芸術科学会論文誌 Vol.3 No.1 pp.8－21

ポイントグラフィックス概説

Introduction to Point-based Graphics

藤本 忠博
Tadahiro FUJIMOTO
fujimoto@cis.iwate-u.ac.jp

今野 晃市
Kouichi KONNO
konno@cis.iwate-u.ac.jp

千葉 則茂
Norishige CHIBA
nchiba@cis.iwate-u.ac.jp

岩手大学工学部
Faculty of Engineering, Iwate University

アブストラクト

本稿では，最近活発になってきた研究分野である“ポイントグラフィックス”に関連して，点集合を扱うこれまでのＣＧ技術と，主にポイントグラフィックスとして最近開発されたレンダリング技術について解説し，ポイントグラフィックスはこれまでのＣＧ技術の自然な発展としてある魅力的な技術分野であることを示す.

キーワード：ポイントグラフィックス，ポリゴングラフィックス，ボリュームグラフィックス，点集合，デプス画像

１．はじめに

ポイントグラフィックス（Point-based Graphics）というセッションや国際会議が設けられるようになってきた. Eurographics では, 2002 年, 2003 年とチュートリアルも開かれており, 国内でもいくつか関連研究の報告がなされてきている [樫山 01，土橋 02，岡根 02，川田 03]. 形状モデルの冠を持ったグラフィックス技術には，現在もっともポピュラーなポリゴングラフィックス，またそれに対するボリュームグラフィックスが挙げられよう．このポイントグラフィックスは，ポリゴングラフィックスにもボリュームグラフィックスにも関係が深く，これまでのＣＧ技術の中から“ポイント”関連技術と解釈できるところを基礎として，体系化を進めるものであり，今後どこまで進展するか興味深い．筆者等のこれまでの研究にもポイントグラフィックスに関係するテーマが多くあり，研究の進展には大いに注目しているところである.

形状モデルとしては，ポリゴンやパラメトリック曲面など“面”モデルが主流であるが,“辺”モデルも古くはワイヤフレームモデルとして，また面を構成するための補助的な表現として存在している．これから考えると，孤立した“点”の集合からなるモデルというのは確かに新しい着眼点のように思える． 少なくとも，データ構造としてはシンプルであり，スキャン変換や LOD（Level of Detail）での取り扱いも楽そうであり，特に膨大なポリゴンを必要とする複雑な物体の取り扱いに効果を発揮するアプローチに思える.

本稿では，これまでのＣＧ技術でポイントグラフィックスと呼べるものと，ポイントグラフィッ

クスを意識することで開発されてきた技術について解説する．なお，サーベイではないので，関連する全ての文献の引用は行わないが，技術の概要を知るに十分な引用は行うつもりである．

２．ポイントデータとＣＧ技術

現在のＣＧ技術，特にレンダリング技術は，ポリゴンモデル（多面体モデル）をベースとして開発されてきている．しかしながら，点集合で表される形状データ（空間情報）も多く，点集合からポリゴンモデルを得ることは一般的には困難な問題である．そのため点集合からポリゴンモデルへの実際的な変換技術や，点集合を直接扱う効率的なレンダリング技術の開発が期待される．

３次元点集合データは，これまでにも以下のような形で自然に現れている．

（１）２次元格子点集合
（２）２次元非格子点集合
（３）３次元格子点集合
（４）３次元非格子点集合

以下，これらに関係するＣＧ技術について簡単に述べる．

２．１　２次元格子点集合のＣＧ技術

２次元格子点集合に属するデータとしては，主に次に示す３種類のものが挙げられる．

・ 標高値データ
・ レンジセンサによるデプス画像
・ Ｚ－バッファ

本節では，これらについて述べる．

・ 標高値データ

これは，DEM（Digital Elevation Model）あるいは高さの場（Height field）とも呼ばれ，国土数値情報など，２次元正方格子点上でサンプリングされた２次元場（主にスカラ場）を表すデータである．正方格子における整数座標値とそのスカラ値をもう一つの座標値とすることで３次元空間上の点が定義される．レンダリングには，最大－最小法や浮動水平線法（floating horizon algorithm）と呼ばれるワイヤフレーム表示，また，これを拡張して線と線の間を塗りつぶして表示する手法などが利用できる．一般的には，格子点を格子の対角線で接続し三角形化の補間を行い，ポリゴンレンダリングを適用するのが容易である（図１参照）．また，高解像度な DEM を得るための高次の補間法 [古舘 01] や，ポリゴン数削減の研究もある [Luebke01]．オーバハングした地形は，配列要素を標高値のリストにすることにより表現可能である．これは後述する LDI（Layered Depth Image）[Shade98] と同じである（３．４節参照）．

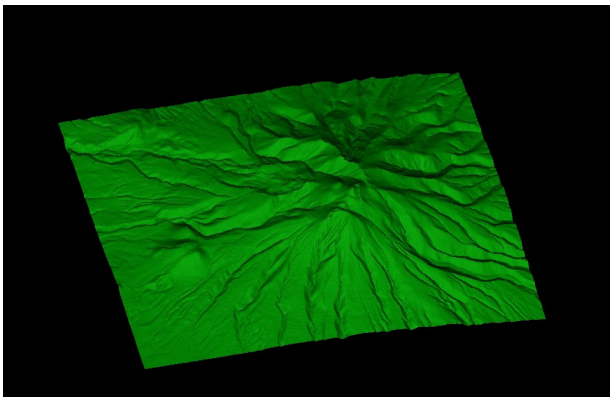


図１　標高値データをポリゴン化した例（国土地理院の数値地図 10 ｍメッシュ（火山標高）のデータを使用）

・ レンジセンサによるデプス画像

物体表面のポイントサンプリングであるが，サンプリングが格子状に行われるためデプス（奥行き）画像と呼ばれる．物体全体のサンプリングデータを確保するためには複数方向からの計測が必要であり，個別に計測したデータ間の位置合わせや，データ削減が問題となる．単純なポリゴン化としては，それぞれのデプス画像について三角形

