置での期待値 $\alpha(\mathbf{u}_i, \mathbf{x}_k)$ は、近傍のパッチの重心位置 $\mathbf{x}_k$ からの重ね合わせの寄与を受けることになる。圧縮された量子化法線ベクトル $\{(x, y, z, \theta, \phi, A, \dots) | i = 1, K\}$ を復号化する際に、三次元座標 $\mathbf{u}_i$ における期待値は、粒子系のボリウムレンダリング^[8]と同様に、ガウス型の核関数を用いて次のように計算される。

$$\alpha(\mathbf{u}_i, \mathbf{x}_k) = \frac{1}{(\pi A_k)^{3/2}} \exp(-|\mathbf{u}_i - \mathbf{x}_k|^2 / A_k) \quad (2.3)$$

この表現変換の特徴として、量子化法線ベクトルは系の積分不変量を保存することができるという性質がある。一般には、符号化と復号化に伴う補間計算には数値誤差が入り込む。ところが、量子化ベクトル表現は、サーフェスデータの面積要素を直接表現しており、系の全面積は変換の前後において保存される。

2.3 ベクトル表現における最適化

ここで、一定のベクトルデータ数で、空間解像度を向上させる最適化を考察する。もちろん、構造化ポリゴンのように分割や合併によって標本点数を変化させることで、解像度を調整することも可能である。サーフェスデータが、特殊な基底の集合で表わされる場合、LOD と逐次転送表示が可能になるポリゴン表現も既に考案されている。^[9]しかしながら、ネットワーク環境でのデータアクセスを考慮した場合、なるべく一定の通信帯域でデータ転送できることが望ましい。そのため、ここでは、ベクトルを変化させずに解像度の向上をはかる標本点位置と転送順序の最適化を行なう。

面要素のダイナミックレンジを支配する重要なパラメーターのひとつは、ベクトル長、つまり平滑半径 $\sqrt{A}$ である。これは、個々の面要素の空間解像度にも対応している。統計期待値の計算を行なった場合、面要素が表現できるデータの最大最小の比、つまりダイナミックレンジを制限してしまう。

三次元データの逐次転送をする際、平滑半径の大きい順に、優先的な送出をすることができれば、受信側の復号化サーフェスデータは、大局的な構造に影響する大きなベクトルによって効率よく表現されることになる。そして、この面積優先順の適合型制御によって、物理的には固定長のベクトル表現が、論理的には非一様な重要度づけを持つサーフェスデータも、高い効率で圧縮表現できるという長所をもつことになる。

本制御方式により、Dividing Cube^[5]のようなピクセルサイズ程度の小さな面積要素だけでなく、大きさの非一様なサーフェスデータについても、圧縮効果が得られるという特長を持つ。ただし、要素面積の大きいベクトルについては、情報が平均化されてしまい、解像度は低下することに注意しなければならない。

また一般にサーフェスレンダリングでは、隣接ポリゴンとの乖離やねじれた面要素による誤表示などの問題が起りうるが、式(2.3)のような平滑フィルターによる統計処理は、重ね合わせの効果により、これらの局所的なエラーに左右されない復号化が可能である。

3. レンダリング・アルゴリズム

3.1 ベクトル表現からの可視化

平滑フィルターの等方性のおかげで、このサーフェスレンダリングでの不透明度計算は単純化される。スキャンライン法のアンチエイリアシングや、レイキャストによるボリウムレンダリングのような、交点の精密計算が不要になるからである。

量子化法線ベクトルが K 本あるときに、レンダリングのための光線積分は以下のように計算される。

$$C_\lambda(\mathbf{u}_i) = \sum_{k=0}^K \left[c_\lambda(\mathbf{x}_k) \alpha(\mathbf{u}_i, \mathbf{x}_k) \cdot \prod_{m=k+1}^K (1 - \alpha(\mathbf{u}_i, \mathbf{x}_m)) \right] \quad (3.1)$$

ただし、 $\mathbf{u}_i = (u_i, v_i)$ は、ピクセル画面座標である。

色輝度のモデルとしては、Phong のシェーディング関数を用いる。

$$c_{\lambda}(\mathbf{x}_k) = [k_{\text{amb},\lambda} + k_{\text{dif},\lambda}(\mathbf{N}(\mathbf{x}_k) \cdot \mathbf{L}) + k_{\text{spe},\lambda}(\mathbf{N}(\mathbf{x}_k) \cdot \mathbf{H})^n] \quad , \quad (3.2)$$

