芸術科学会論文誌 Vol. 9 No. 2 pp. 73-84

複数 LDI を用いた半透明点群の効率的な描画法

吉田 勝久* 藤本 忠博** 原美 オサマ** 千葉 則茂**
* 岩手大学大学院 工学研究科 ** 岩手大学 工学部

本論文では，物体形状を 3 次元点群として効率的に管理する LDI（Layered Depth Image）を利用した半透明物体の効率的な表示法について提案する．本手法では，α 値を持つ 3 次元点群で表現された半透明物体に対して，異なる基準視点による複数の LDI を用いた点群の分割登録を行うことにより，単一の LDI の場合に生じる 3 次元空間上の位置によるサンプリング密度の疎密の問題を改善する．また，複数の LDI に対してマクミランの順序付けアルゴリズムを導入することで，高品質な任意視点画像を効率的に生成する．具体的な 3 次元点群としてボクセルモデルを用い，複雑な形状を持つボクセルモデルに本手法を適用した実験により，その有効性が確認できた．

1．はじめに

物体の表面や内部を 3 次元点の集合（点群）として表現するポイントベース CG の研究が盛んに行われてきている[1]．その典型的なレンダリング法の一つとして，ある基準となる視点から物体を描画した元画像の各画素が持つ情報（その画素に描かれた物体上の点の色，ならびに，視点からのデプス値）から 3 次元点群を構築し，その点群を任意視点のスクリーンに向けて投影することで任意視点画像を生成する方法がある[2,3,4,5]．このとき，一般に，基準視点と任意視点における視線方向やスクリーン位置の違いなどにより，任意視点スクリーン上で基準視点からの 3 次元点が投影されない画素領域（隙間領域）が発生する．これを解決する方法の一つである LDI（Layered Depth Image）[4,5]では，図 1 に示すように，複数の視点から元画像を描画し，その複数の元画像の各画素の情報を一つの基準視点の画素情報に統合する．図 1 中の「○」および「●」は，それぞれ実カメラ 1, 2 の視点から画素に向けたレイでサンプリングされた点であり，各々のサンプリングされた点を LDI の基準視点から画素に向けたレイ上へ統合したものが「□」および「■」である．LDI の各画素には，その画素に向けて基準視点から発したレイと物体とが交差する複数の点の情報（色，デプス値）が視点からのデプス値の順にリスト化されて登録される．この LDI の画素情報から構築した 3 次元点群の投影により，任意視点画像上の隙間領域を埋めることが可能となる．

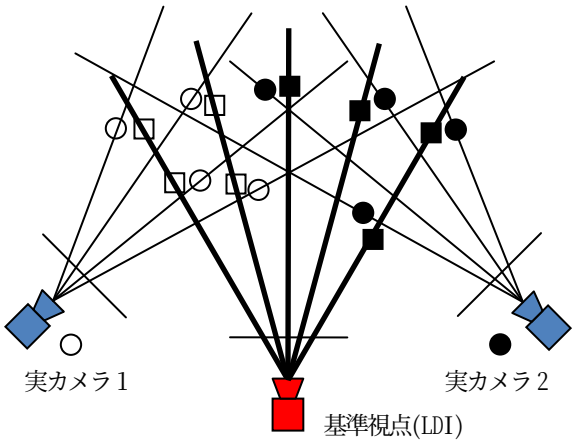

図 1：LDI における基準視点への画素情報の統合（再サンプリング）

一方，隙間領域とは逆に，複数の 3 次元点が任意視点スクリーン上の同一の画素に投影されうる．このとき，対象物体が不透明（α 値を持たない）な場合，任意視点画像の画素ごとに描画すべき（任意視点に最も近い）3 次元点は，Z バッファ等を用いれば容易に判定できる．しかし，半透明（α 値を持つ）の物体を描画する場合は，画素ごとに α 値を持つ 3 次元点群を正しい前後関係で投影する必要があるため，同一の画素に投影されるすべての 3 次元点に関して任意視点からのデプス値によるソートを行って，遠くから近くに向けて正しい投影順によって描画する必要がある．このソートには点の個数に依存した計算時間がかかり，特に，任意視点の移動に応じて高速に任意視点画像を生成したいような場合，このソートの計算時間が問題となる．これを解決する一手法として，LDI にマクミランの順序付けアルゴリ

ズム[6]を適用した高速な任意視点画像の描画法が提案されている[4,5].

実写画像からの 3 次元点のデプス値の推定により LDI を生成する場合[5]などがあることから，LDI の生成は平行投影に限定せず，より一般的な透視投影で実現される必要がある．しかし，LDI は，マクミランアルゴリズムとの組み合わせにより高速な描画が実現できる反面，図 1 に示すように，透視投影の場合，基準視点から各画素に向けたレイの放射状の広がりにより，3 次元空間上で基準視点から遠くになるにつれて，登録されている 3 次元点群の空間分布（サンプリング密度）が疎になり，正確な物体の表現ができなくなってくる傾向がある．結果として，任意視点が基準視点から遠ざかるにつれて，生成される任意視点画像の精度が落ちることになる．また，これに関連する LDI の欠点として，たとえ LDI に統合する前の複数の元画像の視点が広範囲に渡っていたとしても，LDI への統合の過程で，元画像の画素情報が LDI の各画素を通過するレイによって再サンプリングされることになるため，個々の視点位置に依存して元画像の各画素が持っている物体の正確な情報が十分に活かされない，ということが言える(図 1 参照).

そこで，本研究では，半透明物体をあらわすボリュームモデルを対象とし，上記の透視投影によるサンプリング密度の疎密に関連した問題を改善し，広範囲から見た物体の情報（色，デプス値，α値）をなるべく詳細に保持するよう，複数の基準視点に対して LDI を生成し，その複数の LDI に対してマクミランアルゴリズムを適用することで効率的に任意視点画像を生成する手法を提案する．ボリュームモデルの場合，LDI には，物体の表面だけでなく，その内部を構成する 3 次元点も登録されることになる．本手法は任意の半透明ボリュームモデルからサンプリングした点群に対して適用できるが，本論文では，半透明ボリュームモデルとしてα値を持つボクセルモデルを取り上げ，各ボクセルを 3 次元点として扱うこととして議論を進める．提案するアルゴリズム等は，そのまま任意の（不規則な配置の）半透明点群に対しても適用可能である.

2．マクミランアルゴリズム

ある基準視点から描画した元画像について，画素ごとに，その画素に描かれた物体表面上の（一つの）3 次元点の色，ならびに，視点からその 3 次元点までのデプス値が保持されているとする．この 3 次元点を任意視点のスクリーン上に投影することで任意視点画像を生成することを考える．このとき，マクミランの順序付けアルゴリズム[6]は，基準視点と任意視点，ならびに，それらのスクリーンの相対的な位置関係により，効率的に任意視点からの正しい前後関係を満たす 3 次元点の投影順序を決定する.