化し，併合することが考えられるが，重複する領域では不適切なポリゴンが生成されることによって，形状モデルの品質を落とすことがある．また，併合したポイントデータから三角形ポリゴンを生成する手法も研究されている（２．４節参照）．

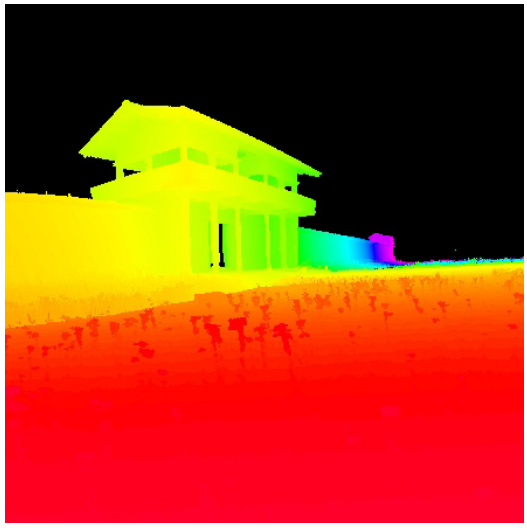
また，計測データの誤差が問題となるときには，計測データから直接ポリゴンを生成することは得策ではない．計測データは参照用として使い，形状モデラによりモデリングするなどの方法が，実用的には採られることも多い．図２ (a) は参照データの一部，(b) はモデリングして得られた形状を表す．また，(c) は (b) に示した形状モデルを利用して作成したＣＧ画像の例である．

・ Ｚ－バッファ

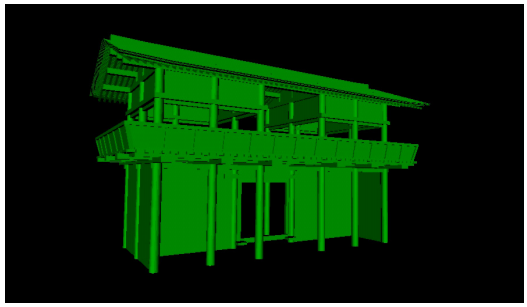
レンダリング処理後のＺ－バッファは，レンダリングの対象としたＣＧモデルに対するデプス画像となる．このとき，それぞれのピクセルに対して，そのピクセルにスキャン変換される全ての点のデプス値をリストとして持たせることで，LDI が得られる．また，これを直交する３つの方向から作ることにより，LDC（Layered Depth Cube）[Lischinski98] とすることも可能である（３．４節参照）．

２．２　２次元非格子点集合のＣＧ技術

計測点が必ずしも格子点上には無いような標高値データなどがこれにあたる．３次元空間上の点は，２次元平面（投影面）上の２次元座標値と，標高値などのスカラ値をもう一つの座標値として表される．ポリゴン化は，投影面上での２次元ドローネ網により３次元空間上の点を接続することで行うことができる．図３ (a) は，青色の点が投影面上の２次元座標であり，この点に関するボロノイ図を示す．(b) は (a) のボロノイ図に対応するドローネ網を示す．これらは，点の実数座標値からグラフの接続関係を求める幾何計算により生成できる他，投影面を２次元配列としたデジタル幾何学によっても生成可能である．



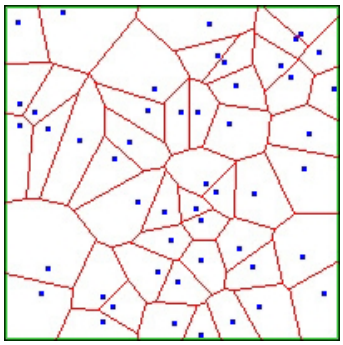
(a)　参照データの一部



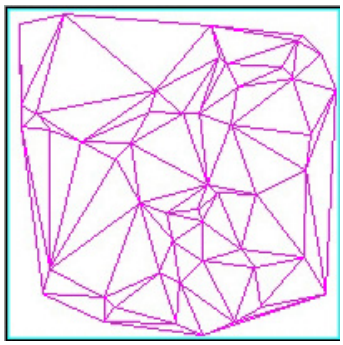
(b)　(a)を参照データとしてモデリングした例（(有)イメージクラフト提供）

(c)　(b)を利用して得られたＣＧ画像（(有)イメージクラフト提供）

図２　デプス画像を参照画像としてモデリングした例



(a)　ボロノイ図



(b)　ドローネ網

図 3　ドローネ網によるポリゴン化の例

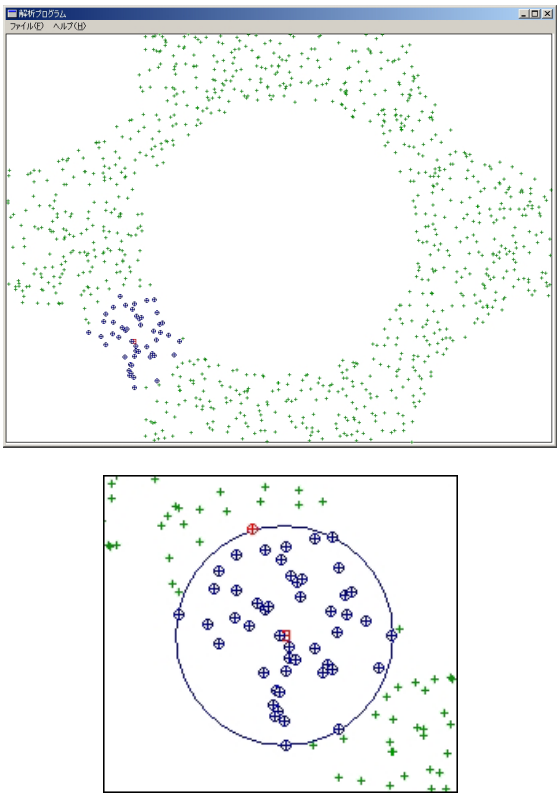


図 4　セルを利用した近接点探索の例

また, これに関連して, 投影面を格子で分割し, 点群がどのセルに含まれるかを判定することで, 近接点を高速に検索する手法も提案されている [Piegl02]. この手法は, ドローネ網を利用する方法と比べて高速である. 図 4 に例を示す. 探索の対象となる点群を緑色で, 注目する点を赤色で, 赤色の点の近接点を青色で示した.