ここで、 $\mathbf{H} = (\mathbf{V} + \mathbf{L})/|\mathbf{V} + \mathbf{L}|$ はハイライト方向。 $\mathbf{N}(\mathbf{x}_k) = \mathbf{A}(\mathbf{x}_k)/|\mathbf{A}(\mathbf{x}_k)|$ は法線方向である。シェーディングには、光源方向 $\mathbf{L}$ 、環境光 k_{amb} 、拡散光 k_{dif} 、そして、指数 n の鏡面反射光 k_{spe} を用いる。添字の λ は色波長の依存性を示す。

量子化法線ベクトルでは、ポリゴンと異なってエッジが存在しないため、境界線を描画する必要がなく、エリasing処理も不要となる。したがって、解像度が低い、少数のベクトル表現からの直接レンダリングは、霧のような画像となってスムージングされることになる。

また、光線積分時の不透明度計算も式(2.3) という解析関数の重み積分をする正確なアルゴリズムが利用できるおかげで、実装方式としては 図4 のように、解析関数の重みをテクスチャとして作成することで、グラフィックワークステーションのハードウェアによるテクスチャマップ機能を利用した高速レンダリングが可能となる。

3.2 高速ブラウジング試験

ポリゴン表現とベクトル表現の比較のために、約 1000 個のポリゴンないしは量子化ベクトルからなる三次元サンプルデータを実際に VRML ブラウザー Ghostspace 上に表示した例を図 5 に示す。両者の差が際立つのは、面積要素の大きい部分である。図 5 の左サイドに表示されたポリゴンデータでは、粗いポリゴンのエッジが見えているが、ベクター表現のレンダリング出力では、平滑化されファジーな柔らかい画像となっている。一方、ベクトルの密集する高解像度部分では、くっきりとしたレンダリング出力となっており、ポリゴンレンダリングに比べて全く遜色はない。

ポリゴンデータ形式は頂点座標とその連結情報から構成されるため、ネットワークの転送プロトコルがデータの無損失を保証しておくことが必須である。一方、ベクトル表現では、ネットワークの帯域が狭かったり、データ損失の発生する回線であっても、各ベクトルの情報が独立であるため、全体の再送要求などは不要で、滑らかな高品質画像がレンダリングできる。また、Ghostspace に実装された逐次転送モードでは、転送される量子化法線ベクトルの数が増すにつれて、レンダリング画像も詳細がはつきりしてくる効果を出している。

ポリゴンレンダリング特にフラットシェーディングでは、スキャンラインの境界となるポリゴンエッジが目立ってしまうのに対して、ベクター表現ではスムーズではやけた境界構造となるのは、各ベクトル長として記録された要素面積に応じた、不透明度フィルターが作用した結果である。スムーズでファジーな輪郭構造は、ベクトル表現が「空隙」や「多成分系」といった複雑な三次元データ構造を容易に扱える柔軟性をも示唆している。

さらに、標準 VRML 形式によるポリゴンデータのサイズが 80KBytes なのに対して、量子化法線ベクトルデータは 14KBytes と約 20 % に削減されている事実から、ネットワークを介した遠隔レンダリングでは、ポリゴンレンダリングよりも効率的であるといえる。

