# 大規模点群のための高速な干渉判定手法*

丹羽　健**　増田　宏***

Efficient Collision Detection for Large-Scale Point-Clouds

Takeru NIWA and Hiroshi MASUDA

Terrestrial laser scanners can capture dense point-clouds from engineering plants. Dense point-clouds are useful for collision detection of objects that move in engineering plants. We propose a real-time collision detection method for large-scale point-clouds. In our method, statuses are classified into collision, no collision, and unknown. The unknown status indicates that points are missing at the object position because of occlusion. Collisions are evaluated on depth-maps, which are generated from point-clouds. Two-resolution of depth-maps are used for improving performance. High-resolution depth maps are used only when objects collide on rough resolution depth maps. Our experimental results show that our method could quickly detect collisions between a large-scale point-cloud and mesh models.

**Key words**: point- cloud, laser scanner, collision detection

## 1. 緒　　言

近年，レーザスキャン技術が向上し，大規模な構造物の高密度点群を短時間で取得することが可能になった．このような大規模な点群データは，現況に忠実な3次元情報を保持している．計測によって得られた現物の大規模点群データを用いて，CADなどで作成した仮想物体との干渉判定が容易に行なえるならば，生産ラインやプラントなどの改修における搬入搬出の検討などが効率的に行なえることが期待できる．

しかしながら，設備の大規模点群とCADモデルとの干渉判定を行なうためには，いくつかの問題点がある．

1つの問題は，大規模設備を計測した点群データは大容量となるため，大規模点群との干渉判定を高速に行なうための手法が求められることである．工業設備は，多くの装置や部材が密に配置されていることが多く，十分な密度で点群を取得するには，数千万から数億点の点群が必要である．点群をメッシュモデルに変換できれば，古典的な干渉計算手法を利用できるが，ポリゴン数が非常に大きくなるため，メモリ容量や計算時間において問題が生じる．

もう1つは，点群の不完全性の問題である．生産設備やプラントでは，計測位置が制約されるために，計測点の欠落が避けることができない．その場合，点群と仮想物体の干渉判定において，計測点が存在しない場所において「干渉しない」と判定してしまう恐れがある．干渉がない場合には，本当に干渉しないのか，他の物体に遮られて計測点が取得できていないのかを区別して扱うことが必要となる．

CADやCGにおいて，干渉判定は古典的な手法であるが，点群を対象とした干渉判定の研究は多くない．Figueiredo[1]ら，Pan[2]らは複数の点群間の干渉判定を，Octreeを用いて高速化する手法を提案した．Klein[3]らは2つの点群モデル間の干渉を陰関数曲面によって判定する手法を提案した．また，Radwan[4]らは，点群と仮想物体のレンダリング時のデプステストを利用した干渉判定を行っている．いずれの手法でも小規模な点群での評価しか示していないが，提案手法の計算コストを考えると，数千万点規模の点群にそのまま適用することは極めて困難である．一方，平岡[5]らは，大規模点群からデプスマップを生成し，仮想物体をデプスマップに投影することで，干渉判定を行なう手法を提案した．この方法では，点の個数Nに対して，計算コストがO(N)であるため，高速な判定が可能となる．しかしながら，我々の評価実験では，Nが非常に大きくなると，計算コストがO(N)であってもリアルタイムで干渉判定は難しく，デプスマップの解像度を粗くする必要があった．

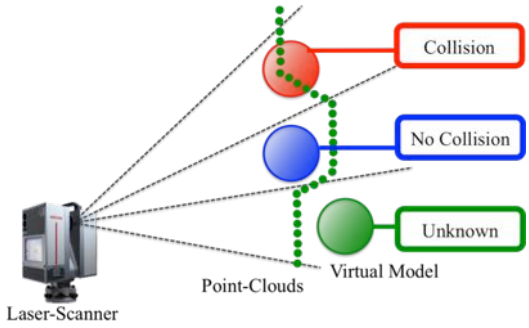



**Fig.1**　Collision detection

そこで，本研究では，2段階のデプスマップを利用することで，測量用のレーザスキャナで取得した大規模点群を用いた高解像度かつ高速な干渉判定手法を提案する．図1に大規模点群を用いた干渉判定の基本的な考え方を示す．レーザ計測では，レーザ光が照射された地点までの距離が計測されるので，その手前には物体は存在していない．従って，図1に示すように，仮想物体が計測点の手前にあるならば干渉は起こらない．干渉が生じるのは，計測点が仮想物体の内部にあるときである．一方，仮想物体が計測点よりも遠い位置にあるならば，その地点には遮られてレーザ光が届かず，計測されていないことを意味するので，干渉不明と判定される．干渉不明のときは，再計測を行なうか，現地で確認する必要がある．

