Vol.2022-GN-115 No.3
Vol.2022-CDS-33 No.3
Vol.2022-DCC-30 No.3
2022/1/20

三次元点群の高速・高品質な可視化のための密度を考慮したオクルージョン推定

森島 正博[1]　小川 剛史[2]

概要：三次元点群は，バーチャル観光やテレプレゼンスなど，現実空間の視覚的な再現を目指すアプリケーションでよく利用される．このようなアプリケーションでは，レンダリングの品質の観点から，点群をもとにメッシュによる 3D モデルを作成することが一般的であるが，高負荷な事前計算や手作業での調節が必要な場合がある．点群を直接用いた高速・高品質なレンダリングを行うため，画像空間でオクルージョンが発生している点を推定する方法が提案されているが，表示する点群の密度に応じたパラメータ調整が必要である．そこで本稿では，画像空間における局所的な点群の密度を考慮し，密度に応じた調整に依存しない推定手法について検討した．

A Study on Occlusion Estimation for Fast and High-Quality Rendering of 3D Point Clouds Considering Density

Masahiro Morishima[1]　Takefumi Ogawa[2]

Abstract: 3D point clouds are often used in applications that aim to visually reproduce the real world, such as virtual tourism and telepresence. Many applications generate meshes from point clouds to render high-quality surfaces, but these approaches require costly computation in preprocessing or manual parameter tuning. To achieve fast and high-quality rendering by directly using point clouds, methods have been proposed to estimate the occluded points in screen space. However, it is necessary to adjust the parameter according to the density of the points. In this paper, we studied an estimation method that does not depend on adjustment according to the density by considering the local density of the points in screen space.

1. はじめに

点群は，物体表面を点の集合として表現したデータであり，フォトグラメトリに関する技術や 3D レーザースキャン技術の発達により容易に取得でき，実在する建物などを三次元的にキャプチャすることで物理的な移動を伴わない鑑賞を可能とするバーチャル観光 [1] や，三次元映像を用いて臨場感のある遠隔コミュニケーションを可能とするテレプレゼンスに関するアプリケーション [2] など，現実空間の視覚的な再現を目指すアプリケーションに利用されている．点群のように三次元的な形状の情報を持つデータを用いることで，6 自由度の視点移動が可能となり，特に近年ではバーチャルリアリティ（VR）の技術も取り入れられ，盛んに研究が行われている [3][4].

これらの応用では，レンダリング品質などの観点から，メッシュ表現による 3D モデルが一般的に利用される．しかし，点群からメッシュを作成することは容易ではなく，細かい部分まで表現するために多数のメッシュが必要となり，その生成に高負荷な計算が必要となる問題や，手動でのパラメータ調整が必要となる問題が生じる [1][5][6].

そこで高負荷な計算や作業を省略するため，直接，点群をレンダリングに用いるアプローチが存在する．このアプローチにおける課題の一つはレンダリング品質であり，本来は物体によって遮蔽されて見えないはずの部分が点と点の間から透けて見え，適切にオクルージョンが反映されない問題がある．このような問題を解決するため，画像空間

[1] 東京大学大学院工学系研究科
Graduate School of Engineering, The University of Tokyo
[2] 東京大学情報基盤センター
Information Technology Center, The University of Tokyo

における処理に基づいてオクルージョンが発生している点を推定し除去することで品質を改善し，さらに VR などのアプリケーションに向けて，高速な動作が可能な手法が提案されている [5]．先行研究では，点の密度が均一であることが前提とされており，点群の密度によってオクルージョン推定に用いるパラメータを調整する必要がある．よって，表示する点群を切り替えた際にパラメータの再調整が必要となる問題や，点群の密度が不均一である場合に，パラメータ調整が困難になり品質が低下するという問題や，視点変更時に再調整が必要となるといった問題が生じると考えらえれる．

著者らの研究グループでは，画像処理に基づく手法 [5] に着目し，深度マップの勾配を用いることで，物体の境界付近でのオクルージョン推定の誤りを改善する手法について検討した [7]．本稿では，画像空間における点群の局所的な密度を考慮することで，密度に応じたパラメータ調整に依存しないオクルージョン推定アルゴリズムの検討を行った．