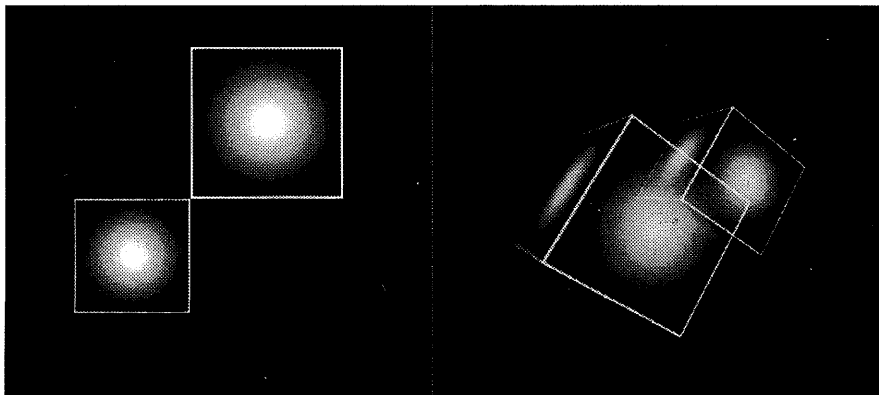


図4. ベクトル表現からのサーフェスレンダリングの原理。ベクトルに垂直な半透明平滑パッチのフィルター関数をテクスチャマッピングの技法を用いて重ねあわせる。

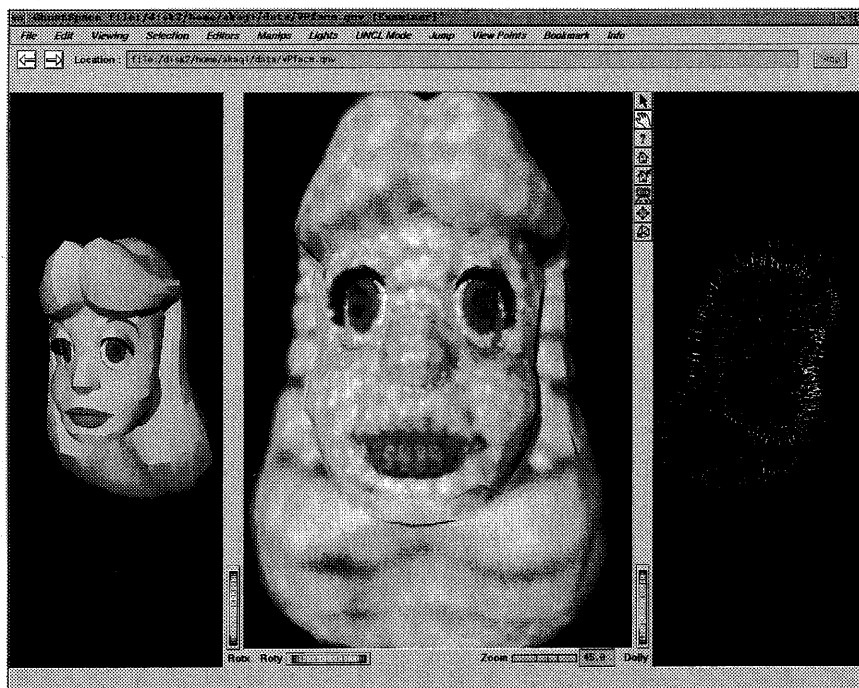


図5. VRML ブラウザー Ghostspace 上での量子化法線ベクトル表現によるレンダリング。画面左は、標準の VRML 形式によるポリゴン表示。画面右は、量子化法線ベクトルの線分表示。VRML 形式のポリゴンデータサイズ 80KBytes に対し、ベクトルデータは 14KBytes である。

4. 結論

今回提案した量子化法線ベクトルによる三次元データ表現方法の特徴とレンダリングアルゴリズムをまとめる。

(1) 量子化法線ベクトル表現は、三次元サーフェスデータをパッチ集合に分割し、個々の独立した法線ベクトルとして記述する。各法線ベクトルは 8Bytes の固定長レコード内に、座標成分、方向成分、面積成分、色属性のデータ構造 $\{x_i, \theta_i, \phi_i, A_i, C_i\}$ を持ち、データサイズの圧縮と ATM 通信に適した高効率転送に利用できる。

(2) 量子化法線ベクトルによるサーフェスレンダリングには、内部密度分布関数フィルターを導入する。面積成分に対応するこの分布関数フィルターの重量によって、任意の解像度の面積要素に対して、汎用性の高い高画質のレンダリング画像が得られる。また実装方式としては、ハードウェアによるテクスチャマップ機能を利用した高速レンダリングが可能である。

これらの結果は、量子化法線ベクトル表現が、サーフェスデータの圧縮転送表示に効果的であることを示している。

文 献

1. D. Nishioka and M. Nagasawa, 情報処理学会第 50 回全国大会論文集 6C10 (1995)
2. G. Taubin, <http://www.austin.ibm.com/vrml/binary> (1996)
3. A. Certain, J. Popovic, T. DeRose, T. Duchamp, D. Salesin and W. Stuetzle, *Computer Graphics Proceedings* (ACM SIGGRAPH) 91-98 (1996)
4. W. Lorensen and H. Cline, *Computer Graphics* 21, 163-169 (1987)
5. H. Cline, W. Lorensen, S. Ludke, C. Crawford and B. Teeter, *Medical Physics* 15, 320-327 (1988)
6. M. Nagasawa, *Digital Video Compression* (ed. A.A.Rodriguez) SPIE Proc. 2419, 288-295 (1995)
7. G. Turk, *Computer Graphics* 26, 55-64 (1992)
8. M. Nagasawa and K. Kuwahara, *Scientific Visualization of Physical Phenomena* (ed. N.M.Patrikalakis, Springer-Verlag Tokyo) 589-605 (1991)
9. H. Hoppe, *Computer Graphics Proceedings* (ACM SIGGRAPH) 99-108 (1996)