２．３　３次元格子点集合のＣＧ技術

ボクセル表現された３次元医用画像や３次元テクスチャなど, ３次元正方格子点上でサンプリングされた３次元場（主にスカラ場）である. 格子点での値は, その点での物体の存在濃度を表すことが多い. ３次元画像処理の手法も適用可能である. レンダリングには, ボクセルからポリゴンモデルに変換してからポリゴンレンダリングを施すことが可能である. 図 5 はそのようにして生成された岩場形状のＣＧ画像である.

また, 直接ボクセル空間から画像を生成するボリュームレンダリング法も多く開発されている. ボリュームレンダリング法としては, レイキャスティング (Ray-Casting), スプラッティング（Splatting）（３．２節参照), シェアワープ（Shear-Warp), および, テクスチャマッピング（Texture Mapping）による方法などが開発されている. 雲などの画像生成に対しては, 散乱光のシミュレーションを含むレンダリング法も開発されているが, レイキャスティング型のレンダリング法は計算時間を多く必要とするため, 近年では, ビルボード型やテクスチャレンダリング型のグラフィックスボードを活用したレンダリング法が研究されている. 区別は付かないと思うが, 図 6 (a) はレイキャスティング型による画像, (b) はテクスチャレンダリング型による画像である.

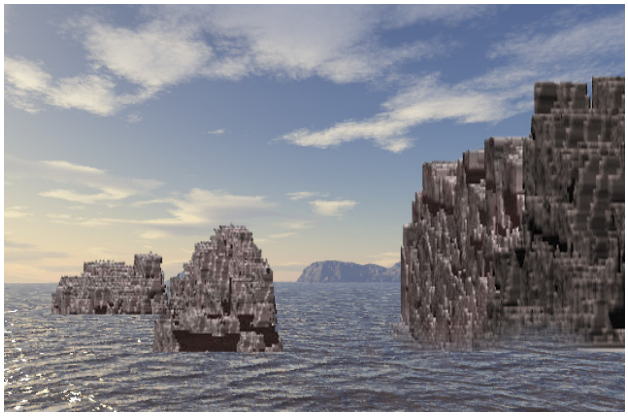
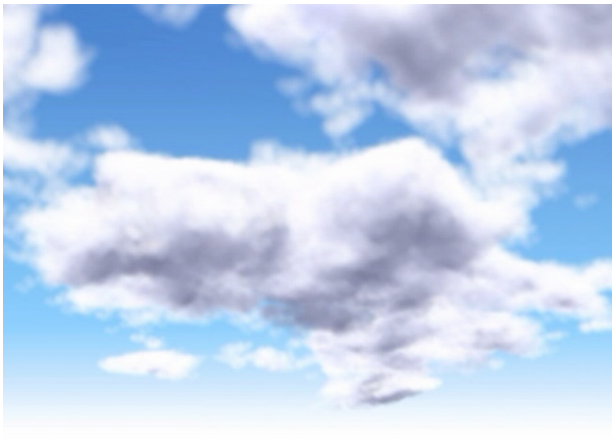


図 5　ボクセルからポリゴン化された岩場形状 [Ito03]

(a)　レイキャスティング型による雲の画像



(b)　テクスチャレンダリング型による雲の画像

図 6　雲のボリュームレンダリングの例

２．４　３次元非格子点集合のＣＧ技術

３次元非格子点集合は大別して次の２つに分けられる.

・ 物体表面のサンプリング点の集合

・ 物体内部のサンプリング点を含む点集合

前者には，レンジセンサによる複数のデプス画像から生成された点集合などが該当する. 図 7 では，建築物の左側と右側の２方向から計測したデプス画像に基づいた点集合を示している．また，後者には，粒子ベースのシミュレーション法における粒子群や，メタボールによるモデリングのような中心点の集合などが該当する.

前者，すなわち，物体表面上のサンプリング点集合のポリゴン化では，近傍点の探索が基本となるが，近傍点が必ずしも面上での近傍点とはならない場合もあるなど難しい問題がある．複数の２次元投影を作成し，その上でドローネ網を生成する方法も提案されている [安川 02]．また，点集合の３次元ボロノイ網を利用して，物体の中心軸（面）を構成する点も含めた３次元ドローネ網を生成し，物体の表面を取り出す手法も提案されている [Amenta98]．しかし，この手法はロバスト性に欠け，良好な結果が出ないケースも多い.

後者，すなわち，物体内部のサンプリング点を含む点集合のポリゴン化では，３次元ドローネ網に基づく方法が考えられるが，どの面が表面を構成するか，特別な判定基準がないと表面ポリゴンの選択は難しい.

また，いずれにおいても，点にメタボールを割り当て，その濃度分布をボクセル空間でサンプリングし，マーチングキューブ法や四面体格子法でポリゴン化する方法も考えられる．図 8 は，このような方法による粘土のＣＧ画像である. ただし，この方法では，生成されたポリゴンは元の点データの位置を通過しないので，物体表面上のサンプリングデータについては原理上の誤差の発生は避けられない.

点集合が雲や炎などのサンプリング点を表す場合には,

・メタボールとボクセルによりボリュームデータを生成し，レイキャスティングする [菊池 02].
・メタボール型の３次元テクスチャをマッピングし，テクスチャレンダリングする [太田 03].
・メタボール型の濃度分布の投影をビルボードで与え, ビルボードレンダリングする[Dobashi00].
・メタボール型の濃度分布でスプラッティングする.

などの方法がある．これらの中には一次散乱をシミュレーションするレンダリング法もある [太田 03，Dobashi00].（ただし，後者３つの手法はほぼ同一の手法であり，区別は明確ではない．後述するスプラッティングを参照.）図 9 は，３次元テクスチャによる爆発火炎のＣＧ画像である.