本研究では，このような判定を出来る限り高速に行なうための手法を考える．本手法の利点は以下のようにまとめられる．

* 原稿受付 平成27年 1月 23日
掲載決定 平成27年 N月 N日
** 学生会員 電気通信大学院（東京都調布市布ヶ丘1-5-1）
*** 正 会 員 電気通信大学院（東京都調布市布ヶ丘1-5-1）

(1) 点群を間引かないので，計測された点群の解像度での干渉判定ができる．
(2) ユーザが仮想物体をマウス等で移動させる動きに追従して，リアルタイムで干渉判定が行なえる．
(3) 仮想物体が点群未計測領域に位置した場合には，その検出が行なえる．

以下，第 2 章では提案手法の概要について示し，第 3 章ではデプスマップを使用した干渉判定手法について述べる．第 4 章ではデプスマップを階層化することで処理を高速化する手法を示す．第 5 章では実データに提案手法を適用することで，提案手法の解像度と計算時間を検証する．

## 2. 提案手法の概要

大規模点群と仮想物体との干渉を判定するための処理手順を**図 2** に示す．また，**図 3** には具体的な処理を図示している．

干渉判定の前処理として，点群から，各ピクセルに距離が記述されたデプスマップを生成する．レーザスキャナは緯度と経度に関して等角でレーザ照射角度を変化させて計測を行なうため，計測点群は格子状に並ぶ．そこで，点群座標をレーザ光源が原点となる球面座標で記述し，緯度と経度を主軸とする 2 次元格子に点を配置して，各ピクセルに計測原点からの距離を記述したデプスマップを作成する（図 3 (b)）．また，点群にはノイズが含まれるため，デプスマップ上でノイズ除去処理をする．本研究では，高速化のために，図 3 (c)に示すような解像度を変えた 2 段階の階層的なデプスマップを作成する．

干渉判定はデプスマップを使用して行う．干渉判定は，最初は粗いデプスマップを用いて行なわれ，干渉する可能性がある場合のみ，詳細なデプスマップで判定する．干渉判定は，図 3 (d)のように，仮想物体をデプスマップの各ピクセルに投影することで行なう．仮想物体が閉じた凸形状であるとすると，各ピクセルには，図 3(d)に示すように，仮想物体の 2 つの面が投影され，それらに挟まれた領域が内部の点となる．そこで，図 3 (e)のように，各ピクセルに投影される点の最近点と最遠点を記録しておく．干渉判定は，図 3 (f, g) に示すように，最近点と最遠点のデプスマップと計測点のデプスマップの位置関係によって判定される．その判定結果に基づいて，ユーザが仮想物体を移動させると，図 3 (h) のように，仮想物体の色を変えて干渉の有無をリアルタイムに提示する．

## 3. デプスマップ上での干渉判定

### 3.1 デプスマップの生成

測量用のレーザスキャナから出力される点群は，光源位置を原点とする座標系で記述されている．レーザの照射方向は，方位角$\theta$と仰角$\phi$の 2 方向に関して一定の角周波数で回転する．そのため，点群座標を球面座標$(\theta, \phi, r)$で表すと，点群は $\theta - \phi$平面に規則正しく並ぶ．そのため，計測された$(\theta, \phi)$の範囲をサンプリング角度 $\Delta\theta$，$\Delta\phi$で離散化した画像を生成し，各ピクセルに距離 $r$ を記述することでデプスマップを生成することができる．点群データの反射強度画像とデプスマップを**図 4** に示す．この図では，デプスマップ上で，距離が近い点を明るく表示している．

