# 設備の目視検査に適した大規模点群の立体視手法

A Stereoscopy of Point Clouds Suitable for Visual Inspection of Facilities

岡本大樹†, 増田宏†
Hiroki OKAMOTO† and Hiroshi MASUDA†

†: 電気通信大学大学院情報理工学研究科, h.masuda@uec.ac.jp

概要: 測量用のレーザスキャナの進歩により, 3 次元環境の幾何情報を高密度点群データとして取得できるようになった. 取得された点群は解像度が高いため, 実世界の 3 次元環境を仮想空間に投影でき, 目視検査等への応用が期待できる. そこで, 本研究は立体視デバイスを用い, 点群を立体視することで 3 次元的な空間認識を支援する手法を提案する. 立体視デバイスを用いた場合, 視線に描画が追従するほどのレンダリングのリアルタイム性と, 高い描画品質が要求される. 本手法では, 立体視に適した可視点抽出を行うことでそれらを実現する.

## 1. 緒言

レーザスキャナの普及により, 多くの現場で点群データを用いた検査作業等の効率化が期待されている. また一方で, 近年安価な立体視デバイスが開発されており, このような立体視デバイスを用い, 点群を三次元的に認識することで, 遠隔地の検査対象に対しての目視検査支援が可能であると考えられる.

立体視デバイスを用いた場合, 映像が視線方向に対して追従するリアルタイム性が必要である. 点群のリアルタイムレンダリングでは様々な研究が為されているが [1-7], その多くが数百万点ほどの小規模な点群に対してのものであり, 数千から数億点を保持する大規模点群に適していない. そこで, 立体視デバイスに適した大規模点群のレンダリング手法について考える.

大規模点群を高速にレンダリングするためには, 点を間引くことが有効である. しかし, 点を間引いた場合, 点群の隙間から本来隠蔽される点が露出し, 描画品質が劣化する. 一方で, 隙間を埋めるために点の描画サイズを大きくすることも考えられるが, 検査対象の細部の情報が失われるといった問題がある.

そこで本研究では, 設備の目視検査において大規模点群の立体視を活用することを目的とし, この目的に適した大規模点群の立体視手法を考える.

## 2. 本手法の概要

### 2.1. 点群

本研究では, 図 1 のような位相差方式のレーザスキャナによって取得された点群を使用する. このような固定式レーザスキャナは, 仰角・方位角の 2 自由度を持ち, レーザ光を全方位に照射することで, 点群データを取得する. 取得された点群は, スキャナを中心とする三次元座標とレーザ反射強度を保持している. また, スキャナにカメラを搭載することで, 色情報を付加することもができる.

固定式レーザスキャナでは全周囲の点群データを取得するため, 数千万点の点群を短時間で取得することが可能である. また, 固定式レーザスキャナによる計測では複数箇所で計測することが多く, そのような場合, 数億点を越える点群規模になる可能性もある. 本稿では, 通常のデジタル画像と同等以上の解像度で取得された点群を総称して大規模点群と呼ぶ.


図 1 位相差式レーザスキャナ

### 2.2. 立体視デバイス

ヘッドマウントディスプレイ(HMD)型の立体視デバイスでは，左右の目にそれぞれ視差のある映像を見せることで，疑似的に両目視差立体視を実現する．

本研究では，安価な立体視デバイスとして Oculus VR 社の Oculus Rift を使用する(図 2)．Oculus Rift は一般的な液晶ディスプレイ(1920×1080)と魚眼レンズからなり，ディスプレイに，図 3 のような映像を投影し，この映像を魚眼レンズにより補正することで広視野角を実現している．また，ユーザの頭の向きに映像が追従するヘッドトラッキングが実装されている．