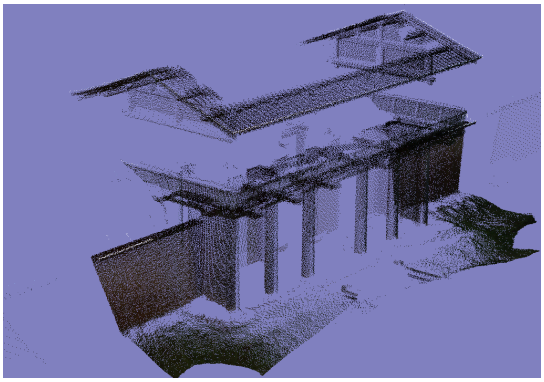


図 7　物体表面上のサンプリング点集合の例

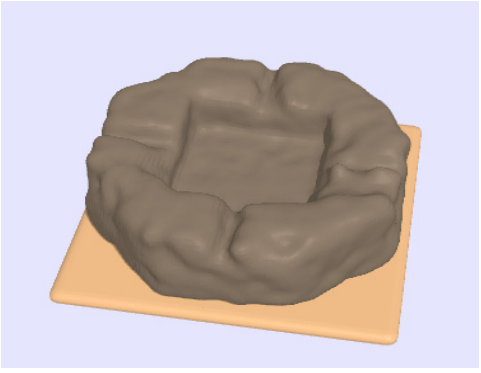


図 8　メタボールによりポリゴン化された点集合（粘土モデル）の例 [小田 01]

図 9　３次元テクスチャによる爆発火炎のＣＧ画像の例 [竹下 02，Takeshita03]

３．ポイントベースのレンダリング法

ここでは，ポイントグラフィックスにおけるレンダリング法として，最近キーワードとして用いられているいくつかの手法について解説する．なお，以下の文中では，説明の上で必要となる用語のうち，原論文中で英語で表現されている用語のいくつかについては，適宜，適当な日本語に置き換えて表記した.

一般に, ポイントベースのレンダリング法では，物体を点の集合とみなし，各点を個別の描画要素（primitive）としてレンダリングを行う. 例えば，物体が複雑な形状を持ちポリゴン化に向かない場合，あるいは，ポリゴン化すると個々のポリゴンがピクセルサイズよりも小さくなってしまうような詳細な形状データを扱う場合には，ポリゴンベースのレンダリングに比べ，ポイントベースのレンダリングが有効であると考えられる．一般に，点は，あらゆる幾何形状を構成する最も基本的な要素である．実際，文献 [Levoy85] では，点をさまざまな幾何モデルをレンダリングする際の共通の描画要素としてとらえ，(a) 幾何モデルから点集合への変換，(b) 点集合のレンダリング，という処理の流れにより，異なる種類の幾何モデルを共通のポイントベースのレンダリングパイプラインにより描画するというアイデアが提案されている.

３．１　後方マッピングと前方マッピング

ポイントベースレンダリングでは，スクリーン上のピクセルと３次元空間上の物体を構成する点との相対的な関係が重要となる．これに関連して，物体のスクリーンへの投影について，後方マッピング（backward mapping）と前方マッピング（forward mapping）という対比的な方法が考えられる [Westover90]．後方マッピングでは，スクリーンから物体に対してマッピングを行う．具体的には，スクリーン上のピクセルごとに，そのピクセルを通る視線を３次元空間上に投射することで，そのピクセルの描画に影響する物体上の点を求め，そのピクセルの描画色を決定する．この場合には，一般に，１つのピクセルに対して複数の点が影響し得る．一方，前方マッピングでは，物体からスクリーンに対するマッピング，すなわち，物体上の点をスクリーン上のピクセルに対して投射する．この場合には，通常，１つの点が複数のピクセルに影響を与える．ここで，点は数学的には０次元であり大きさを持たないが，描画時にピクセルを覆うためには有限の大きさを持たせなければならない．この描画時に点に与える大きさは，ピクセルの大きさとの相対的な関係から適切な値を選択する必要があり，ポイントベースレンダリングを考える上で重要な検討項目である．

３．２　スプラッティング

ポイントベースレンダリングの手法として，現在，最も良く用いられている基本的な手法の一つにスプラッティング（splatting）[Westover90] がある．これは，上記の前方マッピングに属するものであり，本来，３次元ボリュームデータをレンダリングする手法として提案された．この手法では，図 10 に示すように，描画要素となる３次元空間上の点を再構成核（reconstruction kernel）と呼ばれる球状のボリュームで囲むことで有限の大きさを持たせ，その再構成核の濃度と点の濃度との積を投影方向について積分したものをスクリーンに投影することで，その点を描画する．具体的には，スクリーン上で再構成核が投影される範囲を求め，その範囲内のピクセルごとに積分値を求める．実際には，あらかじめ再構成核を積分した結果（これを footprint と呼ぶ）をテーブルに登録しておき，レンダリング時にそのテーブルを参照する．その際，ガウス関数等により footprint をモデル化したものをテーブルに登録しておくこともできる．再構成核の形状と大きさは生成画像の品質にとって重要であり，小さすぎる場合には物体の表示に穴が開き，大きすぎる場合には画像がぼやけてしまうため，適切なものを選択する必要がある．また，レンダリング時には，近傍の点の情報を利用する等によりシェーディングを行うことも可能である．なお，この手法でスクリーンに投影される再構成核で囲まれた３次元空間上の点のことをスプラット（splat）と呼び，また，再構成核がスクリーン上に投影されたものである footprint のことを kernel と呼ぶ場合もある．