### 3.2 メッシュモデルの投影

干渉判定は仮想物体をデプスマップ上に投影することで行なう． 干渉判定に使用する仮想物体は閉じたメッシュモデルとする．ここでは，メッシュモデルを構成するすべての三角形をデプスマップ上に投影する．このとき，三角形内部のデプス値は，三角形の各頂点のデプス値を補間して計算する．**図 5** に示すように，三角形の頂点がデプスマップ上の点 $A$，$B$，$C$に投影されるとき，三角形内部の点 $X$でのデプス値を考える．点 $X$を通り横軸に水平な直線が三角形の 2 辺と交わる点を求め，距離 $a$, $b$, $c$,

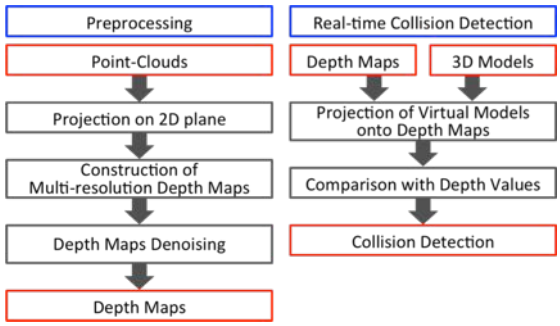
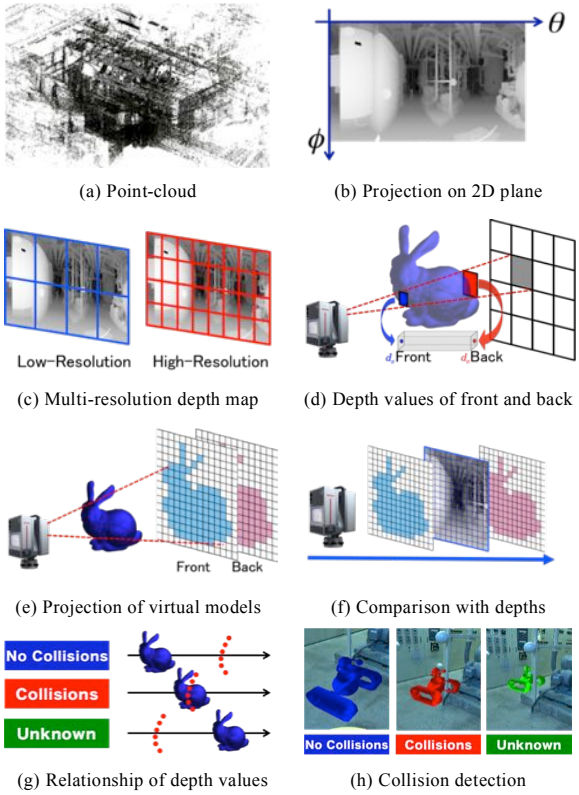



**Fig.2** Process of collision detection




(a) Point-cloud (b) Projection on 2D plane
(c) Multi-resolution depth map (d) Depth values of front and back
(e) Projection of virtual models (f) Comparison with depths
(g) Relationship of depth values (h) Collision detection

**Fig.3** Overview of collision detection process

(a) Reflectance image of point-cloud

(b) Depth map

**Fig.4** Generation of depth maps

$d$, $e$, $f$ を図 5 に示すように決める．このとき，点 $A$, $B$, $C$ のデプス値を $I_A$, $I_B$, $I_C$ とすると，点 $X$ のデプス値，$I_X$ は，以下のように計算できる．

$$I_X = \frac{f(bI_A+aI_B)}{(e+f)(a+b)} + \frac{e(dI_A+cI_C)}{(e+f)(c+d)}$$

厳密には，**図 6** に示すように，直線を $\theta-\phi$ 平面上に投影した図形は直線にはならない．そのため，直線近似しても問題が生じないように，三角形の各辺が十分に小さくなるようなメッシュ分割が必要となる．そこで，本手法を適用する際に必要となるメッシュの三角形の大きさを見積もる．