まず，基準視点 e_0 と任意視点 e_1 を直線で結び，次に，その直線と基準視点のスクリーン（基準スクリーン）との交点 p を求める．この交点 p をエピポーラ点と呼ぶ．そして，エピポーラ点 p の基準スクリーン上での位置と，基準視点 e_0 と任意視点 e_1 の前後関係で投影の順序を決定する．具体的には，それらによって投影方法の場合分けが生じ (図 2, 3 参照)，p の位置によって 9 通りの場合に分けられ，e_0 と e_1 の前後関係によって 2 通りの場合に分けられるので，全部で 18 通りのパターンが存在する．e_1 が e_0 よりも前方にある場合は，図 2 のように，p に近づくような投影順序になる.

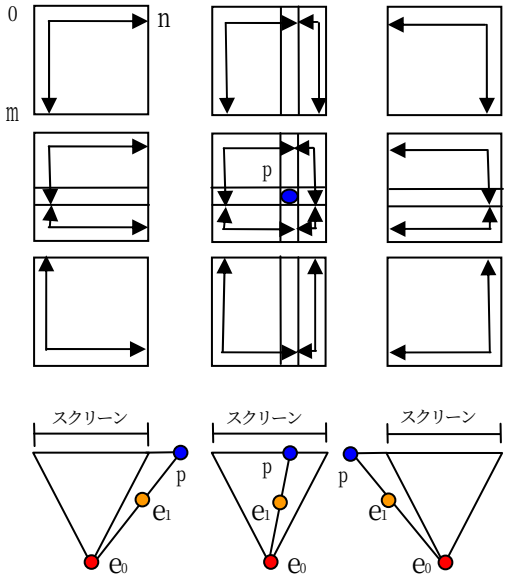

図 2：マクミランアルゴリズムの投影順序（e_1 が e_0 より前方にある場合）

例えば，図 2 の 9 通りのパターンのうちの中央のパターンは，p が基準スクリーン内に含まれる場合で

ある．この場合，p を通る水平ならびに垂直方向の 2 直線（p を含む 1 画素分の幅を持つ領域）により基準スクリーンが 4 つの部分領域に分割される．このとき，その 2 直線の領域内の画素は，それが隣接するいずれの部分領域に含めてもよい．そして，各部分領域について，その領域内の画素群を p に近づく順序（図 2 中の矢印の方向）で走査し，各画素が保持する 3 次元点を任意視点へ向けて投影していく．この投影順序は，常に，任意視点から見て遠くから近くに向かう順序で 3 次元点を投影する．なお，4 つの部分領域を処理する順序は任意でよい．図 2 の残りの 8 通りのパターンは，p が基準スクリーンから外れた場合である．これらの場合には，基準スクリーンを含む無限平面上で p の位置を考えることで，基準スクリーンは 1 つか 2 つの部分領域に分割される．そのいずれの場合でも，中央のパターンと同様，p に近づく順序（図 2 中の矢印の方向）で部分領域内の画素群を走査し，3 次元点の投影を行う．一方，上記とは逆に，e_1 が e_0 よりも後方にある場合は，図 3 のように，p から遠ざかるような投影順序になる．

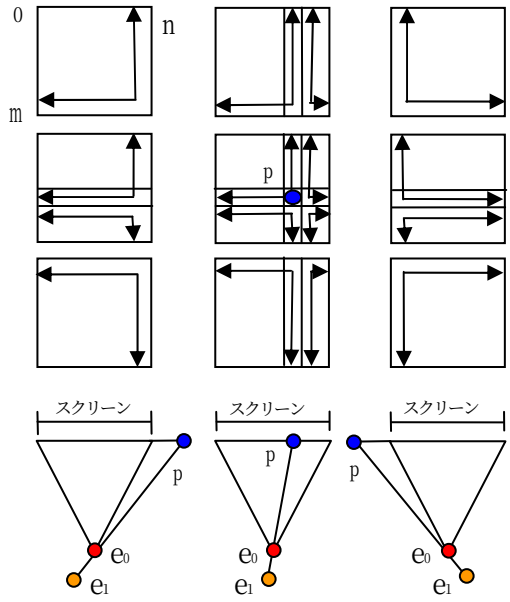

図 3：マクミランアルゴリズムの投影順序（e_1 が e_0 より後方にある場合）

以上のアルゴリズムにより，投影される 3 次元点を任意視点からのデプス値によってソートすることなく，任意視点から見て遠くから近くに向かう順序での効率的な投影が可能となる．

上記のアルゴリズムは，基準スクリーン上の各画素が二つ以上の 3 次元点を保持している場合，すなわち，LDI に対しても有効である[4,5]．LDI に適用する場合には，上述の順序で基準スクリーン上の各部分領域内の画素群を走査する過程で，各画素を訪れた際に，その画素に登録されている複数の 3 次元点を任意視点から見て遠くから近くに向かう順序で投影する．このとき，それらの 3 次元点は LDI 中でデプス値の順でリスト化されて登録されているため，あらためてデプスソートすることなく，そのリストを一方向に巡るだけでよい．

3．複数 LDI による半透明物体の効率的な表示

本研究では，半透明物体を表現したα値を持つボクセルモデルに対して，次の二つを実現することを目的とする．

1) ボクセルモデルの複数 LDI への分割登録
2) 複数 LDI にマクミランアルゴリズムを適用した効率的な任意視点画像の描画

以下，それぞれの手法について説明する．なお，以下で述べる問題点等の議論は，ボクセルモデルに限らず，不規則な分布を持つ一般の 3 次元点群に対しても当てはまり，提案する手法は任意の 3 次元点群に対して有効である．

3．1　ボクセルモデルの複数 LDI への分割登録

ボクセルモデルの LDI への登録では，各ボクセルを 3 次元点として扱い，LDI のスクリーンに向けた投影を行う．そして，個々のボクセルを，それが投影された画素に対して登録する．ボクセルモデルを単一の LDI に登録する場合には，1 節で述べた基準視点からのレイの放射状の広がりにより 3 次元空間上でのサンプリング密度に疎密が生じるという問題（図 1 参照）が大きく悪影響を及ぼす．すなわち，図 4 に示すように，基準視点から遠くなるほど一つの画素に対してより多くのボクセルを登録することになり，その画素の中心を通過するレイから遠いボクセルも登録せざるを得ない．このとき，仮に一つの画素に登録されたボクセル群が基準視点からのデプス値の順で正しくリスト化されたとしても，2 節で述べた方法で任意視点への投影を行う際，そのリ