2. 関連研究

2.1 点群を直接使用するアプリケーション

メッシュによる 3D モデルではなく，点群を直接用いて VR 空間における鑑賞を可能とするシステムとして，Bonatto らの研究 [6] が挙げられる．メッシュを用いない理由として，点群に含まれるノイズがメッシュ作成に大きく影響することや，フォトリアリスティックな結果を得るためには大量のメッシュを生成する必要があることが挙げられている．また，Virtanen らの研究 [8] では VR 空間での点群の移動や，点群への着色など，インタラクティブなアプリケーションが提案されており，点群を直接使用することで，アプリケーションの作成が単純になることが述べられている．

Mekuria らは，点群を用いた三次元映像によって，リアルタイムの遠隔コミュニケーションが可能となるシステムを提案している [9]．リアルタイムという観点からは，メッシュの作成を行わず，点群を直接使用する方が適していると述べられている．

本稿では，このように点群を直接用いるインタラクティブなアプリケーションにおいて，レンダリング品質を改善することを目的としている．

2.2 点群のレンダリング手法

点群の高品質なレンダリング手法として，Splatting[10] が挙げられるが，法線などの情報が必要であり，事前に高負荷で時間のかかる計算が必要となる場合がある [5][8]．

事前の計算を必要としない手法として，それぞれの点を，正方形や円として大きく表示する手法がある．シンプルな手法ではあるが，点と点の間が補間され，オクルージョンもある程度再現することができる．Schütz らは，正方形や円ではなく放物面を用いることによって，より高品質な結果を得る手法を提案している [11]．一方で，これらの手法では滑らかな面を得ることが難しいなど，品質に関する課題は存在する [5]．

高品質かつ事前計算を必要としない手法として，画像処理に基づく手法 [5][12] が挙げられる．これらの手法の流れの一部を図 1 に示す．まず，画面に投影された点群の深度マップを用いて，オクルージョン推定を行う．この時，判定対象の点と，画像空間における近傍領域内の点が推定に用いられる．オクルージョン推定により，遮蔽されていると判定された点は除去される．次に，除去されずに残った点を膨張させて，点が存在しない画素を補間する処理が行われる．Bouchiba らの研究 [5] では，Pinuts らの研究 [12] の計算量と補間処理の改善などが行われている．

ここで，オクルージョン推定に用いる近傍領域のサイズは，近傍の点が含まれるために十分な大きさである必要がある一方で，大きすぎても推定誤りが起こりやすくなるため，適切に定める必要がある．文献 [5] では，画素 i の平滑化済みの深度値 $z[i]$ を用いて，視点の近くの物体ほど大きく，遠くの物体ほど小さく表示されるのに合わせて，近傍領域の一辺の長さ $L[i]$ を式 (1) のように深度値に反比例するように定められている．

$$L[i] = \frac{p}{z[i]} \tag{1}$$

ただし，p は点群の密度，画面の解像度，視野角を含む調整が必要なパラメータである．高速化のため，近傍領域のサイズは，level 0 (3×3)，level 1 (6×6)， level 2 (12×12) のように一辺の長さが 2 のべき乗に比例するように段階的に決定する必要があり，最終的な近傍領域サイズのレベル $l[i]$ は，$l[i] = \log_2 L[i]$ で決定される．

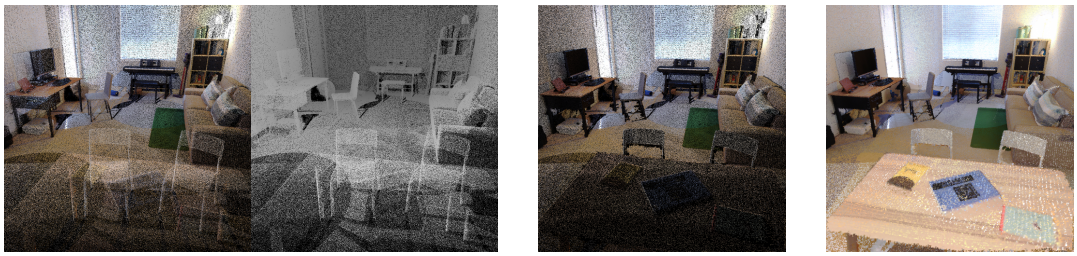
3. 密度を考慮した近傍領域サイズの決定

文献 [5] では，深度値から適切な近傍領域サイズを計算するために，点群の密度に応じてパラメータ p を調整する必要がある．そこで本稿では，深度値ではなく点群の密度を用いて適切な近傍領域サイズを計算することを提案する．具体的には，画像空間上での局所的な密度を計算し，近傍領域サイズを定める手法を検討する．本手法により，オクルージョン推定の際に点群の密度に応じたパラメータ調整の必要がなくなる事，また，場所によって密度が異なる場合にそれぞれの場所で適切な近傍領域サイズが計算できるため，より高品質な結果が得られる事が期待される．