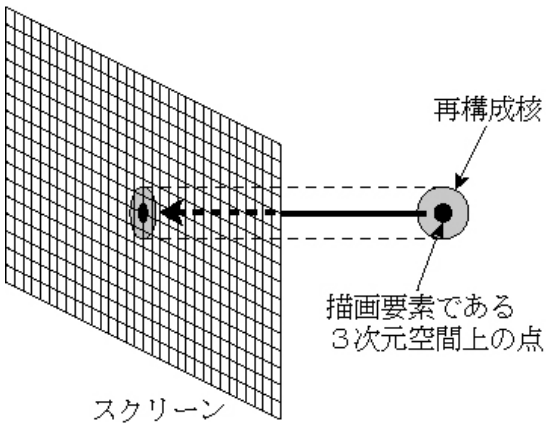


図 10　スプラッティング

３．３　イメージベースレンダリング

ポイントベースレンダリングは，与えられた参照画像をもとに任意視点からの出力画像を生成する，いわゆる，イメージベースレンダリング

（Image-based Rendering）と関連が深い．イメージベースレンダリングでは，多くの場合，参照画像中のピクセル値から目的とする出力画像中のピクセル値を求める必要がある．これは，参照画像中の“ピクセル”を物体上の“点”とみなせば，ポイントベースレンダリングとして考えることができる．このピクセルからピクセルへのマッピングは，前述の後方マッピングならびに前方マッピングのいずれの方向でも考えることができ，実際，スプラッティングを用いた手法もいくつか見られる．また，これらのマッピングのことをワーピング（warping）と言うことも多い．

３．４　層化デプス画像

スプラッティングを用いたイメージベースレンダリングの一手法として層化デプス画像（LDI：Layered Depth Image）[Shade98] がある．LDIは，３次元空間上の任意に決めた視点位置とスクリーンに対して，そのスクリーン上のピクセルごとに，そのピクセルに投影される（そのピクセルを通る視線が交差する）全ての物体上の点に関する色値とデプス値の組（レイヤ要素）をリストとして持たせたものである．このリスト中の最小のデプス値のみを保存したものが通常のＺ－バッファである．図 11 に LDI の例を示す．LDI はＣＧモデルおよび実写画像のいずれからも生成可能である．ＣＧモデルを用いる場合には，個別に生成した複数の参照デプス画像の各ピクセルを LDI の視点とスクリーンに合わせてワーピングする，あるいは，レイトレーシング法やＺ－バッファ法で各ピクセルのサンプリング点を計算した際に全デプス値を保持する，といった方法で生成できる．また，複数の実写画像を用いる場合には，コンピュータビジョンの手法により実写画像間のピクセルどうしの対応関係およびデプス値を推定することで LDI を生成する．レンジセンサ等により直接的に実写のデプス画像が獲得できる場合には，LDI の生成はより容易になる．

LDI から任意視点の出力画像を生成する際には，増分ワーピングアルゴリズム（incremental warping algorithm）と呼ばれるアルゴリズムが用いられる．これは，LDI の各ピクセルが持つレイヤ要素の (x, y, z) を出力画像の座標系の (x, y, z) へと変換する座標変換行列を分解し，スキャンライン順での座標変換を逐次的な値の加算によって実行可能にしたものであり，変換の高速化が図られる．さらに，このアルゴリズムにはマクミランの順序付けアルゴリズム（McMillan's ordering algorithm ） [McMillan95a ， McMillan95b] が利用されており，座標変換された各レイヤ要素が出力画像上で可視かどうかを奥行きソート等で判定することなく，変換順での上書きで正しい出力画像が高速に生成される．また，各レイヤ要素はスプラッティングにより描画される．このため，レイヤ要素には，色値とデプス値に加えてスプラットに関する情報も与えておく．

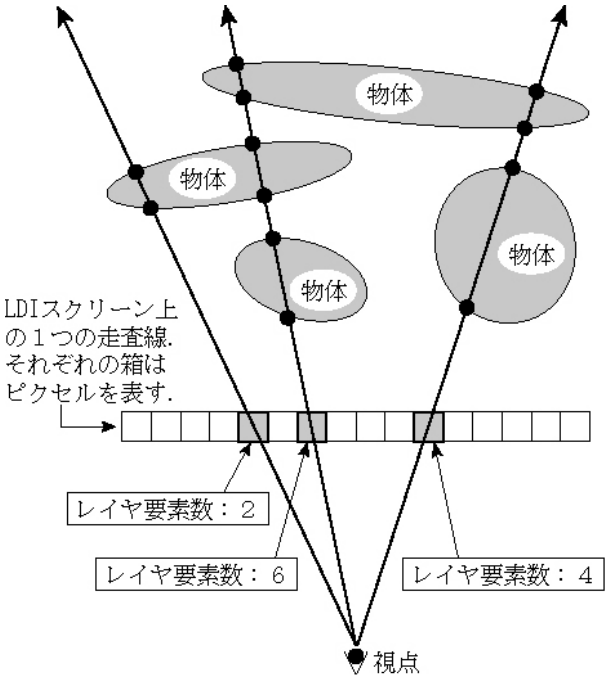


図 11　層化デプス画像（LDI）

LDI による出力画像の生成を考えた場合，さまざまな視点位置への対応のため，３次元空間上のできるだけ多くの位置と方向をカバーする複数の LDI を生成しておくことが望ましい．特に，立方体状の領域を考え，その各軸方向に関する直交する３つの LDI からなる LDC（Layered Depth Cube）[Lischinski98] は，有効な形式の一つである．また，物体を囲む３次元空間を８分木構造により再帰分割し，その各ノードに LDI を割り当てることで LDI を階層構造化し，３次元空間上での位置と参照画像の解像度に応じた適切な LDI ピクセルのサンプリングを実現したものが，LDI tree [Chang99] である．通常の LDI が画像中心のデータ表現法であるのに対して，LDI tree は物体中心の表現法であると言える．これにより，出力画像の視点やスクリーン位置および解像度に合わせた，より効率的なレンダリングが可能となる．