図 2 立体視デバイス

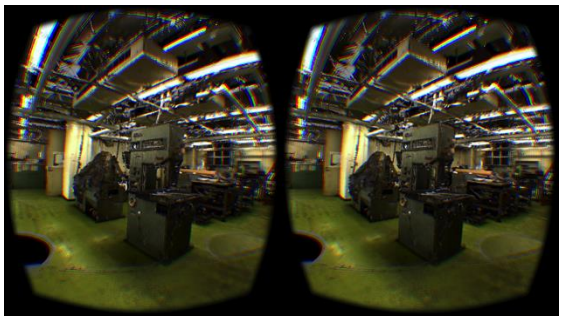


図 3 立体視デバイスでの視差映像

### 2.3. 本手法のシステム構成

本手法を用いたシステムの構成を図 4 に示す．本システムでは，Oculus とコントローラよりユーザの視線方向と位置情報を取得し，それに応じて描画の更新を行う．また，大規模点群への適用と処理の高速化の観点から，描画に用いる点群はメモリ上に保持するものと HDD 上に保持するものに分ける．HDD 上に保持する点群は前処理として算出する．各モジュールについて以下に示す．

- **Oculus**：Oculus Inc.によって開発された立体視デバイスで，ジャイロセンサを用いユーザの視線方向の変化を検出する．また，Oculus SDK によって生成された図 3 のような立体視画像の表示を行う．
- **Controller**：ユーザ操作による位置情報の変化を取得する．本研究では，市販のゲーム用機器を用いた．
- **Oculus SDK**： Oculus Inc.が提供している SDK で，ユーザの位置情報，視線方向の情報をレンダリングエンジンに送信し，描画を更新する[12]．また，レンダリングエンジンによって生成された画像を図 3 のような Oculus に適した樽型形状の画像へと変換する．
- **Rendering Engine**：Oculus SDK より与えられた視点位置，視線方向の情報より，3 次元環境をレンダリングする．本研究では，レンダリングエンジンとして OpenGL を使用した．
- **Visible Points Extractor**：本研究で提案する大規模点群から可視点抽出を行うモジュールである．描画を更新する際は，RAM 上に保持したデプスマップにアクセスし，可視点情報をレンダリングエンジンに送信する．ただし，視点が一定距離以上移動した場合，HDD 上から描画に使用する可視点群を選択し，RAM 上へ読み込むことで描画する可視点群を更新する．また，コントローラから取得された視点位置の変化量から視点位置を算出し，Oculus SDK で用いられる座標系へ変換する．

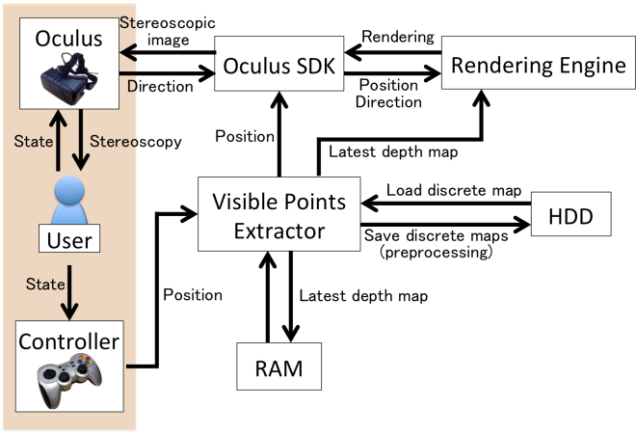



図 4 システム構成図

### 2.4. 大規模環境点群の立体視

目視検査を考えた場合，物体や空間を認識できる程度の描画品質が必要である．また，検査を行う場合，特定の位置から対象を注視することから，視点移動中の高精細な描画は必要ないといえる．

そこで，本手法では特定の位置で周囲を見渡す際は，視線追従の可能な高速かつ高精細なレンダリングを行うものとし，視点移動中の描画品質の劣化は許容するものとする．ただし，視点位置が停止した際は， 短時間の前処理時間の後，高速かつ高精細なレンダリングを実現する．

ここで，頭を傾けて周囲を見渡すことは稀であると考え，視点位置での頭の運動は水平方向の回転のみとする．

回転運動に対して描画が追従する必要がある．しかし，既存のレンダリング手法では，回転運動に追従するほどの処理速度を実現しようとすると，点を間引かなくてはならず，描画品質の著しい劣化が生じる．そのため，HMD への実装には適していない．