3.1 画像空間での点群の密度の算出

まず，点群を投影した深度マップを用いて点の密度を計算する．本稿では，画像全体をグリッド状に分割し，各グリッド内に存在する点を数えることで局所的な点の密度を算出した．この時のグリッドの大きさは，16 × 16 ピクセ

Vol.2022-GN-115 No.3
Vol.2022-CDS-33 No.3
Vol.2022-DCC-30 No.3
2022/1/20



(a) 各点を1画素として画面に投影した
カラーマップと深度マップ

(b) オクルージョン
推定と除去

(c) レンダリング結果

図 1 画像処理に基づく点群のレンダリング品質の改善
Fig. 1 Overview of point cloud rendering by screen space operators.

ルとした．

また，ある点が遮蔽されていると判断するためには，その点を遮蔽している点が近傍領域に含まれる必要がある．したがって，近傍領域サイズの計算のためには，各グリッド内に存在する点の中で，特に視点に近い手前側に存在する点の密度が重要であると考えられる．そこで本稿では，各グリッドにおける深度値の最小値 $z_{\min}$ を基準として，以下の式 (2) を満たす点 i を数えてグリッドの面積 ($16^2 = 256$) で除算することで密度を計算した．

$$z[i] - z_{\min} < e \tag{2}$$

ただし，e は密度の計算に用いる点を深度値によって区別するためのしきい値であり，本稿では $e = 0.04$ とした．

3.2 密度を用いた近傍領域サイズの計算

文献 [5] では，近傍領域サイズの一辺の長さ $L[i]$ は深度値に反比例するように定められているため，点群に含まれる点同士の間隔が一定であれば，透視投影後の画像空間において，近傍領域に含まれる点の数はおおよそ一定になると考えられる．本稿ではこのことに着目し，近傍領域に含まれる点の数が一定値 C になるように $L[i]$ を定めることを検討した．具体的には，3.1 節で求めた，画素 i が含まれるグリッドの密度を $\sigma[i]$ として，$\sigma[i] \cdot L[i]^2 = C$ が成り立つように，パラメータ p' を用いて式 (3) のように定めた．

$$L[i] = \frac{p'}{\sqrt{\sigma[i]}} \tag{3}$$

4 節の評価実験により p' の値を実験的に決定した．

最終的な近傍領域サイズの段階 $l[i]$ は，文献 [5] と同様に $l[i] = \log_2 L[i]$ で決定することを検討したが，図 2(a) のように物体の境界付近で近傍領域サイズが大きくなってしまう場合があった．そこで，$l[i]$ のマップに 3×3 のメディアンフィルタを適用することで，図 2(b) のように近傍領域サイズの補正を行った．

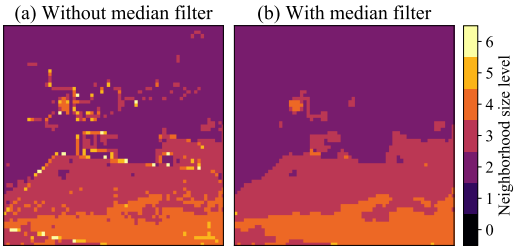


図 2 メディアンフィルタを用いた近傍領域サイズの補正
Fig. 2 Correcting neighborhood size using the median filter.