ストを一方向に巡ってボクセルを投影することが任意視点から見て正しい前後関係での投影にはならない場合が生じる．例えば，図 5 に示すように，ボクセル 1〜5 が基準スクリーンに対してほぼ平行に並んでいる場合を考える．これらが一つの画素に登録される場合，ボクセル 2 と 4，ボクセル 1 と 5 のデプス値（基準視点からの距離）がそれぞれ非常に近いことから，リスト化される順序が一定方向（1→5 の順や 5→1 の順）ではなく，例えば，デプス値の昇順（小から大）に 3→2→4→1→5 というような順でリスト化されてしまう．しかし，実際には，いずれの任意視点から見ても，正しい前後関係のためには 1→5 あるいは 5→1 の一定方向の順で投影されなければならないため，２節で述べたリストを一方向に巡る方法では任意視点画像の描画結果に矛盾が生じる．さらに，基準視点から遠い領域では，3 次元空間をまばらに通過するレイに対して，各ボクセルをどのレイ（を発している画素）に登録するかを決定する方法も重要である．そして，3 次元空間上で近隣のボクセルどうしを隣接するどの画素に登録するかにより，マクミランアルゴリズムの投影順序では必ずしも任意視点からの正しい前後関係の順序で投影が行われない場合が生じうる．

よって，本研究では，単一の LDI ではなく，複数の LDI に対してボクセルモデルを分割登録する．すなわち,3 次元空間上に複数の LDI の基準視点を置き，各ボクセルを最も適した LDI に登録する．具体的には，個々のボクセルごとに，その中心点からの距離が最も近いレイを発している画素に登録する．ただし，3 次元空間上での距離計算のコストを軽減するため，下記のような近似を用いる．図 6 において，基準視点からボクセルの中心点までの距離（デプス値）を Dx とする．また，ボクセルの中心点を LDI のスクリーン上に投影し，その投影点を含む画素の画素中心までの距離を Lp，基準視点から投影点までのデプス値を Dp とする．このとき，その画素中心を通るレイからボクセルの中心点までのスクリーンに平行な方向における距離 Lx は次のように求められる．

$$Dp:Dx = Lp:Lx$$

$$Lx = (Dx*Lp)/Dp$$

ここで，基準視点からスクリーン中心までの（画素によらない）距離 D_0 で Dp を近似することで次式を得る．

$$Lx \quad (Dx*Lp)/D_0 \propto Dx*Lp$$

この距離 Lx を各 LDI について求める．そして，すべての LDI のうちで，その距離が最小となる LDI の画素に対して，そのボクセルの登録を行う．そして，すべてのボクセルの登録を終えた後，LDI ごとに，共通の画素に登録されたボクセルどうしで，基準視点からのデプス値によるソートを行い，デプス値の順でリスト化する．この複数の LDI への分割登録により，各ボクセルがなるべく自分の近くを通るレイを発している LDI の画素に登録されるため，前述の問題が軽減される．

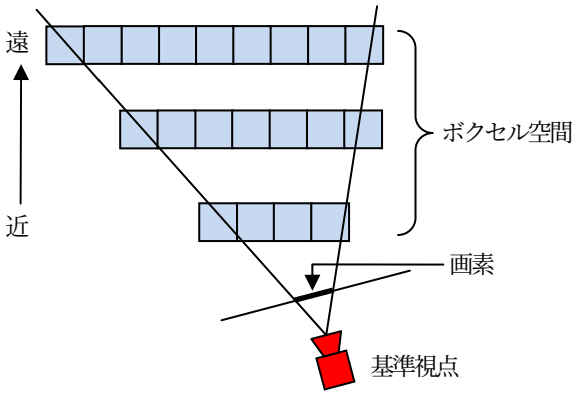

図 4：一つの画素に登録されるボクセル

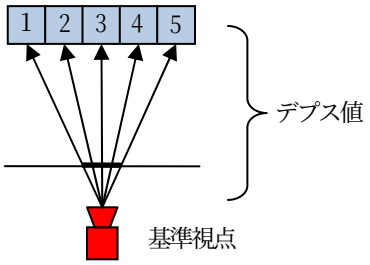

図 5：一つの画素に登録される近隣のボクセルに対するデプス値によるリスト化

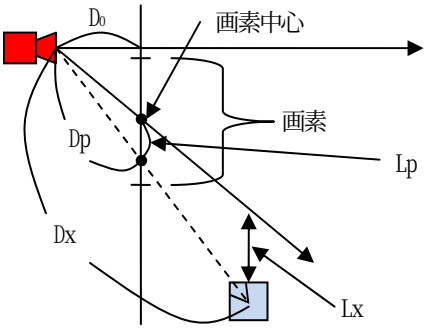

図 6：ボクセルとレイの距離計算

3．2　複数LDIによる任意視点画像の描画

単一の LDI の場合には，2節で述べたマクミランアルゴリズムの適用により容易に任意視点画像生成ができる．しかし，複数の LDI を用いる場合，個々の LDI に対して2節の方法を適用しただけでは，異なる LDI が持つ3次元点の間で任意視点から見た正しい前後関係を判定できない. そこで, 本研究では, 任意視点のスクリーンに LDI ごとの投影バッファを用意する(図 7 参照)．この投影バッファは任意視点画像と同じ解像度を持ち，各画素には，その画素位置に LDI から投影される（複数の）3次元点の ID を登録していく．この投影バッファを用いて，下記のアルゴリズムにより任意視点画像を描画する．

1) 各 LDI にマクミランアルゴリズムを適用し，投影バッファに3次元点群を投影する．このとき，投影バッファの同一の画素に投影される 3 次元点は，マクミランアルゴリズムの働きにより任意視点から見て遠くから近くに向かう順序（任意視点からのデプス値の降順（大から小））で投影されるため，そのままの順序でバッファに登録していく．図 7 の①，②，③，…は，それぞれ，異なる LDI の投影バッファについて，任意視点画像の同じ画素に対応する部分を示す．
2) すべての LDI の全3次元点について 1)の投影が終了した後，任意視点画像の画素ごとに，各投影バッファに登録された 3 次元点群について，すべての投影バッファのうちで（任意視点からの）デプス値が大きなものから順に取り出していき，任意視点スクリーンへの描画を行う．具体的には，n 個の LDI に対して，描画する 3 次元点がなくなるまで，以下の処理を行う．

2-1) 各投影バッファの先頭の登録3次元点を取り出し，その n 個の点をデプス値に関して降順（大から小）にソートし，n 個の要素を持つ描画リストに入れる．図7では，投影バッファ①，②，③，…の先頭 A，B，C，…を取り出してソートを行うことになる．
2-2) 描画リストの先頭要素の点（デプス値が最大）を取り出し，任意視点スクリーンに描画する．
2-3) 2-2)で描画した点が属していた投影バッファから次の登録3次元点を取り出し，描画リスト中に残っている n-1 個の点とデプス値を比較し，降順となる位置に挿入する．
2-4) 2-2)に戻る．

上記のアルゴリズムにより，任意視点画像の画素ごとに，すべての 3 次元点が正しい前後関係の順序で描画される．このアルゴリズム中で，最も大きな計算量を占める部分は2-3)であるが，描画リストに残っている n-1 個の点は常にデプス値に関してソート済みであるため，この比較の計算量のオーダーは O(n)である．n 個の LDI に登録された 3 次元点の総個数を m とし，n << m とすると，全3次元点を描画するための計算量のオーダーは O(n m)となる．

