Vol. 46 No. 1 情報処理学会論文誌 Jan. 2005

テクニカルノート

Sequential Point Clusters：大規模モデルに対する効率的なポイントベーストレンダリングシステム

岡本 泰英† 山崎 俊太郎†† 池内 克史†

モデリング技術の向上により，大規模な3次元モデルの生成が容易になった反面，生成された大規模モデルを効率的に描画することは困難になっている．そこで本論文では，Point Based Rendering による効率的な描画システムの考察をし、Sequential Point Trees による多重解像度描画アルゴリズムにデータの幾何的な位置によりクラスター化することで，描画データ量を削減する Sequential Point Clusters への拡張を提案する．実際に，面数が数千万の大規模なモデルを用いて本手法の実用性を検証するとともに，最適なクラスターサイズを探りその有効性を明らかにする．

Sequential Point Clusters: Efficient Point-based Rendering Method for Huge 3D Models

Yasuhide Okamoto,† Shuntaro Yamazaki†† and Katsushi Ikeuchi†

Advancement in modeling technologies has enabled us to obtain very huge 3D models. But the conventional rendering methods are not rapid enough for rendering such models interactively. In this paper, we verify a system for effectively rendering models based on using the Point-Based Rendering technique. In addition, we present the extension of Sequential Point Trees to Sequential Point Clusters. This method is accomplished by using positional clustering, which can reduce the amount of processing data. We have verified the efficiency of our proposed method by rendering very huge models, and demonstrated the superiority of clustering over other methods.

1. はじめに

コンピュータビジョンの応用例に実物体の3次元デジタルモデル化がある．近年では計測技術等の発展により非常に高精細な3次元モデルの生成が可能となっているが，これは同時にデータ量の増大を促し，グラフィックス描画が困難になる等の問題が起きている．

CG の分野で3次元モデルの描画は，モデルをポリゴン（三角形）により描画するポリゴンレンダリングが最も使われているが，描画処理の複雑さのためポリゴン数の増大に伴い描画速度は非常に低くなる．

大規模モデルの効率的描画法の一つに，描画シーンごとに適切な詳細度のモデルを描画する level-of-detail（LOD）がある．また，描画単位にポリゴンではなく処理の軽いポイントを用いたポイントベーストレンダリング[1] も提案されている．高密度なモデルではポイントでも十分な画質での描画が可能である．

我々はこれらの先行研究を踏まえ，GPU を用いたポイントベーストレンダリング手法による，従来よりも効率的な描画法を提案する．

2. 提案手法

本手法は QSplat[2] と Sequential Point Trees (SPT)[3] の手法を基礎としている．それに加え，点群のクラスタリングを行うことで SPT の描画処理における枝狩りを効率化する．本章ではまず先行研究のアルゴリズムを説明し，提案手法を述べることとする．

2.1 先行研究

QSplat は多重解像度描画のためにモデルを点群の木構造に変換する．モデルのポリゴンを位置・法線・半径等の情報を持つポイント（点）に変換し，できた

† 東京大学
The University of Tokyo
†† 産業技術総合研究所
National Institute of Advanced Industrial Science and Technology