4. 評価実験

4.1 使用した点群

本稿では，Park らの研究 [13] で使用された，LIDAR スキャナで屋内を測定した RGB カラーを含む点群[*1]を使用した．点群の密度による影響を明確に評価するため，元の点群からサンプリングして作成した点群を用いて実験を行った．具体的には，元の点群から 1000 万点をランダムにサンプリングした高密度な点群と，その点群からさらにランダムに 100 万点をサンプリングした低密度な点群を準備した．また，特定のオブジェクトのみを低密度とするなど，高密度な部分と低密度な部分が存在するように統合した点群も作成した．これらの点群と 2 種類の視点の組み合わせた合計 10 個のシーンで実験を行った．使用した視点と点群の統合の際に着目したオブジェクトを図 3，シーンの詳細を表 1 に示す．

また，文献 [5] では，背景，つまり，点が投影されていない画素についてもオクルージョン推定を行っているが，本稿では屋内環境で取得された点群のみを実験で使用したため，既存手法，提案手法ともに，比較の際には背景のオクルージョン推定は省略し，背景の画素はすべて補間処理による上書きの対象とした．

4.2 品質の評価方法

4.2.1 評価指標

画像の品質の定量的な指標として，PSNR（Peak signal-to-noise ratio）を用いた．PSNR は，ある画像の品質を対応する高品質な基準画像との類似度として定義する指標で，値が大きいほど高品質であることを表す．本稿では，オクルージョン推定の結果を評価するために，深度マップ

[*1] http://redwood-data.org/indoor_lidar_rgbd/

Vol.2022-GN-115 No.3
Vol.2022-CDS-33 No.3
Vol.2022-DCC-30 No.3
2022/1/20

(A) (B)

図 3 使用した視点 (A, B) と着目したオブジェクト
Fig. 3 Viewpoints (A, B) and objects focused on.

表 1 使用したシーン
Table 1 Scenes used in the experiment.

シーン名	視点	密度に関する詳細
(A1)	(A)	高密度（1000 万点）
(A2)	(A)	低密度（100 万点）
(A3)	(A)	高密度（画像左半分）＋ 低密度（右半分）
(A4)	(A)	低密度（table）＋ 高密度（table 以外）
(A5)	(A)	低密度（desk）＋ 高密度（desk 以外）
(A6)	(A)	高密度（desk）＋ 低密度（desk 以外）
(B1)	(B)	高密度
(B2)	(B)	低密度
(B3)	(B)	低密度（desk）＋ 高密度（desk 以外）
(B4)	(B)	高密度（desk）＋ 低密度（desk 以外）

に関する PSNR を算出した．

4.2.2 メッシュと基準画像の作成

点群から作成したメッシュを用いて理想的なオクルージョン推定を行い，その結果を補間することで PSNR の計算に用いる基準画像を作成した．可能な限り公平な基準画像を生成するため以下のようにメッシュを作成した．

まず，CloudCompare[*2] と MeshLab[*3] を用いて，点群に対してノイズの除去と法線の計算を行った．メッシュの生成は，3D データ処理用のライブラリである Open3D[14] により，ball pivoting algorithm（BPA）[15] を適用することで行った．BPA で使用されるパラメータであるボール半径 ρ を適切に設定する必要があり，本稿では ρ を変化させて複数のメッシュを生成し，適切であると考えられる ρ の値を主観で 4 個選択した．その後，それぞれの値で生成されるメッシュを統合して，最終的なメッシュデータとした．また，処理時間の関係から，全体を格子状に分割し，それぞれの部分についてメッシュの生成を行った．分割の際に，格子よりも少し大きく切り出すことで隣接する部分に重なりを生じさせ，後述する正解データの作成に影響が出ないようにした．

パラメータ調整だけでは対応できない部分は，手作業によるメッシュの修正を行った．修正の際は，明らかに不必要なメッシュの削除と，既存の頂点に対して不足しているメッシュの追加のみを行い，頂点の移動や追加，削除は行っていない．

次に，オクルージョン推定の精度を判定するための正解データを構築したメッシュから作成した．各点 i について，深度値 $z_p[i]$ とメッシュをレンダリングした画像の点 i に対応する画素の深度値 $z_m[i]$ を比較し，$z_p[i] - z_m[i] > \epsilon$ であれば点 i は遮蔽されている点とし，そうでなければ遮蔽されていない点としてラベル付けした．なお定数 ϵ は 0.005 とした．各視点における基準画像は，高密度な点群を投影した深度マップに対してオクルージョン推定の正解に従って除去処理を行った結果を，推定時と同様に補間することで作成した．