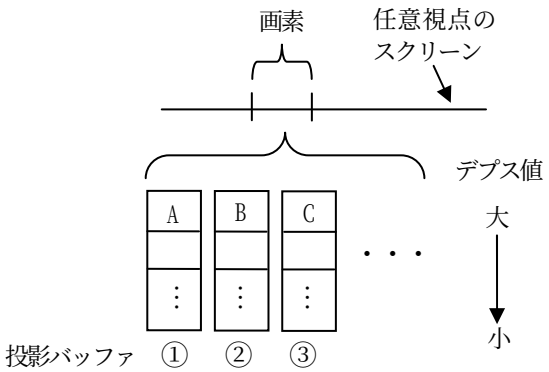

図7：投影バッファ

4．実験

4．1　実験概要

本手法の有効性を検証するため，半透明の樹木のボクセルモデルを用いて任意視点画像生成を行った．樹木のような詳細で複雑な形状を持つ物体は，ポリゴンモデルとして表現した場合，レンダリング時のエリアシングの問題，すなわち，スクリーン上に投影されたポリゴンの詳細度が 1 画素以下となることでジャギーが発生し，視点の移動時にフレーム間でのフリッカ（ちらつき）を引き起こす場合がある．このため，特に視点から遠方に物体がある場合などは，ポリゴンモデルの代わりに，視点からの距離に応じたボクセル解像度を持つボクセルモデルを用いることでエリアシングを軽減するという方法が良く用いられる．本実験で使用する樹木のボクセルモデ

ルも，元になるポリゴンモデルから生成した．この生成方法は，まず，目的とする解像度のボクセル空間を設定し，その中に元となる樹木のポリゴンモデルを配置する（図 8 参照）．このボクセル空間内において，ボクセルを順番に走査していき，ポリゴンが存在する座標でのボクセルを残していく．最終的に残ったボクセルで形成されたモデルが目的のボクセルモデルとなる．本実験では，これを効率的に行うため，グラフィックライブラリ OpenGL を利用した．具体的には，図 9 中の青色部分に示すような幅が 1 ボクセル分の 3 次元スライス領域を考え，この領域をビューボリュームとした図 9 中の走査方向への平行投影により，その領域内のボクセルの 2 次元解像度と同じスクリーン解像度でポリゴンを描画し，スクリーンの各画素に描かれた色情報（r, g, b）とα値を対応するボクセルに与える．そして，ポリゴンが描かれない画素に対応するボクセルを削除する．スライス領域を図 9 中の走査方向に 1 ボクセルずつずらしながら上記の処理を行うことで，最終的なボクセルモデルに残すボクセルを求めていく．

実験では，生成したボクセルモデルに対して，複数の基準視点（LDI）を設定し，３．１節の方法で各ボクセルを適切な基準視点の LDI へ分割登録した．そして，生成された複数 LDI に対して，３．２節の方法を適用し，任意視点のスクリーンへスプラッティング[7]により描画を行った．

以降，実験によって得られた結果を示す．

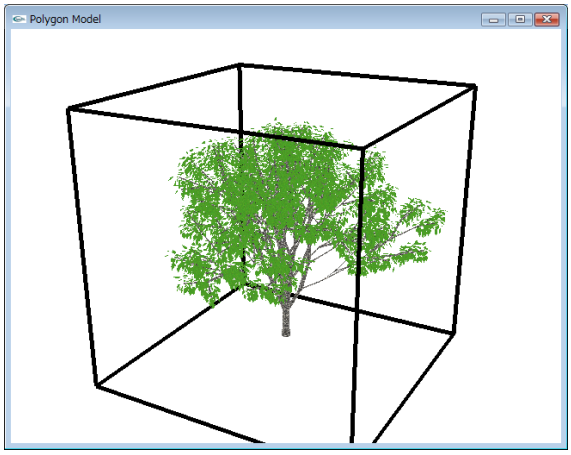

図 8：元となるポリゴンモデルとボクセル空間

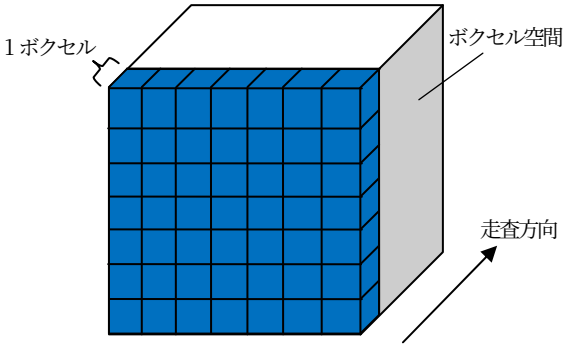

図 9：ボクセル空間の走査

４．２　ボクセルモデルの生成

４．１節で説明した手法で生成された樹木のボクセルモデルを図 10 に示す．ボクセル空間の解像度は 200×200×200 であり，生成されたボクセルモデル中のボクセル数は 46,311 個である．このボクセルモデルを使用して任意視点画像生成を行った．

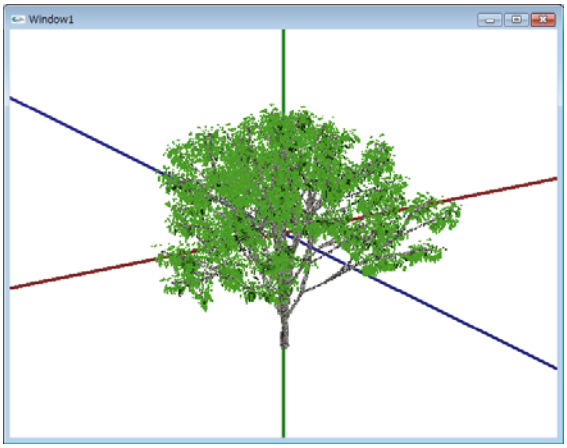

図 10：生成されたボクセルモデル（ボクセル数：46,311）

４．３　ボクセルモデルの複数 LDI への分割登録

図 10 のボクセルモデルを 3 個の LDI へ分割登録した結果を図 11 に示す．３．１節で述べた方法により，個々のボクセルを各 LDI の基準視点のスクリーンへ投影し，その投影点の画素中心からの距離と基準視点に対するデプス値によって登録すべき最適な LDI を決定した．図 11 中の赤，青，緑の四角形が各基準視点であり，モデル中の個々のボクセルの表示色は，登録された LDI の基準視点の色と対応している．３．１節の分割登録の方法により，各ボクセルが自分に近い基準視点の LDI に登録される傾向が強いことが確認できる．各 LDI に対して，その基準視点に近く，

さらに，画素中心に近いボクセルを登録することによって，基準視点から遠いサンプリングが疎になる空間領域に位置するボクセルを登録する機会が減り，結果として，生成される任意視点画像の精度が向上することが期待される．