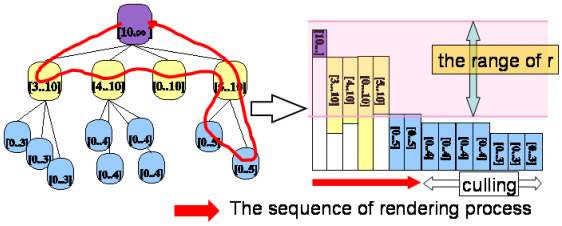


図 1 階層構造から 1 次元リストへの変換
Fig. 1 Sequentialization of a hierarchy structure

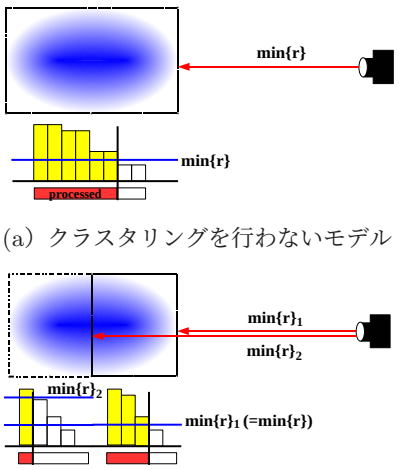
点群データを葉とする木構造をボトムアップに生成する．近隣の複数の点を包含する点を定義してこれらの親とし，これを再帰的に繰り返すことで点群から木構造を生成する．描画の際には、QSplat ではこの木構造を視点の変化の度に探索する．探索中の点の描画サイズが適切な詳細度になる場合，具体的には描画サイズが閾値 ϵ より小さくなる場合にその点を描画しバックトラックして探索を続ける．そうでなければ子を再帰的に探索する．こうして適切な解像度で描画できる．

SPT ではこの再帰的な探索処理を次のように逐次処理へと変換する (図 1)．逐次化された判定処理は GPU 上で実装が可能となり，その性能を生かした多重解像度描画が可能になる．QSplat の探索の際の描画判定式は，点の描画サイズは半径 R とその中心から視点までの距離 r により $\frac{R}{r}$ で表されることから，$\frac{R}{r} < \epsilon$ となる．各々の点の半径と閾値は不変なので，判定は r にのみ依存し，式は $r > \frac{R}{\epsilon}$ と置換できる．$\frac{R}{\epsilon}$ はその点を描画する r の下限なので r_{min} とする．さらにこの点が描画されるのは，親の点でこの判定が偽となる場合のみなので，親の点の r_{min} を r_{max} と定義すると，この点が描画される r の範囲は $r_{min} < r \leq r_{max}$ となる．つまりそれぞれの点に対し r_{min}, r_{max} を予め計算し保持させることで，木の探索ではなく 1 次元点群リストを逐次的に判定することで描画ができる．

SPT ではさらに点群リストを r_{max} の大きい順にソートすることにより，判定処理の効率化をする．このソートは，モデル中の全ての点で取りうる r の最小値 $\min r$ よりも r_{max} が小さい点の描画判定をまとめて省略することを可能にする．これにより，小さ過ぎる点の枝狩りをすることができる．

2.2 Sequential Point Clusters

SPT では，過度の枝狩りを防ぐため，r の最小値 ($\min r$) を閾値としてモデル全体の枝狩りをしているが，この方法だと視点から遠い部分であっても近い部分と同程度の枝狩りしかできず非効率的である．提案手法はこの SPT の欠点を克服し，より多くの点を枝狩りすることを目指す．そしてこの手法を Sequential Point Clusters（SPC）と名づける．



(a) クラスタリングを行わないモデル

(b) 2 個のクラスターに分割したモデル

図 2 クラスタリングによる枝狩りの効果．モデル下の棒の列は r_{max} でソートした点群リスト，黄色の棒は枝狩りで残った点を表す．
Fig. 2 The prunning of positional clustering. The row of bars is the list of points, and yellow poles are rest points after prunning.

2.2.1 位置情報によるクラスタリング

点群を位置関係によりクラスタリングする手法について説明する．

クラスタリングは木構造生成の際に行う．木構造をボトムアップに生成する過程で，近隣の点群を包含する新たな点を生成し，それを親とする処理をするが，同時にその点の包含する点（木構造中でその点の子孫となる点）の総数を調べ，それが規定値以上ならばその点群を 1 つのクラスターとする．これにより点群は近隣の点でまとまったクラスター群へと分割できる．

このクラスタリングにより枝狩りを効率性することが可能で，その詳細を 2.2.2 句で述べる．

2.2.2 クラスタリングの効果

クラスタリングした際の枝狩りの効率化の様子を図 2 に示す．まず，クラスタリングを行わない場合 (図 2(a)) では，モデル 1 つに対し，$\min r$ のみを閾値として枝狩りをする．そのため 1 つのモデルにおいて視点に近い部分も遠い部分も枝狩り率は同じである．遠い部分については描画されない小さな点が近い部分に比べて多く，枝狩りには改良の余地があるといえる．次にモデルを 2 分割した場合 (図 2(b)) を考える．枝狩りの閾値は，各々のクラスターに対して定義し視点からクラスターへの最小距離 $\min r_i$ と設定する．視点から遠いクラスターの閾値は近いクラスターのそれに比べ大きな値となり，視点から遠いクラスターほど多