３．５　サーフェル

LDI tree と同様，LDI を利用した階層構造によりポイントベースで描画対象とする物体を表現する方法にサーフェル（surfel：surface element）[Pfister00] がある．サーフェルでは，８分木の各ノードに LDC を割り当てた LDC tree を用いる．１つのサーフェルはレンダリングのための幾何情報とシェーディング情報を与えた物体上の１点として定義され，物体上の全サーフェルを LDC tree により管理することで効率的なレンダリングが実現される．サーフェルを管理する LDC tree の例を図 12 に示す．ただし，この図は，便宜上２次元で表現，すなわち，３軸方向の LDI のうちの２つ（図中の LDI-1，LDI-2）について，それらのスクリーンを残りの１つの LDI スクリーンに平行な面で切断した断面を表す． この LDC tree の構築のためには，まず，物体全体を囲む立方体領域を考え，その３軸方向に関する LDI ピクセルのサンプリングにより LDC を生成し，これを最大解像度（レベル 0）の LDC とする（図 12 (a) 参照）．このとき，３つの LDI 中の各ピクセルが持つ個々のレイヤ要素がそれぞれ物体上の１点，すなわち，１つのサーフェルとされる．図 12 (a) では，LDI-1 と LDI-2 に関するピクセルごとのサンプリングをそれぞれ青と赤の矢印線で示し，緑色で示す物体の表面上の青丸と赤四角がそれぞれの矢印線によってサンプリングされたサーフェルを示す．そして，このレベル 0 の LDC から開始し，その３つの LDI のスクリーン解像度（サンプリングするピクセル数）をスクリーンの２軸方向のそれぞれで 1/2 ずつ再帰的に減らしていくことで，最大解像度の $(1/2)^n$ の解像度を持つレベル n = 1, 2, 3, … の LDC を生成していく（図 12 (b), (c) 参照）．続いて，あらかじめ決めた値 b を用いて，各レベルの LDC を一辺が b 個のピクセルを持つ小立方体領域（ブロック）の集合に細分割する．ここで，１つのブロックは，それぞれが b^2 個のピクセルを持つ３つの LDI から構成される１つの LDC となる．図 12 は b = 4 の例であり，(a), (b)，および，(c) に示す各レベルごと，互い違いに色の濃淡を変えた四角領域で各ブロックを表す．レベル n が大きくなるに従って空間全体のブロック数は 1/8 ずつ減っていき，これにより，各レベルのブロックをノードとした８分木構造を持つ LDC tree が構築される．このとき，各ブロックの LDC が持つ３つの LDI を 3-to-1 reduction と呼ぶ操作で１つの LDI に変換することも可能である．この操作は，出力画像の品質を多少悪化させてしまうという欠点はあるものの，データ量の軽減とレンダリング速度の向上を図ることができる．

以上の LDC tree のための物体形状のサンプリングと同時に，テクスチャのサンプリングを行う．具体的には，物体上の各サーフェルの位置で物体表面に接する円盤（ディスク）を考え，これをテクスチャ空間に楕円盤としてマッピングして

EWA フィルタ（Elliptical Weighted Average filter）[Heckbert89] によりフィルタリングすることで，そのサーフェルに与えるテクスチャ値を求める．このとき，円盤の半径をいくつか変えてマッピングすることで，１つのサーフェルに複数のテクスチャ値を与えたサーフェルミップマップ（surfel mipmap）を生成する．

以上により生成された LDC tree とサーフェルミップマップにより効率的なレンダリングが可能となる．レンダリング時には，まず，LDC tree をトップダウンで（低解像度から高解像度へ）トラバースし，ブロック単位でのビューフラスタムカリング（view-frustum culling）とバックフェースカリング（backface culling）を行う．続いて，同様に，LDC tree のトップダウンのトラバースにより，描画に最適なレベルのブロック，具体的には，１ピクセルに対して適切な数のサーフェルが投影されるブロックを選択する．この１ピクセルあたりのサーフェル数を調整することで，描画速度と画像品質の制御が可能となる．その後，選択された各ブロック内のサーフェルについて，可視性判定のためのスプラッティング（visibility splatting）の実行，ミップマップレベルの決定によるテクスチャ値の算出，シェーディング処理，および，サーフェルのスクリーン投影による出力画像の生成を行うことで，レンダリングが完了する．

なお，物体の具体的なサンプリング方法とサンプリング点の管理のしかたは異なるが，上記のサーフェルに類似したデータ表現法として，物体の存在する３次元空間をボクセル空間で離散化し，ボクセルごとに，その内部に存在する物体要素（サブポリゴン）に関する密度情報とシェーディング情報を保持する異方性３次元テクスチャ法が提案されている［村岡 99］．