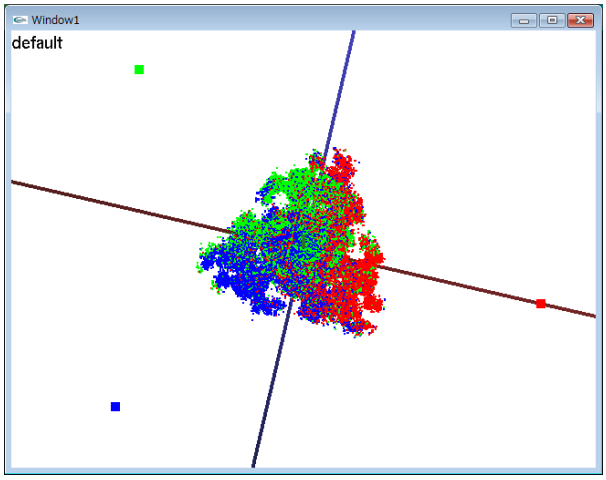

図 11：ボクセルの分割登録（モデルを上から見た図）

4．4　任意視点画像の生成

3．1 節で述べたように，ボクセルモデルを 1 個の LDI に登録する際には，基準視点から遠い空間領域上のボクセル群については，一つの画素に対してより多くのボクセルが登録されてしまうことによる問題が生じうる．そこで，この問題を改善するために複数の LDI を用いる提案手法の有効性を検証するため，図 10 のボクセルモデルに対して，LDI の個数を 1 個，および，6 個としたそれぞれの場合について，任意視点画像の生成を行い，生成画像の品質を比較した．また，それらとの比較のため，LDI とマクミランアルゴリズムを用いず，全ボクセルに対して任意視点からのデプス値によるデプスソートを行うことで任意視点からの前後関係を判定することによる正確な任意視点画像の生成も行った．これらの 3 つの場合（デプスソート，1 個の LDI，6 個の LDI）のそれぞれについて，はじめに，図 10 のボクセルモデル全体に対する任意視点画像の生成を行った．続いて，より詳細な比較を行うため，図 10 のボクセルモデルを用いて図 12 中の四角で示した部分を奥行き方向に切り取った空間領域内のボクセルだけを残した部分モデル（ボクセル数：8,065）に対する任意視点画像の生成を行った．図 13 は，この部分モデルを斜め横から見た図（矢印①が図 12 の視点の方向）である．さらに，ボクセルを 3 次元点として任意視点スクリーンへ投影する際の前後関係の正しさを詳細に観察するため，図 14，15 に示すように，図 12，13 の部分モデルを赤と黒の 2 色に色付けし，任意視点画像を生成した．なお，LDI を 1 個とした場合の LDI の基準視点は，図 13 の矢印①の方向から部分モデルを見る位置に配置した．また，LDI を 6 個とした場合の基準視点は，図 16 のようにボクセルモデルを取り囲むように配置した．そして，実験結果として図 17～25 に掲載した任意視点画像は，図 13 の矢印②の方向からモデルを見るように任意視点を配置して生成した．

上記のそれぞれの条件で生成した任意視点画像（静止画）の図番号，ならびに，任意視点を動かしたアニメーション動画のファイル名（付属のコンテンツ動画ファイル）を表 1 にまとめる．