そこで，本手法では描画に最適な点群を判別し，必要最低限の点群のみをレンダリングする．ここで，描画に最適な点群とはユーザから可視となる点群のことである．元の大規模点群から可視点群のみを選別して描画することで，描画点数を削減し，レンダリングの高速化が臨める．また，可視点は全て描画対象となることから，品質の劣化が生じない．

ユーザが移動した場合，視点位置の変化に伴い，可視点も変化する．しかし，その都度可視点群の選別を行うのはコストがかかり，レンダリングのリアルタイム性の欠如に繋がる．目視検査を想定した場合，固定位置からの高精細な描画が実現できればいいため，ここでは，ユーザが一定距離を移動した場合のみ，描画点群の切り替えを行う．また，その際使用する可視点群は，予め前処理として選別を行い， HDD 上に保持しておく．

## 3. 可視点群の抽出

### 3.1. デプスマップの生成

固定視点からの可視点を抽出するために，左右の目を中心とする全周パノラマ距離画像（図 5）を生成する．ここで，この距離画像を「デプスマップ」と呼ぶ．デプスマップ上で可視でない点は，視点位置から見えない点であるため，描画対象から除去する．これにより，固定視点からの可視点抽出を行う．

### 3.2. 左右の目からの可視点

立体視を行う場合，左右の目からの映像を生成する必要があるため，左右の目を中心とするデプスマップを 2 つ生成する必要がある．このとき，視差が微小であるため，左右の目からの可視点は共通するものが多く存在する．そこで本手法では，片方の目のデプスマップを生成し，もう一方の目のデプスマップは

図 5 デプスマップ

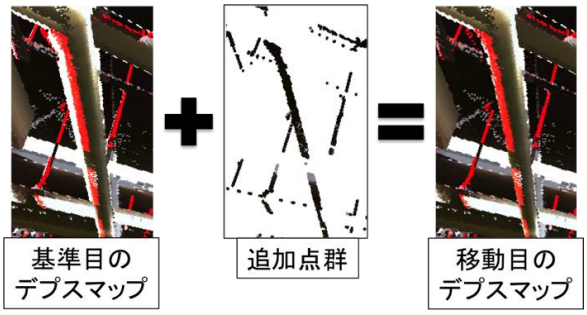



図 6 追加点群の取得と移動目のデプスマップ

そこに点を追加することで生成する．

ここで，デプスマップを生成する方の目を「基準目」と呼び，頭の回転中心とする．もう一方の目は「移動目」とし，基準目を中心とした円軌道を動く．

移動目で追加する点群は，基準目では不可視となる点である．このような点は，基準目のデプスマップ上で物体のエッジ付近に生じる．そこで，デプスマップ上で奥行き値を比較し，エッジ箇所を検出する．次に検出されたエッジ付近において，視差と奥行き値の変化から移動目からの可視点を取得する．図 6 は，基準目のデプスマップに移動目からの可視点を追加した結果である．図 6 の赤い部分は物体のエッジにあたる点群である．追加点を取得することで，赤点付近に存在した隙間が補間されたことがわかる．

## 4. 視点移動時の描画

目視検査を目的とするため，視点移動時の描画の劣化は許容する．ただし，視点移動中は最低限周囲の環境を把握できる必要がある．

特定位置からの可視点群を使用し，視点移動を行った場合，移動前に不可視点だったものが可視点となることがある．しかし，可視点抽出の際，そのような点群は破棄されているため，移動後の視点からの本来可視点となる点がレンダリングされず，描画品質を維持できない．

視点の移動距離が微小であった場合，可視不可視が変化する点が少ないため，ある程度の周囲環境の認識は行えると考えられる．そこで，図 7 の青点のように予め点群環境内に離散的にデプスマップを保持することを考える．視点移動の際，常に視点から最も近い離散点の可視点群を用いてレンダリングを行うことで，周囲環境を認識できる程度の描画品質を維持する．

たとえば，視点位置が図中赤い軌跡の用に動いた場合，視点位置から最近となる離散点を算出し，$P_1$, $P_2$, $P_3$, $P_4$, $P_5$ のように可視点群を順々に切り替える．これにより，描画品質は劣化するものの，周囲環境を把握でき，かつリアルタイムなレンダリングを維持できる．このとき，離散点での可視点群は前処理として算出され，HDD 上に保持しておく．