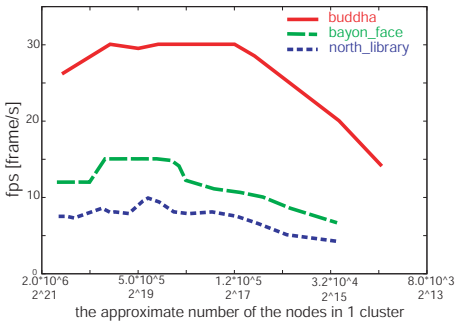


図 3 クラスターサイズに対する描画速度の変化:入力モデルのサイズは上から 1M，6M，9M ポリゴン
Fig. 3 The performance to the size of clusters:the numbers of polygon are 1M, 6M, 9M from upper.

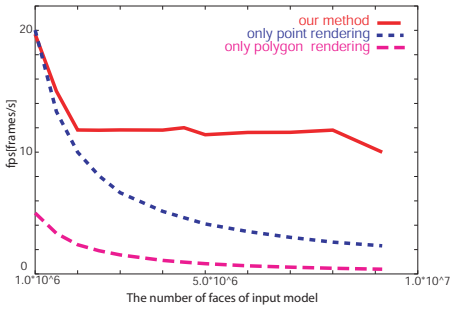


図 4 各手法による入力モデルのサイズに対する描画速度の変化
Fig. 4 The performance to the size of the input model by several methods.

くの点を枝狩りできる．視点から最も近いクラスターの枝狩り率は，クラスタリングをしない場合のそれと等しい（$\min\{\min r_i\} = \min r$ であるため）．そのため，クラスタリングをした場合の総枝狩り率はしない場合より大きくなることがいえる．

SPT のアルゴリズムのボトルネックは描画判定処理や実際の描画処理といった GPU 上での処理にある．枝狩りの効率化により GPU に送る点の数を減らすことができるため，クラスタリングの意義は大きい．

図 2 の議論を再帰的に行えば，枝狩り効率は細かくクラスター化するほど上がる．しかし細かいデータの増加でオーバーヘッドも増えるため，過度に細かくするべきではないと思われる．次章ではクラスターサイズと描画速度の関係について実験・考察を行う．

3. 実験結果

本手法は CPU は Intel Pentium4 の 3.2GHz，GPU に ATI RADEON9800XT という環境で実験し，プログラミングには C++及び DirectX を使用した．

本節では，大規模モデルの描画速度を測定とクラスター化による描画速度の変化について考察を行う．

3.1 クラスタリングによる効率性の検証

次の実験では，クラスターのサイズに対する描画速度の変化を測定した．

図 3 にクラスターサイズ（クラスターが含む点数）に対する描画速度の変化を 3 つのモデルについて示す．このグラフから，各々のモデルの描画速度にピーク部分が存在することが分かる．

なぜ描画速度にピークがあり，その後減少するのか原因を考察する．細かくクラスター化することは，逐次性が低下することを意味する．GPU は同じ処理をあるデータ配列の要素に対し逐次的に適用する，といったタスクには非常に有利である．しかし，クラスタリングによりデータを細かく分割することにより逐次性が下がり，その有用性が生かされなくなる．これは同時に，この処理を 1 度行う際にかかるオーバーヘッドの時間が比例して増加することも意味する．特に GPU へのデータ通信処理は，現在の性能では非常にコストが高いため，その回数の増加によるオーバーヘッドが総描画時間に与える影響は大きい．このような理由により速度低下が起こると考えられる．

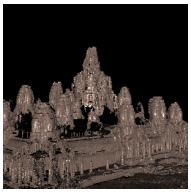
そのため，より良いパフォーマンスを得るには，適切なクラスターサイズになるようなクラスタリングをする必要がある．グラフより，ピーク部分は 1 クラスターの点数が約 500k の所で出現している．つまり，クラスターサイズがおよそ 500k 個の点を持つようにクラスタリングするのが最適であり，この時のパフォーマンスはクラスタリングしない場合（図 3 でグラフの一番左）よりも描画速度が高いことが示された．グラフからはその速度はおよそ 20% 上昇しており，クラスタリングすることの有用性を示す結果となった．