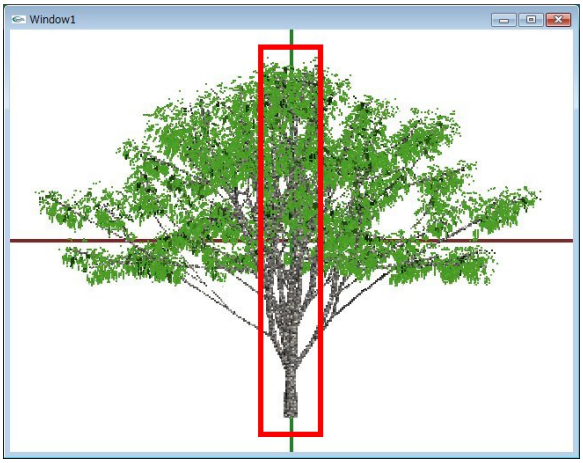

図 12：部分モデルの範囲

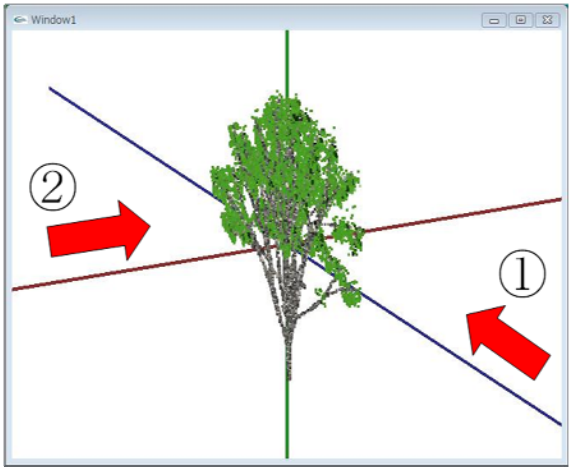

図 13：図 12 の四角部分だけを斜め横から見た図

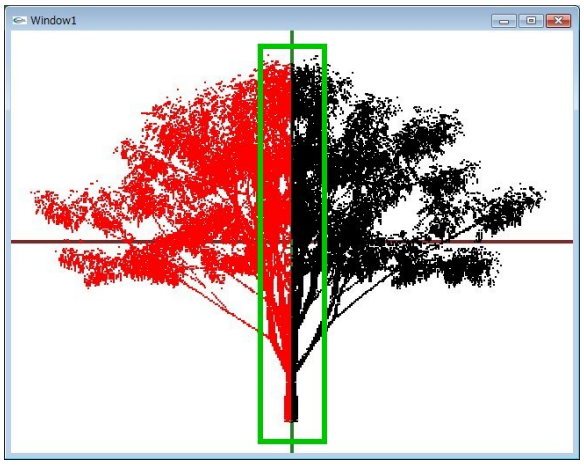

図 14：2 色に色付けしたボクセルモデル

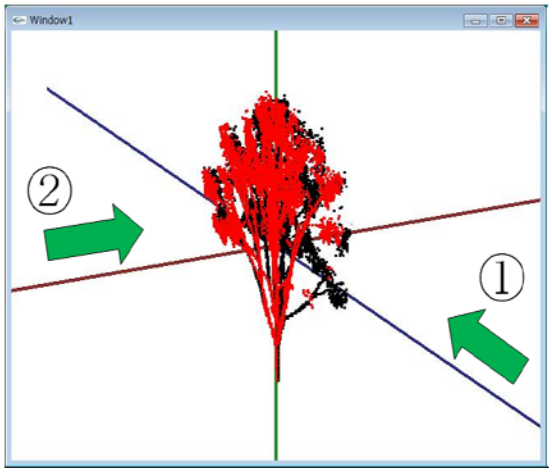

図 15：図 14 の四角部分だけを斜め横から見た図

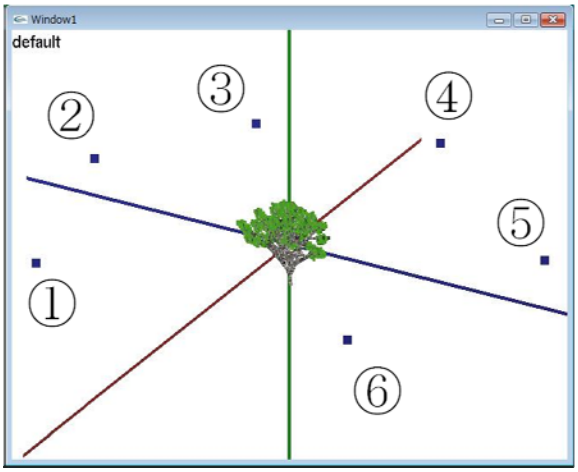

図 16：6 個の基準視点の位置関係

表 1：条件ごとの実験結果の図番号と動画ファイル名
（LDI が 1 個と 6 個の場合については，各欄内の上段が図番号，下段が動画ファイル名）

	ボクセルモデル全体	部分モデル（カラー版）	部分モデル（赤黒版）
デプスソート	図 17	図 20	図 23
LDI：1 個	図 18 LDI_1_whole.avi	図 21 LDI_1_part_COL.avi	図 24 LDI_1_part_RB.avi
LDI：6 個	図 19 LDI_6_whole.avi	図 22 LDI_6_part_COL.avi	図 25 LDI_6_part_RB.avi

4．4．1　ボクセルモデル全体に対する任意視点画像生成

ボクセルモデル全体に対して生成された任意視点画像として，図 17 は任意視点からのデプス値によるデプスソートで生成されたもの，図 18 は 1 個の LDI から生成されたもの，図 19 は 6 個の LDI から生成されたものである．また，図 18 と図 19 の条件のもとで任意視点を動かしたアニメーション動画も生成した．図 17,18,19 の静止画からでは差異を確認するのは難しいが，アニメーション動画による比較では若干の差異が確認できた．図 18 の条件でのアニメーションでは，視点移動に伴って樹木内部のちらつきが確認できた．これに対して，図 19 のアニメーションでは図 18 で確認されたちらつきが軽減されている．これにより，複数 LDI によるボクセルの分割登録の効果が確認されたと言える．

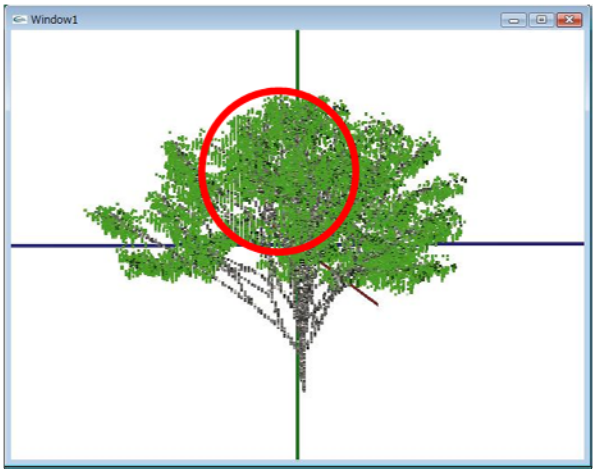

図 17：ボクセルモデル全体に対してデプスソートにより生成された任意視点画像

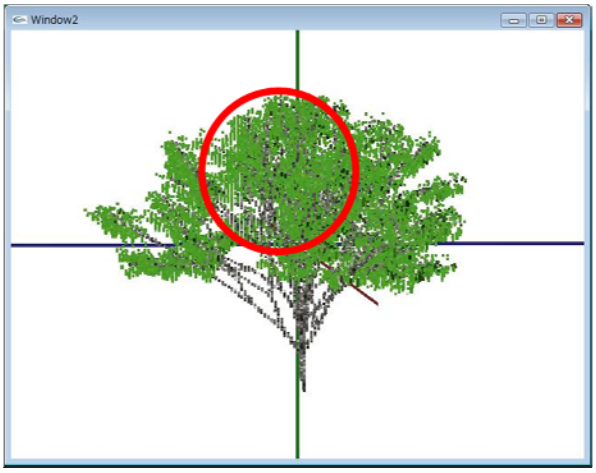

図 18：ボクセルモデル全体に対して 1 個の LDI から生成された任意視点画像

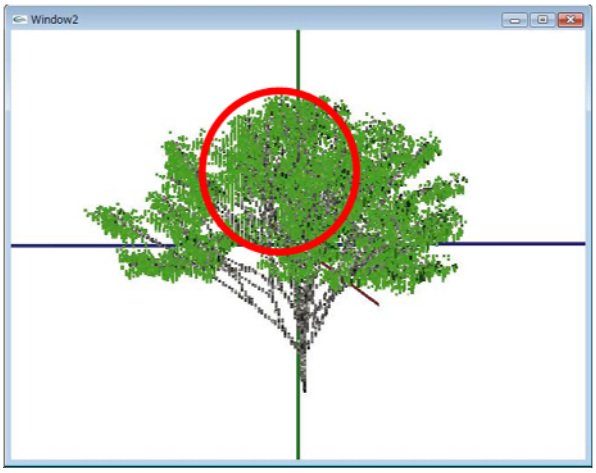

図 19：ボクセルモデル全体に対して 6 個の LDI から生成された任意視点画像

4．4．2　部分モデルに対する任意視点画像生成

部分モデルに対して生成された任意視点画像として，図 20 は任意視点からのデプスソートで生成されたもの，図 21 は 1 個の LDI から生成されたもの，図 22 は 6 個の LDI から生成されたものである．図 20 と図 21 の赤丸内を比べると，図 20 の正確な任意視点画像に比べ，図 21 では本来は葉の緑色のボクセルが描画されるべき部分に後方にある枝の灰色のボクセルが描画されてしまっている．一方，図 22 では，図 21 に比べて，正しく描画される緑色のボクセルが増している．

しかし，樹木のように複雑で細かいモデルの場合，枝葉の重なりにおいて，ボクセルの前後関係が多少異なっていても，図 20，21，22 からでは差が分かりにくい．そこで，部分モデルを赤と黒の 2 色に色付けして同じ条件で任意視点画像を生成したものが図 23，24，25 である．図 23 の正確な任意視点画像に比べ，1 個の LDI だけから生成された図 24 では，特に緑丸内の部分のように，本来は前方の赤色のボクセルが描画されるべき部分に後方の（本来は隠れるはずの）黒色のボクセルが描画されてしまっている．これは，3．1 節で述べた問題，具体的には，一つの画素に複数のボクセルが登録される際に，図 15 の矢印①の LDI の基準視点の方向に対して，矢印②の任意視点の方向に並ぶ互いに近い（基準視点からのデプス値が近い）ボクセル群に図 5 のような状況が発生し，LDI の一つの画素上で基準視点からのデプス値によりリスト化されたボクセルの順序が矢印②の方向での正しい前後関係と一致しなかったことが原因の一つであると考えられる．一方，LDI を 6 個とした図 25 では，ボクセルの分割登録により，互いに近い複数のボクセルが一つの画素に登録されることが軽減されるので，図 24 に比べて，後方の黒色のボクセルが描画されることが減少し，任意視点画像の精度が向上している．これにより，複数 LDI によるボクセルの分割登録の効果が確認されたと言える．