点群が疎である領域において，離散点を生成し，可視点抽出を行った場合，十分な3次元環境の認識を行えない可能性がある．そこで，離散点の生成箇所の範囲を主成分分析によって求められた点群の分布から決定する．

まず，離散デプスマップの生成範囲を決定する．点群を地面と水平な平面に投影し，それらに対し主成分分析を施す．次に，求められた第一，第二主成分のスコアを量子化する．各スコアにおける点群数を算出し，閾値以下となる領域を離散点の生成範囲から除外する．ここでは，閾値を全点数の 3%とした．

生成した離散点からの可視点群は，バイナリ形式で HDD 上に保持し，視点移動の都度，読み込んで描画を切り替える．ただし，視点位置が停止した場合，可視点抽出を行い，高品質なレンダリングを行う．

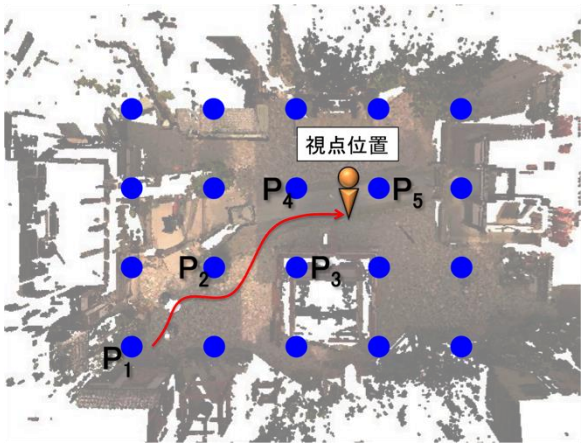


図 7　離散デプスマップ

## 5.　本手法の評価

### 5.1.　実験環境

ここでは，立体視デバイスとして Oculus Rift DK2 を用いた．検証では，図 8 に示す 3 箇所から測定された約 9000 万点の屋内環境点群を使用した．また，検証の際の視点位置は，点群の計測原点の中点付近とした．本手法の検証に用いた PC は Intel Core i5-4440 CPU (3.10GHz), 16GB RAM であり，GPU は NVIDIA GeForce GT 640 を用いた．

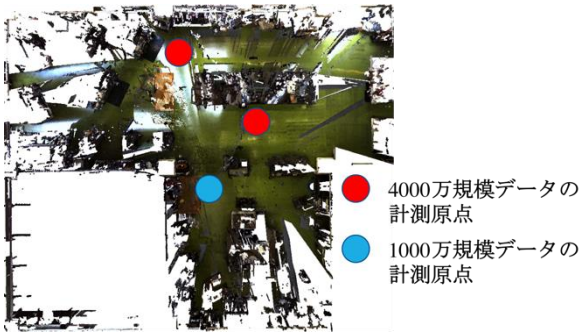


図 8　実験環境の点群

### 5.2.　可視点抽出の処理時間

可視点抽出までの前処理時間を検証した．結果を表 1 に示す．点群数が増えるにつれ，基準目のデプスマップの生成時間が増加していることがわかる．約 9000 万点規模の点群で約 9 sec ほどの処理時間を要した．この処理時間は前処理時間であるため，以降の点群のレンダリングでは常に高速かつ高品質なレンダリングが可能となる．

表 1　可視点抽出の前処理時間

| 総点数 | デプスマップの構築 | 追加点の取得 | 合計 |
|---|---|---|---|
| 1000 万点 | 1.68 | 1.93 | 3.61 |
| 5000 万点 | 4.94 | 1.93 | 6.87 |
| 9000 万点 | 6.86 | 1.94 | 9.03 |

### 5.3.　離散デプスマップの生成と読込時間

図 8 中の 4000 万規模の点群 2 つを用いて離散デプスマップを生成した．離散デプスマップの生成間隔は 2m とした．生成箇所は図 9 のようになった．15 個の離散デプスマップを生成するに要した時間は 80.7 sec である．