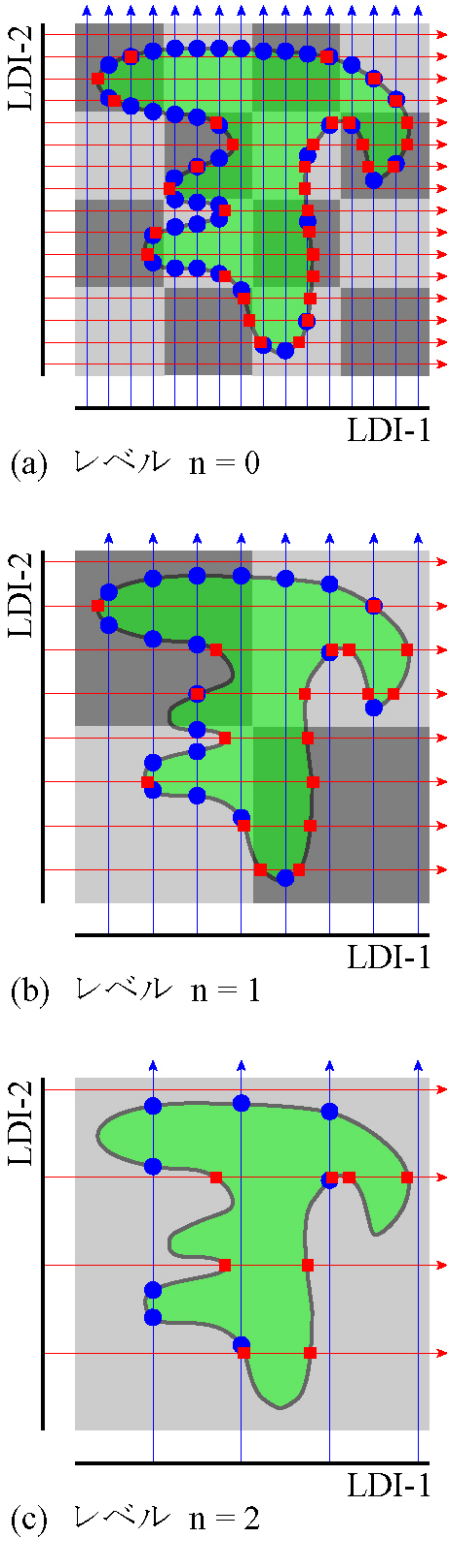


図 12　サーフェルの LDC tree（２次元で表現）

３．６　サーフェイススプラッティング

上にも述べた EWA フィルタは，点集合で表現された物体表面へのテクスチャマッピングを行うための有用な手法の一つである. これに関連して，従来の物体空間ベースの EWA フィルタ（source space EWA filter）を改良し，スクリーン座標系での効率的なテクスチャサンプリングが可能となるスクリーン空間ベースの EWA フィルタ（screen space EWA filter）を提案し，点集合で表現された物体表面を，テクスチャマッピングを施して効率的にレンダリングする手法がサーフェイススプラッティング（surface splatting）[Zwicker01] である．この手法では，まず，物体を構成する点 $\mathbf{P}_k$ ごとに, その物体表面上の近傍でのローカルな２次元座標系（物体座標系）$\mathbf{u}$ に関する基底関数 r_k，ならびに，テクスチャのための係数 w_k（実際にはＲＧＢ成分あり）を与え，それらを近傍の点で重み付け加算して得られる物体座標系 $\mathbf{u}$ に関するテクスチャ関数 f_c を考える. 通常のテクスチャマッピングでは，スクリーン上のピクセルごとに，そのスクリーン座標値 $\mathbf{x}$ に対応する物体上の物体座標値 $\mathbf{u}$ を求め，それに対応するテクスチャ値 $f_c(\mathbf{u})$ をサンプリングする. 一方，スクリーン空間ベースの EWA フィルタでは，スクリーン座標値 $\mathbf{x}$ について直接サンプリングを行う．具体的には，まず，各基底関数 r_k をスクリーンに投影し, ローパスフィルタを適用することで，スクリーン座標系 $\mathbf{x}$ に関する再サンプリング核（resampling kernel）ρ_k を求める．そして，その再サンプリング核 ρ_k と係数 w_k を近傍の点で重み付け加算することで，スクリーン座標系 $\mathbf{x}$ に関するテクスチャ関数 g'_c を求める．そして，最終的に，ピクセル座標値 $\mathbf{x}$ ごとにテクスチャ値 $g'_c(\mathbf{x})$ をサンプリングする. 物体座標系 $\mathbf{u}$ に関して再サンプリング核を定義する従来の EWA フィルタでは，ピクセルごとのサンプリングのためにスクリーンから物体への後方マッピングが必要であり，多くの計算時間を要する．一方，スクリーン空間ベースの EWA フィルタでは，スクリーン座標系での直接的なサンプリングにより効率的なレンダリングが可能となる．さらに，基底関数とローパスフィルタにガウス関数を用いた場合には, 再サンプリング核もガウス関数となり，計算の効率化が図られる．また，この手法では，フレームバッファのレイヤ化による点の描画順序に依存しないαブレンディング，ならびに，ピクセルごとの再サンプリング核の値の総和に基づくα値の算出によるエッジのアンチエリアシングも提案されている.

３．７　Ｑスプラット

先にも述べたように，ポイントベースレンダリングが有効であると考えられるケースの一つは，ピクセルサイズに比べて詳細な形状データを扱う場合であるが，これに関連して，大規模なメッシュデータを多重解像度により効率的にポイントベースでレンダリングする手法に Q スプラット（QSplat）[Rusinkiewicz00] がある．この手法では，対象とするメッシュ上の頂点をリーフ（末端のノード）とした木構造として，バウンディング球の階層構造（bounding sphere hierarchy）を構築する．この木構造中の各ノードは，各階層ごとの適切な詳細度でメッシュ形状を表現する点を囲むようなバウンディング球であり，親ノードのバウンディング球が子ノードのバウンディング球を包含する．実際に，各ノードはデータとして，バウンディング球の中心位置と半径，メッシュに対する法線，法線錐体（normal cone）に関する情報，ならびに，木構造を探索するための情報を持ち，オプションとして色情報を持つ．これらの情報は, 親子間での相対化やテーブルの利用など，それぞれが適切な方法で量子化されることでデータ量の削減が図られる. また, これらのノードは，木構造のルートからリーフに向かう順序，すなわ

ち，低解像度から高解像度に向かう順序に幅優先順でデータ保持される．これにより，レンダリング時には，低解像度から高解像度へ漸進的にロードすることで大規模データの効率的な表示が可能となる．実際のレンダリング方法としては，ルートからリーフの方向へ木構造中のノードを再帰的にたどり，ノードの持つバウンディング球のスクリーン上への投影像の大きさがしきい値以下になったところで，そのノードを描画する．その際，バウンディング球によるビューフラスタムカリング，ならびに，法線および法線錐体によるバックフェースカリングにより，不必要なノードの処理を避ける．また，スクリーン上へのノードの描画にはスプラッティングが用いられる．このとき，スプラットの形状（四角形，円盤，ガウス関数等）や大きさ，あるいは，ノードの持つ法線とスクリーンとのなす角度によってスプラットを傾けるかどうか等を変更することにより，生成画像の品質および描画速度を制御することができる．

４．おわりに

本稿では，最近のキーワードであるポイントグラフィックスについて，点集合を扱うこれまでのＣＧ技術と，主にポイントグラフィックスとして最近開発された技術について解説し，“ポイントグラフィックス”はこれまでのＣＧ技術の自然な発展にあることを示した．

レンダリングにおいては，点集合によれば，必要な画像解像度に適する密度でのサンプリングデータを得ることが容易であり，単純な LOD 技術の開発も可能である． このことは，特に膨大なポリゴンを必要とする複雑な物体の取り扱いにおいて大きな効果を発揮すると思われる．

いずれにせよ，これまでのポリゴングラフィックスとボリュームグラフィックス双方に関係が深く，統一的に扱えるレンダリング技術の開発に貢献するものとして，大きな期待を感じさせてくれる．

筆者らも“ポイントグラフィックス”に関係の深い研究を進めてきており，この分野へ少しでも貢献できればと考えている．

謝辞

本稿で紹介した研究の一部は「夢県土いわて戦略的研究推進事業」の支援による．

参考文献

[樫山 01] 樫山，藤代，“サーフェルによる連続体／不連続体の統一的表現”，情報処理学会研究報告「グラフィクスと CAD」，No.104–013，pp.53–58，2001.