ここでは，光源から対象物までの水平距離が $D$ のときに，長さ $L$ の直線を $\theta-\phi$ 平面上に投影した場合を考え，直線からのずれが 0.5 ピクセル以下になるような長さ $L$ を計算する．一般に，$D$ が小さいほど，直線からのずれは大きくなる．レーザ計測のサンプリング角度を 0.036°（360° を 10000 分割）とすると，計算の結果，$D$ = 1m のときで 直線の長さ $L$ は 7.3cm 以下，$D$ = 0.5m で 3.5cm 以下となった．従って，干渉を調べたい範囲に応じて，これらの長さ以下になるようにメッシュ分割を行えば，写像の非線形性の影響は受けない．

また，本手法では，デプスマップには最近値と最遠値を記述するため，凹形状の部分での干渉判定を正確に行ないたい場合には，仮想物体を凸形状の集合に分割し，それぞれに対して干渉判定を行なうことが必要である．現在の実装では，凸形状への分割は，ユーザが切断面を設定することで行なっている．

### 3.3 デプス値比較による干渉判定

干渉判定はデプスマップ上のピクセルごとに行なう．干渉判定に使用する仮想物体を閉じた凸形状としているので，仮想物体の投影範囲内には各ピクセルに表面と裏面の２つのデプス値が投影される．物体の表側のデプス値を $d_1$，裏側のデプス値を $d_2$ とする．デプスマップが持つ計測点群のデプス値を $r$ とすると，各ピクセルでの $d_1$，$d_2$，$r$ の値の大小関係によって，仮想物体と点群との干渉状態は**図 8** のように，以下の３通りに分けることができる．

(1) $d_1 > d_2 > r$：干渉なし （図 8 (a)）

計測原点から計測点群まで遮るものがないオープンスペースに仮想物体が置かれているため，このピクセル位置では干渉が生じていないと判定できる．

(2) $d_1 \le r \le d_2$：干渉あり （図 8 (b)）

計測点が仮想物体の内部にあるため，このピクセルで干渉が生じていると判定できる．

(3) $r < d_1 < d_2$：干渉不明 （図 8 (c)）

レーザスキャナでは，物体の背後にある点は取得できないため，**図 9** のように，点群が存在しないオクルージョン領域が発生する．仮想物体がこの領域に投影された場合，そのピクセルは干渉判定には用いることができない．

以上の判定は，仮想物体が投影されるすべてのピクセルについて行なう．１つでも干渉ありと判定されたピクセルがある場合は，「干渉あり」とする．この計算では，１つのピクセルでも干渉があれば，以降のデプス値の比較処理を打ち切る．「干渉あり」のピクセルが存在せず，干渉判定ができないピクセルが存在する場合は，「干渉不明」とする．全てのピクセルで干渉がない場合

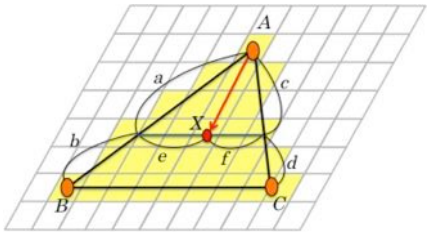
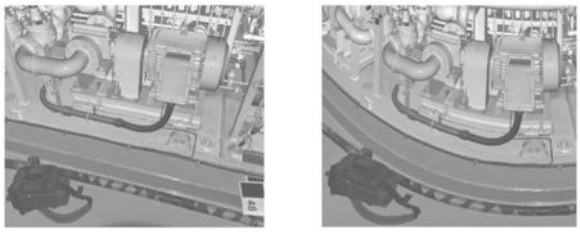