視点移動時の離散デプスマップの読み込み時間は

約 0.30 秒と高速に読み込むことができた．視点移動後は，デプスマップを再生成するため，6.2 節の可視点抽出の処理時間ののち高速かつ高品質なレンダリングを行う．

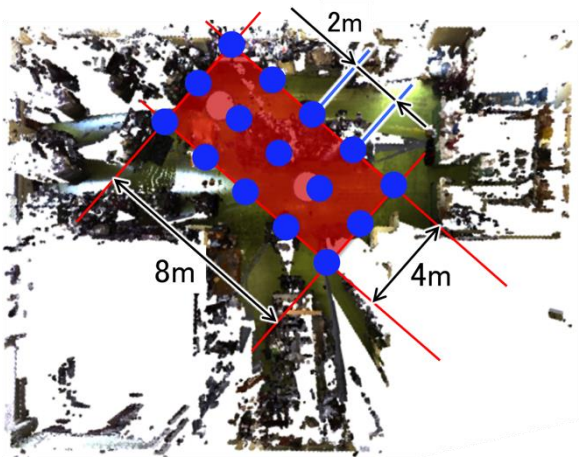



図 9　離散点の生成

### 5.4.　リアルタイム性と描画品質

没入型 HMD の利用において，映像の遅延はユーザに不快感を与える可能性がある．そのため，レンダリングには高いフレームレートが求められる．ここでは，満たすべきフレームレートを 75 fps と定める．

最適なフレームレートを実現するためには，描画点数を約 100 万点まで減らす必要がある．しかし，描画点数を減らした場合，図 11(a)のように著しく描画品質が劣化する．

本手法を用いてレンダリングを行った場合のデプスマップ解像度とフレームレート関係を図 10 に示す．デプスマップ解像度を 2430×1215 と定めた場合，75 fps を越えるフレームレートを実現した．また，描画品質においても，図 11(b)に示す通り，高精細にレンダリングできていることがわかる．

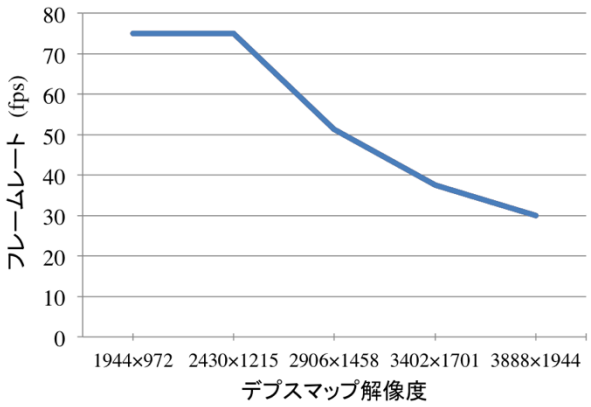
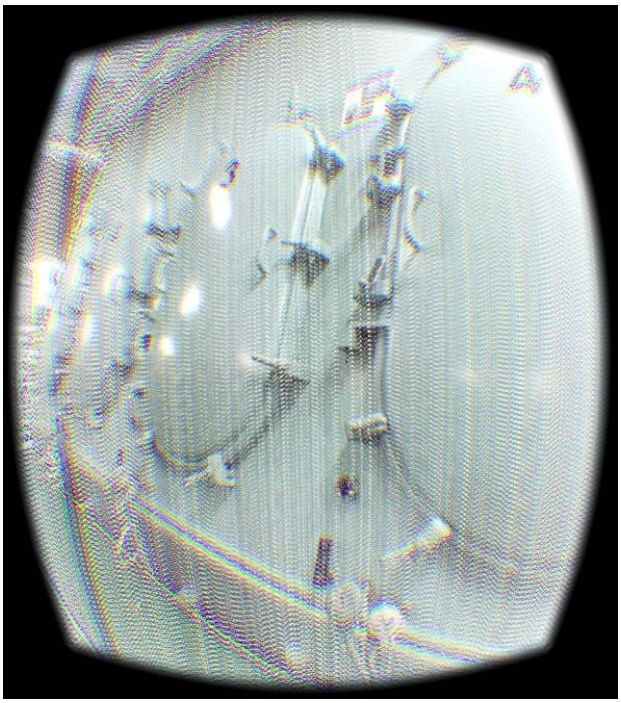