[土橋 02] 土橋，山本，西田，“ポイントサンプルジオメトリのため相互反射計算法”，Visual Computing／グラフィクスとＣＡＤ合同シンポジウム予稿集，2002.

[岡根 02] 岡根他，“不規則点群の周波数解析”，情報処理学会研究報告「グラフィクスと CAD」，No.109–003，pp.11–16，2002.

[川田 03] 川田，金井，“イメージベースドポイントレンダリングによるレンジスキャンデータの描画”，情報処理学会研究会報告「グラフィクスと CAD」, No.112–003, pp.13–18, 2003.

[古舘 01] 古館守通，渡辺孝志，“単調関数補間を用いた等高線データからの DEM 生成”，GIS －理論と応用，9 巻，1 号，pp.19–27，2001.

[Luebke01] D. Luebke, “Developer’s Survey of Polygonal Simplification Algorithms,” *CG&A*, Vol.21, No.3, pp.24–35, 2001.

[Shade98] J. Shade, S. J. Gortler, L. He, and R. Szeliski, “Layered Depth Images,” *SIGGRAPH 98*, pp.231–242, 1998.

[Lischinski98] D. Lischinski and A. Rappoport, “Image-Based Rendering for Non-Diffuse Synthetic Scenes,” *Rendering*

Techniques '98, pp.301–314, 1998.

[Piegl02] L. A. Piegl and W. Tiller, “Algorithm for finding all k nearest neighbors,” *Computer Aided Design*, Vol.34, No.2, pp.167–172, 2002.

[Ito03] T. Ito, T. Fujimoto, K. Muraoka, and N. Chiba, “Modeling Rocky Scenery Taking into Account Joints,” *CGI2003*, pp.244–247, 2003.

[安川 02] 安川, 金谷, 眞鍋, 千原, “3 次元点群データの面張りに関する研究”, ビジュアルコンピューティングワークショップ予稿集, 2002.

[Amenta98] N. Amenta, M. Bern, and M. Kamvysselis, “A New Voronoi-Based Surface Reconstruction Algorithm,” *SIGGRAPH 98*, pp.415–421, 1998.

[小田 01] 小田, 村岡, 千葉, “仮想粘土の粒子ベース・ビジュアルシミュレーション”, 情報処理学会論文誌, Vol.42, No.5, pp.1142–1150, 2001.

[菊池 02] 菊池, 藤本, 村岡, 千葉, “雲粒子の熱交換モデルによる積乱雲のビジュアルシミュレーション”, 芸術科学会 NICOGRAPH 春季大会, pp.17–18, 2002.

[太田 03] 太田, 田村, 藤田, 藤本, 村岡, 千葉, “一次散乱光を考慮した OpenGL ベース・3D テクスチャレンダリング―3D テクスチャ内部に存在する点光源への対応―”, 芸術科学会 NICOGRAPH 春季大会, pp.85–86, 2003.

[Dobashi00] Y. Dobashi, K. Kaneda, H. Yamashita, T. Okita, and T. Nishita, “A Simple, Efficient Method for Realistic Animation of Clouds,” *SIGGRAPH 2000*, pp.19–28, 2000.

[竹下 02] 竹下, 太田, 田村, 藤本, 村岡, 千葉, “爆発火炎の粒子ベースビジュアルシミュレーション法”, 芸術科学会第 18 回 NICOGRAPH 論文コンテスト, pp.115–120, 2002.

[Takeshita03] D. Takeshita, S. Ota, M. Tamura, T. Fujimoto, K. Muraoka, and N. Chiba, “Particle-Based Visual Simulation of Explosive Flames,” *Pacific Graphics 2003*, pp.482–486, 2003.

[Levoy85] M. Levoy and T. Whitted, “The Use of Points as Display Primitives,” Technical Report TR 85–022, University of North Carolina, 1985.

[Westover90] L. Westover, “Footprint Evaluation for Volume Rendering,” *SIGGRAPH 90*, pp.367–376, 1990.

[McMillan95a] L. McMillan, “Computing Visibility Without Depth,” Technical Report 95-047, University of North Carolina, 1995.

[McMillan95b] L. McMillan, “A List-Priority Rendering Algorithm for Redisplaying Projected Surfaces,” Technical Report 95–005, University of North Carolina, 1995.

[Chang99] C. F. Chang, G. Bishop, and A. Lastra, “LDI Tree: A Hierarchical Representation for Image-Based Rendering,” *SIGGRAPH 99*, pp.291–298, 1999.

[Pfister00] H. Pfister, M. Zwicker, J. van Baar, and M. Gross, “Surfels: Surface Elements as Rendering Primitives,” *SIGGRAPH 2000*, pp.335–342, 2000.

[Heckbert89] P. Heckbert, “Fundamentals of Texture Mapping and Image Warping,” Master's thesis, University of California at Berkeley, Department of Electrical Engineering and Computer Science, 1989.

[村岡 99] 村岡, 千葉, “異方性３次元テクスチ

ャ法”，画像電子学会誌，Vol.28，No.2，pp.131–139，1999.

[Zwicker01] M. Zwicker, H. Pfister, J. van Baar, and M. Gross, “Surface Splatting,” *SIGGRAPH 2001*, pp.371–378, 2001.

[Rusinkiewicz00] S. Rusinkiewicz and M. Levoy, “QSplat: A Multiresolution Point Rendering System for Large Meshes,” *SIGGRAPH 2000*, pp.343–352, 2000.