ピーク範囲は入力サイズが大きくなる程狭くなっているが，これはモデルサイズが大きいほど，クラスターサイズを変えた際のクラスター数の変化が大きく，それに伴い速度変化も大きくなるためと思われる．グラフにない，より大きなモデルについてもピークは狭いものの存在した．

3.2 描画速度

図 5 に 4 つのモデルのデータ生成時間と，同一解像度での描画速度を示す．ポリゴン数が 10M を超えるモデルでも毎秒 10 フレームを達成している．また，提案手法と従来の単純な描画法（多重解像度を用いないポイント・ポリゴン描画）の，描画速度の変化を図 4 に示す．このグラフから従来の描画法はサイズの増大で速度が一様に減少するが，提案手法は速度の減少が 10fps 付近で緩やかになっている．これは多重解像度の採用により微細な点を省いた効果といえる．






model	happy buddha	north library（Bayon）	towers （Bayon）	3towers（Bayon）
Input polygons	1,087,716	9,162,909	13,939,070	18,132,893
Preprocessing time	42.8s	201.0s	377.2s	471.5s
Rendering time	19.7fps	14.5fps	15.0fps	10.0fps

図 5 描画結果およびパフォーマンス
Fig. 5 Rendering results and performance

4. ま と め

本論文では大規模なモデルを高速に描画するためのシステムを提案した．その具体的手法として先行研究である SPT を拡張し，点群のクラスタリングをすることで枝狩り処理を効率化した Sequential Point Clusters を提案した．これは同時に，SPT では行われていなかった画面外の点の除去などのさらなる効率化が可能になっている．ただし，クラスター数の過度な増大は逐次性の低下によるコストも増大させるため，適度なクラスターサイズにする必要があり，実験により 1 クラスターの点数が約 500k 個の時に最適となることが分かった．同時にその時の描画速度は SPT の場合に比べて，約 20% 高くなるという結果が得られた．

しかし，本手法のデータ構造は，QSplat で行われている木構造を利用した圧縮法の実現が困難になっている．大規模データを扱うという目的からデータ圧縮の必要性は高いため，これは将来の課題とする．

参 考 文 献

1) M. Levoy and T. Whitted. The use of points as a display primitive. Technical report, University of North Carolina at Chapel Hill Technical Report TR 85-022, 1985.
2) Szymon Rusinkiewicz. Marc Levoy. Qsplat:a multiresolution point rendering system for large meshes. In *Proc. SIGGRAPH 2000, Computer Graphics Proceedings*, pages 343–352, 2000.
3) Carsten Dachsbacher. Christian Vogelgsang.Marc Stamminger. Sequential point trees. In *Proc. SIGGRAPH 2003, Computer Graphics Proceedings*, pages 657–662, 2003.

(平成 17 年 1 月 20 日受付)
(平成 17 年 1 月 20 日採録)

岡本 泰英

2004 年東京大学理学部情報科学科卒業．同年東京大学大学院情報理工学系研究科コンピュータ科学専攻修士課程入学，現在に至る．主にコンピュータグラフィックスに関する研究に従事．

山崎俊太郎（正会員）

1999 年東京大学理学部情報科学科卒業, 2004 年東京大学大学院情報理工学系研究科コンピュータ科学専攻博士課程修了, 同年より独立行政法人産業技術総合研究所研究員．博士（情報理工学）．電子情報通信学会, 情報処理学会, IEEE, ACM 各会員

池内 克史（正会員）

1973 年京都大学工学部機械工学科卒業．1978 年東京大学大学院工学研究科情報工学専攻博士課程修了．工学博士．MIT 人工知能研究所，電総研，CMU 計算機科学部を経て，1996 年より東京大学生産技術研究所教授．人間の視覚機能，明るさ解析，物体認識，人間による組立作業の自動認識などの研究に従事．論文賞（ICCV90, CVPR91, AIJ92, ロボット学会誌97, IEEE R&A 誌98, MIRU2000, 日本 VR 学会誌00）受賞．電子情報通信学会，人工知能学会，日本ロボット学会，日本 VR 学会，OSA，IEEE（Fellow）各会員．