4.3 フレームレートの評価方法

シーン (A1) において 1000 回の描画を実行するのに要する時間を計測し，1 秒間あたりの描画の実行回数（フレームレート）の平均値を算出した．

4.4 実験環境

実装は Unity 2020.1.14f1 を用い，GPU による並列処理にはコンピュートシェーダを用いた．レンダリングの解像度は 1024×1024 と設定し，実行は 3.40 GHz Intel Core i7-6700 CPU，NVIDIA GeForce GTX 1660 Ti を用いた環境で行った．

4.5 結果と考察

4.5.1 品質の評価

既存手法 [5] と本手法について，各シーンでパラメータ p, p' を変化させたときの PSNR を図 4 に示す．ただし，横軸のスケールを揃えて比較するため，各手法のシーン (A1) における PSNR のピーク位置 ($= p_0,\ p'_0$) を用いて，それぞれ $\hat{p} = p/p_0,\ \hat{p}' = p'/p'_0$ で正規化した値を横軸として設定した．また，PSNR は各シーン，手法における最小値が青，最大値が赤となるようにカラーマッピングして示した．既存手法 [5] では，異なるシーン間でピーク位置が異なる場合があるが，本手法ではピークの位置はどのシーンでも $\hat{p}' = 1$ 付近になっている．この結果を参考に各シーンの品質を確認し，本手法で用いるパラメータの値を $\hat{p}' = 1.2$ と設定した．

次に，PSNR の比較を行った結果を図 5 に示す．比較の際の基準は，各シーンに本手法（$\hat{p}' = 1.2$）を適用したときの PSNR の値とした．図 5 に基準としたパラメータの値をバツ印として示した．また，横軸は図 4 と同様に正規化されたパラメータである．比較結果は，PSNR が 0.2 dB 程度異なると主観的にも違いが分かるという目安に従い，より高品質（基準より 0.2 dB 以上大きい），同程度の品質（基準との差の絶対値が 0.2 dB 以内），より低品質（基準より 0.2 dB 以上小さい）の 3 種類に分類して示した．シーン

[*2] https://www.cloudcompare.org/
[*3] https://www.meshlab.net/

Vol.2022-GN-115 No.3
Vol.2022-CDS-33 No.3
Vol.2022-DCC-30 No.3
2022/1/20

(A2)-(A6)，(B4) では，既存手法でパラメータ調整を行って PSNR を最大化した場合と同程度，あるいは，より高品質な結果が，パラメータを固定した提案手法で得られている．図 6 に既存手法 [5] で PSNR が最大となるレンダリング結果と，提案手法によるレンダリング結果の一部を示す．提案手法の方が高品質な結果が得られるシーン (A6) では，提案手法の方が物体の境界付近のブロックノイズが少ないように見える．これは，特に点群の密度が不均一な場合，深度を用いるよりも密度を用いた方が密度がより適切な近傍領域サイズが求まる場合があるためだと考えられる．

一方で，図 5 を見ると，シーン (A1)，(B1)-(B3) のように，調整を行った既存手法よりも提案手法の方が低品質となる場合もある．例えばシーン (B3) では，図 6(B3) のように，提案手法でディスプレイが透けているのが目立つ．原因の一つとして，密度の計算時にしきい値 e による手前側の点の区別が適切にできておらず，ディスプレイ後方の壁部分の点も手前の点として数えられ，密度が正しく計算できていないことが考えられる．したがって，本手法で用いたしきい値 e の適切な値は，シーンによって異なる場合があると考えられる．また，シーン (A4) では，図 6(A4) のように，提案手法ではテーブルの一部に穴が開いている．原因の一つとして，密度を計算する際のグリッドのサイズが不十分で，テーブル部分の点が含まれないグリッドが存在し，密度が正しく計算できない場合があることが考えられる．またシーン (B1) では，図 6(B1) のように，提案手法で机の脚の周りに黒いノイズが目立っている．点群を確認すると，画面上のノイズの位置には点群自体のノイズと思われる点が存在しており，このようなノイズの点が密度の計算に影響し，提案手法の品質低下につながったと考えられる．

また，本稿では提案手法のパラメータを $\hat{p}' = 1.2$ としたが，シーン (B1)，(B3) ではより高品質な結果が得られるパラメータが存在する．よって，ピークの位置は厳密には揃っておらず，原因としては，上記のような品質を低下させる要因によってシーンによって少しずつピーク位置が異なっているということが考えられる．