また，図 21 と図 22 の条件のもとで任意視点を動かして生成したアニメーション動画から，さらにはっきりと複数 LDI による効果が確認できる． 図 21 の条件のアニメーションでは，視点移動に伴って，葉の緑色のボクセルと枝の灰色のボクセルが前後に急激に入れ替わる部分が多数あり，不自然に感じるアニメーションとなっている．しかし，図 22 の条件のアニメーションでは，ボクセルどうしの急激な前後の入れ替わりが軽減し，なめらかなアニメーションとなっている．同様に，図 24 と図 25 の条件でのアニメーションでは，単純な色分けのモデルにしているため，先ほど説明した図 21 と図 22 のアニメーションでの差異がよりはっきり分かるようになっている．

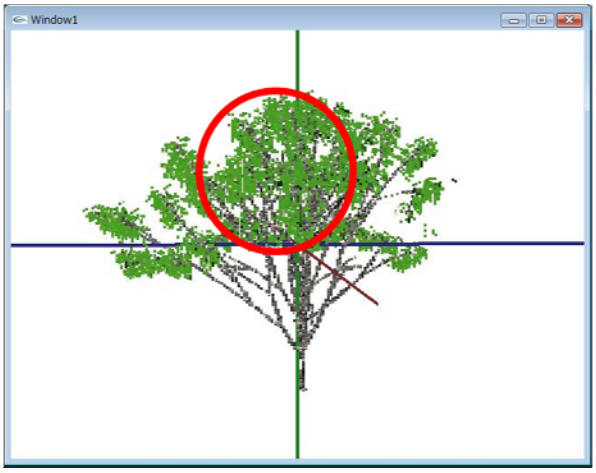

図 20 : 部分モデルに対してデプスソートにより生成された任意視点画像（カラー版）

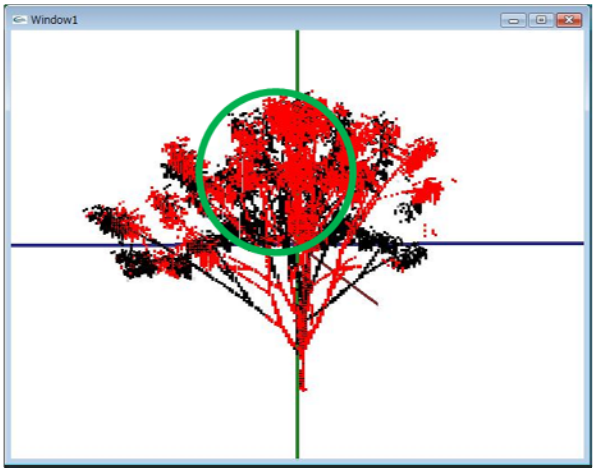

図 23 : 部分モデルに対してデプスソートにより生成された任意視点画像（赤黒版）

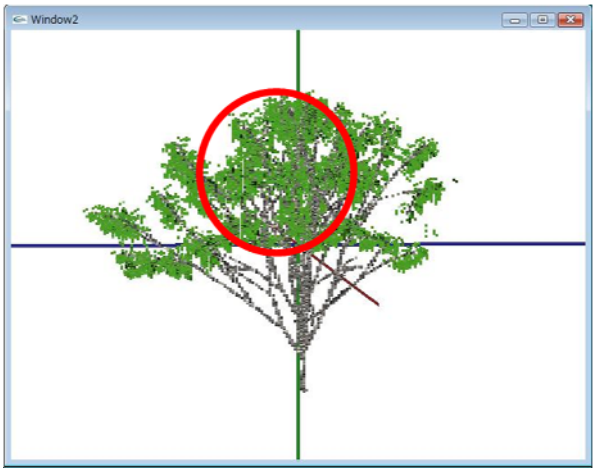

図 21 : 部分モデルに対して 1 個の LDI から生成された任意視点画像（カラー版）

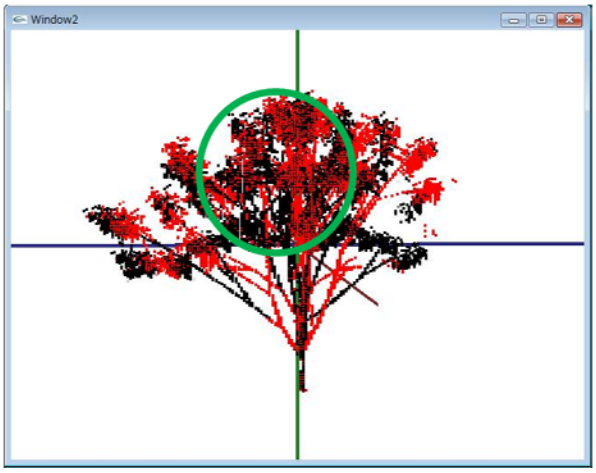

図 24 : 部分モデルに対して 1 個の LDI から生成された任意視点画像（赤黒版）

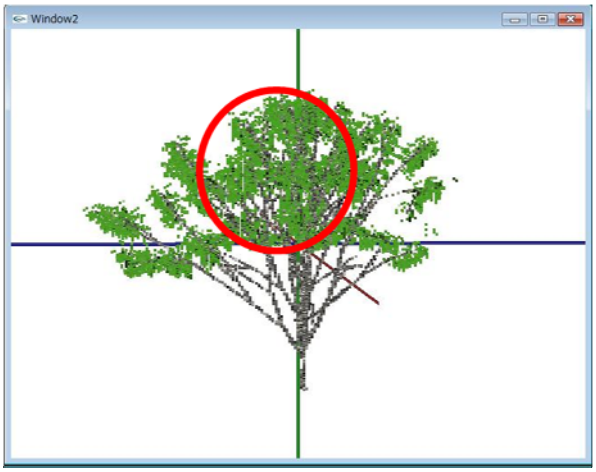

図 22 : 部分モデルに対して 6 個の LDI から生成された任意視点画像（カラー版）

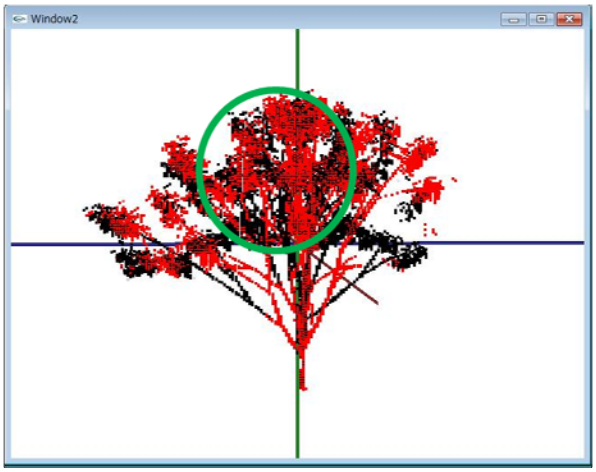

図 25 : 部分モデルに対して 6 個の LDI から生成された任意視点画像（赤黒版）

４．５　実行時間

本手法の実行時の処理を以下に示す．

1. 基準視点による LDI の生成：目的とする個数の基準視点を 3 次元空間上に設定し，３．１節の方法により，それぞれの LDI を生成する．
2. 任意視点画像の生成：任意視点を移動するごとに，以下の処理を行う．