図 10　デプスマップ解像度とフレームレート

## 6.　結言

デプスマップを用いることで，可視点抽出を容易に行い，描画品質を劣化させない高速な点群のレンダ



(a)約 100 万点の点群

(b)本手法適用時

図 11　立体視デバイス上での描画品質

リング手法を提案した．また，離散デプスマップを用い，予め視点移動時の可視点群を取得しておくことで，視点移動時のリアルタイム性を確保した．また，それらの可視点群をHDD上に保持することで，メモリワークを考慮した描画の切替えを行った．これらより，目視検査に適した大規模点群の立体視手法を提案した．

今後は，検査対象の計測などのインタラクションの追加やアノテーションによる仮想空間での情報管理の手法について検討していく予定である．

## 参考文献


[1] S. Rusinkiewicz and M. Levoy: QSplat: A multiresolution point rendering system for large meshes, In Proc. ACM SIGGRAPH, Computer Graphics Proceedings, pp.343-352, 2000.

[2] C. Dachsbacher, C. Vogelgsang and M. Stamminger: Sequential point trees, In Proc. SIGGRAPH, Computer Graphics Proceedings, pp.657-662, 2003.

[3] 岡本泰英，山崎俊太郎，池内克：Sequential Point Clusters: 大規模モデルに対する効率的なポイントベースドレンダリングシステム，情報処理学会論文誌，vol.46, no.1, pp.1234-1237, 2005.

[4] R. Pajarola, M. Sainz and R. Lario: Xsplat: External memory multiresolution point visualIzation, In Proceedings IASTED International Conference on Visualization, Imaging and Image Processing, pp.628-633, 2005.

[5] S. Katz, A. Tal and R. Basri: Direct visibility of point sets, ACM Transactions on Graphics (TOG), vol.26, no.3, 24, 2007.

[6] J. Liang, F. Park and H. Zaho: Robust and efficient implicit surface reconstruction for point clouds based on convexified image segmentation, Journal of Scientific Computing, vol.54, no.2-3, pp.577-602, 2013.

[7] R. Preiner, S. Jeschke and M. Wimmer: Auto Splats: Dynamic Point Cloud Visualization on the GPU, EGPGV, pp.139-148. 2012.

[8] R. Machado e Silva, C. Esperanca, R. Marroquim and A. A. F. Oliveira: Image Space Rendering of Point Clouds Using the HPR Operator, Computer Graphics Forum, pp.178-189, 2014.

[9] C. Wei-Chun and C, Wen-Chih: Real-Time 3D Rendering Based on Multiple Cameras and Point Cloud, Ubi-Media Computing and Workshops (UMEDIA), 2014 7th International Conference on IEEE, pp.121-126, 2014.

[10] J. P. Wann, S. Ruston and M. Mon-Williams: Natural problems for stereoscopic depth perception in virtual environments, Vision research, vol.35, no.19, pp.2731-2736, 1995.

[11] K. Ukai and P. A. Howarth: Visual fatigue caused by viewing stereoscopic motion images: Background, theories, and observations, Displays, vol.29, no.2, pp.106-116, 2008.

[12] Oculus VR and LLC: Oculus Rift Developer Guide, pp.8-16, 2015.



**岡本大樹**：2014 年，電気通信大学情報理工学部知能機械学科卒業．同年，同大大学院情報理工学研究科修士課程在学中．バーチャルリアリティ，コンピュータグラフィックスに関する研究に従事．

**増田宏**：1985 年，東京大学工学部精密機械工学科卒業．1987 年，同大大学院工学系研究科修士課程修了．同年，日本アイ・ビー・エム（株）入社，東京基礎研究所に勤務．1998 年，東京大学大学院工学系研究科 准教授，2013 年より電気通信大学大学院情報理工学研究科 教授．形状モデリング，3次元計測，コンピュータグラフィックス，CAD に関する研究に従事．工学博士．