以上より，特定のシーンではレンダリング品質に改善の余地があり，本手法によって品質が低下してしまう条件の調査や，密度計算の改善手法やノイズへの対策の検討が必要であると考えられる．また，本稿ではシーンによらず同じ値のパラメータを使用する手法を検討したが，より多様なシーンで品質の改善を可能とするため，最適なパラメータを自動で選択するような手法の検討も有効だと考えられる．

4.5.2 フレームレートの評価

シーン (A1) におけるフレームレートは，既存手法 [5] ($\hat{p} = 1$) は 209 FPS（Frames per second）であり，本手法は 187 FPS であった．本手法のフレームレートは既存手法のものよりも低下するが，より高性能なハードウェアによるフレームレートの改善などが可能であることを考慮すると，十分に実用的な値であると考えられる．ただし，密度計算の近似や並列化など，アルゴリズムの工夫によるフレームレートの改善の余地はあると考えられる．

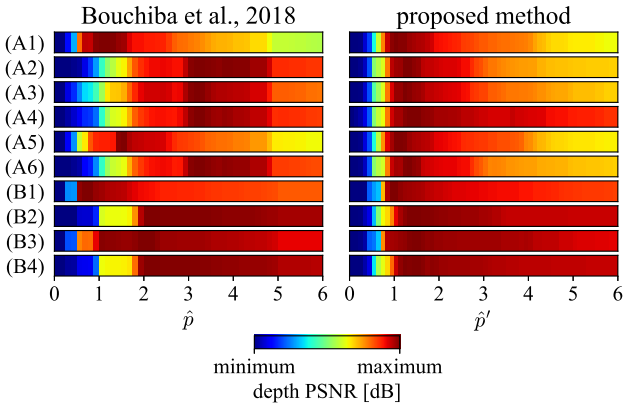


図 4 各手法，シーンにおける PSNR
Fig. 4 PSNR for each method and scene.

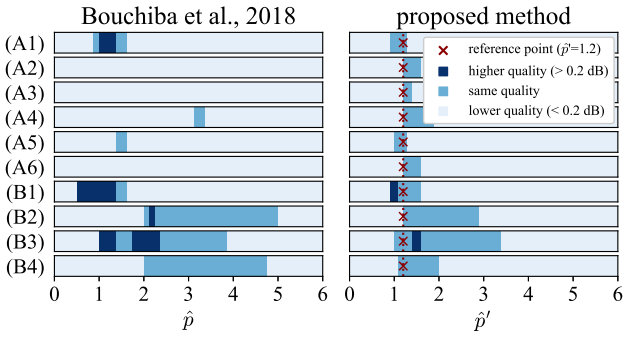


図 5 各手法，シーンにおける PSNR の比較
Fig. 5 Comparison of PSNR for each method and scene.

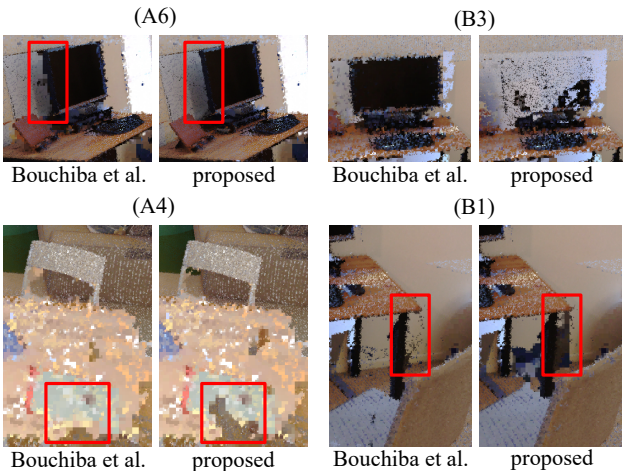


図 6 レンダリング結果の比較
Fig. 6 Comparison of the rendering results.