2-1. 任意視点の位置により，各 LDI の基準スクリーン上のエピポーラ点を求める．

2-2. マクミランアルゴリズムを適用して各 LDI 中の登録 3 次元点(ボクセル)を任意視点に投影し，LDI ごとに投影バッファに一時保持する．

2-3. ３．２節のアルゴリズムにより，すべての投影バッファの登録 3 次元点を任意視点からのデプス値の大きいものから順に描画する．

任意視点が動くたびに 2-1～2-3 を繰り返すことで動的に任意視点画像が生成できる．ここで，1 は最初に一回だけ行えばよいので前処理とし，2-1～2-3 を描画処理とすると，図 10 のボクセルモデルの描画について，計測した実行時間は表 2 となった．なお，描画処理の実行時間は，任意視点をランダムに移動させて複数フレームの描画行い，1 フレームあたりの平均を取った．LDI の数が増加するにつれて，描画処理速度が遅くなるが，ボクセル数が 46,311 個のモデルに対して，LDI が 7 個の場合でも，約 17 FPS 程度の描画速度が実現できている．ここで，LDI とマクミランアルゴリズムを用いずに任意視点の移動のたびに 3 次元点（ボクセル）をデプスソートして描画する場合の 1 フレームあたりの描画処理の実行時間は，ほぼ，表 2 中の LDI が 1 個の場合の LDI 生成に要する時間に相当する(LDI の生成においては，画素ごとの登録 3 次元点のデプスソートに大部分の実行時間が費やされるため)．よって，LDI を複数個にした場合でも，マクミランアルゴリズムにより描画の高速化が実現できていることがわかる．なお，本実験では，デプスソートにクイックソートを用いた．また，LDI の個数に対する「LDI 生成」と「描画処理」の実行時間の関係は，それぞれ図 26 と図 27 のグラフのようになった．配列の初期化などのフレーム描画ごとの諸々の前処理，また，任意視点スクリーン上での LDI ごとの 3 次元点の投影位置の分布のバランスなどにより，３．２節で述べた O(n m) のオーダーには完全には一致していないが，一定の 3 次元点の個数 m (= 46,311)のもとで，LDI の個数 n に対して，おおよそ O(n)のオーダーの傾向になっていることが確認できた．

表 2：実行時間

LDI の数	LDI 生成（秒）	描画処理（秒/フレーム）
1	0.753	0.0099
3	2.029	0.0368
5	3.841	0.0438
7	5.536	0.0573

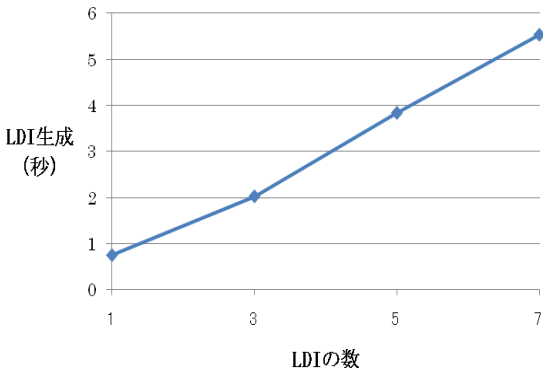

図 26：実行時間（LDI 生成）

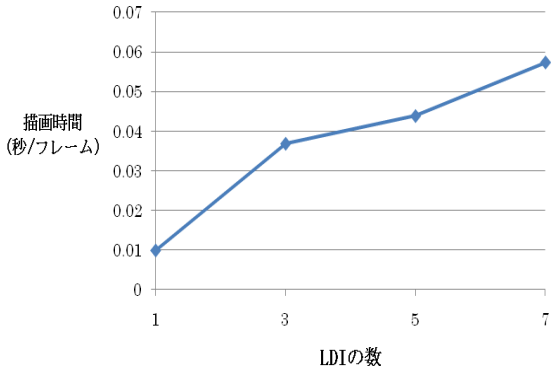

図 27：実行時間（描画処理）

また，実行環境を以下に示す．

OS　：Windows Vista Business

CPU ： Intel Core2 Quad CPU Q9550 @ 2.83GHz

GPU ： GeForce 9600 GT

プログラム開発環境 ： Microsoft Visual Studio 2005 Professional Edition

5．おわりに

本論文で提案した手法により，ボクセルモデルを単一の LDI ではなく，複数の LDI に分割登録することで，任意視点画像の精度を向上させることができた．また，各 LDI から任意視点のスクリーンへ 3 次元点（ボクセル）の投影を行う際に，LDI ごとにマクミランの順序付けアルゴリズムを適用することで任意視点に対して遠くから近くに向かう順序で 3 次元点を格納した投影バッファを利用し，高速に任意視点画像を生成することができた．

本論文で提案した手法は，ボクセルモデルに限らず，任意の 3 次元点群に対して有効である．そこで，今後，さまざまな点群モデルに対して本手法を適用し，有効性を検証する予定である．また，互いに近い 3 次元点どうしを複数の LDI により管理する方法を改善し，任意視点に投影した際の前後関係の判定がより正確に行える方法を検討していく予定である．さらに，複数の LDI に対して，より高速な任意視点画像の生成方法を提案することも今後の課題である．

参考文献

[1] 藤本忠博，今野晃市，千葉則茂，“ポイントグラフィックス概説”，芸術科学会論文誌，Vol.3，No.1，pp.8-21，2004.

[2] Shenchang Eric Chen，Lance Williams，“View Interpolation for Image Synthesis”，Proceedings of SIGGRAPH 1993，pp.279-288，1993.

[3] Nelson Max，“Hierarchical Rendering of Trees from Precomputed Multi-Layer Z-Buffers”，Eurographics Rendering Workshop 1996，pp.165–174，1996.

[4] Steven J. Gortler，Li-wei He and Michael F. Cohen，“Rendering Layered Depth Images”，Technical Report MSTR-TR-97-09，Microsoft Research，1997.

[5] Jonathan Shade，Steven J. Gortler，Li-wei He and Richard Szeliski，“Layered Depth Images”，Proceedings of SIGGRAPH 1998，pp.231-242，1998.

[6] Leonard McMillan，“Computing Visibility Without Depth”，Technical Report 95-047，Department of Computer Science，University of North Carolina，1995.

[7] Lee Westover，“Footprint Evaluation for Volume Rendering”，Proceedings of SIGGRAPH 1990，pp.367–376，1990.

[8] 吉田勝久，藤本忠博，原美オサマ，千葉則茂，“複数 LDI を利用した半透明点群の効率的な表示法”，第 25 回 NICOGRAPH 論文コンテスト，2009 年 10 月.