5. おわりに

本稿では，点群の高速かつ高品質なレンダリング手法において，点群の密度に応じたパラメータ調整を行うことなくレンダリング品質の改善を可能とするため，画像空間における局所的な密度を考慮したオクルージョン推定を提案し，有効性を評価した．評価実験では，既存手法でパラメータ調整を行った結果と同程度，あるいは，それ以上の品質の結果が本手法によって得られる場合がある事が確認され，本手法の有効性が示された．一方で，密度計算が不正確であることに起因する品質の低下といった課題も確認された．今後は，密度計算の改善手法の検討などを通して，品質の改善に取り組む予定である．

謝辞 本研究の一部は JSPS 科研費 19H04150 の研究助成によるものである．ここに記して謝意を表す．

参考文献

[1] Poux, F., Valembois, Q., Mattes, C., Kobbelt, L. and Billen, R.: Initial User-Centered Design of a Virtual Reality Heritage System: Applications for Digital Tourism, *Remote Sensing*, Vol. 12, No. 16, p. 2583 (2020).

[2] Martin-Brualla, R., Pandey, R., Yang, S., Pidlypenskyi, P., Taylor, J., Valentin, J., Khamis, S., Davidson, P., Tkach, A., Lincoln, P., Kowdle, A., Rhemann, C., Goldman, D. B., Keskin, C., Seitz, S., Izadi, S. and Fanello, S.: LookinGood: Enhancing Performance Capture with Real-Time Neural Re-Rendering, *ACM Trans. Graph.*, Vol. 37, No. 6 (2018).

[3] Richardt, C., Tompkin, J. and Wetzstein, G.: Capture, Reconstruction, and Representation of the Visual Real World for Virtual Reality, *Real VR–Immersive Digital Reality*, Springer, pp. 3–32 (2020).

[4] Ponto, K. and Tredinnick, R.: High-Resolution Interactive Immersive Renderings of Real-World Environments, *2020 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, pp. 825–826 (2020).

[5] Bouchiba, H., Deschaud, J.-E. and Goulette, F.: Raw point cloud deferred shading through screen space pyramidal operators, *Proceedings of the 39th Annual European Association for Computer Graphics Conference: Short Papers*, pp. 25–28 (2018).

[6] Bonatto, D., Rogge, S., Schenkel, A., Ercek, R. and Lafruit, G.: Explorations for real-time point cloud rendering of natural scenes in virtual reality, *2016 International Conference on 3D Imaging (IC3D)*, pp. 1–7 (2016).

[7] 森島正博，小川剛史：三次元点群の高速・高品質な可視化のためのオクルージョン推定に関する一検討，VR 学研報，Vol. 26, No. CS-1, pp. 1–6 (2021).

[8] Virtanen, J.-P., Daniel, S., Turppa, T., Zhu, L., Julin, A., Hyyppä, H. and Hyyppä, J.: Interactive dense point clouds in a game engine, *ISPRS Journal of Photogrammetry and Remote Sensing*, Vol. 163, pp. 375–389 (2020).

[9] Mekuria, R., Blom, K. and Cesar, P.: Design, implementation, and evaluation of a point cloud codec for tele-immersive video, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 27, No. 4, pp. 828–842 (2017).

[10] Zwicker, M., Pfister, H., Van Baar, J. and Gross, M.: Surface splatting, *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pp. 371–378 (2001).

[11] Schütz, M. and Wimmer, M.: High-quality point-based rendering using fast single-pass interpolation, *2015 Digital Heritage*, Vol. 1, pp. 369–372 (2015).

[12] Pintus, R., Gobbetti, E. and Agus, M.: Real-time rendering of massive unstructured raw point clouds using screen-space operators, *Proceedings of the 12th International conference on Virtual Reality, Archaeology and Cultural Heritage*, pp. 105–112 (2011).

[13] Park, J., Zhou, Q.-Y. and Koltun, V.: Colored point cloud registration revisited, *Proceedings of the IEEE International Conference on Computer Vision*, pp. 143–152 (2017).

[14] Zhou, Q.-Y., Park, J. and Koltun, V.: Open3D: A Modern Library for 3D Data Processing, *arXiv:1801.09847* (2018).

[15] Bernardini, F., Mittleman, J., Rushmeier, H., Silva, C. and Taubin, G.: The ball-pivoting algorithm for surface reconstruction, *IEEE transactions on visualization and computer graphics*, Vol. 5, No. 4, pp. 349–359 (1999).