**Fig.5** Projection of mesh onto depth map



(a) Perspective image　　(b) $\theta$ - $\phi$ image

**Fig.6** Distortion of straight lines on $\theta$ - $\phi$ plane

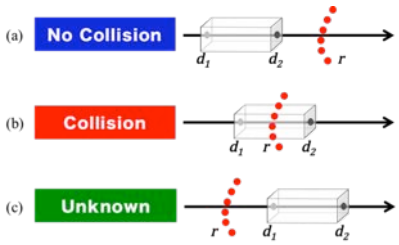



**Fig.7** Relationships among depth values

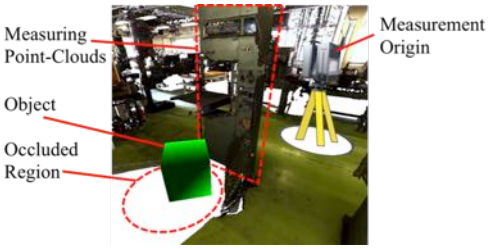



**Fig.8** Object placed in an occluded region

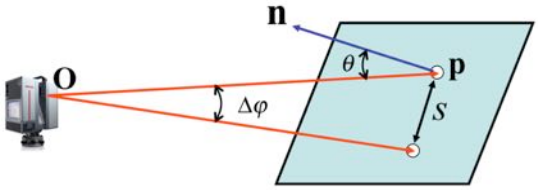



**Fig.9** Detection of outliers using neighbor points

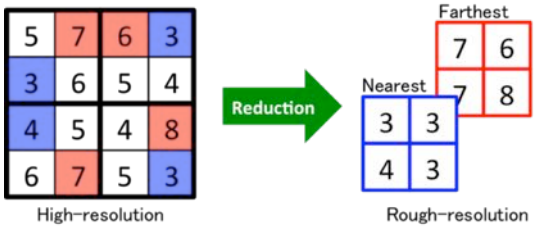



**Fig.10** Generation of multi-resolution depth maps

に限り，「干渉なし」と判定される．

### 3.4　デプスマップ上でのノイズ処理

一般に計測点群には異常値が含まれるため，干渉が誤判定される可能性がある．そこで，計測点から異常値の除去を行なう．

異常値は周囲の点群に比べて計測原点からの距離が大きく異なることが特徴である．そこでデプスマップ上の各ピクセルにおいて 4 近傍とのデプス値の比較を行なうことで異常点を判別する．**図 10** に示すように，点 p と隣接点が同一平面上に乗っているとき，点間距離 $s$ は近似的に以下の式で計算できる [4]．

$$s = \frac{|p|}{cos\theta}\Delta\varphi$$

ここで $\Delta\varphi$ はサンプリング角度（ラジアン），$\theta$ は点 p の位置ベクトルと平面の法線 $n$ のなす角度である．対象とするピクセルが異常値である場合，角度 $\theta$ は大きくなる．本研究では，閾値を $\theta = 75°$とした．点 p のすべての隣接点に関して，点間距離がこの式で計算される距離　$s$ よりも大きいならば，点 p は異常値と判定されて除去される．

## 4.　階層的なデプスマップ

### 4.1　階層デプスマップの生成

干渉判定を高精度で行なうためには，デプスマップを高解像度にする必要がある．一方で，解像度を高くした場合，デプス値を比較すべきピクセル数は増加し，リアルタイム性が損なわれてしまう．そこで，本研究では，干渉判定の精度を低下させることなく高速化を実現するために，階層的なデプスマップを用いる．

**図 10** に基本的な考え方を示す．粗いデプスマップを作成するために，詳細デプスマップを等間隔に区分し，各ブロックで最大と最小のデプス値を求める．そして，最大値と最小値のそれぞれを用いて，2 つの粗いデプスマップを生成する．干渉判定は，1 つの詳細デプスマップと 2 つの粗いデプスマップを用いて行なう．本研究では，ブロックの大きさを16×16とした．従って，粗いデプスマップでは，ピクセル数が約 256 分の 1 に削減される．

### 4.2　階層デプスマップ上での干渉判定

干渉判定では，まず粗いデプスマップを用いて判定を行い，粗いデプスマップ上で干渉の有無が判定できない場合に限り，詳細なデプスマップを用いて判定される．詳細なデプスマップを用いる場合でも，判定に必要なピクセル領域が限定されるため，干渉判定に要する計算時間を大幅に削減することができる．

まず，粗いデプスマップ上での干渉判定について説明する．「干渉なし」「干渉不明」の多くのケースでは粗いデプスマップのみで判定可能である．**図 11** のように，仮想物体を粗いデプスマップ上に投影したときの最近と最遠のデプス値を $d_1$，$d_2$ とし，またそのときの 2 つの粗いデプスマップのデプス値を $r_s$（最近値），$r_e$（最遠値）とする．

このとき，以下の条件が満たされるとき，粗いデプスマップのみから正確な判定が可能である．

(1)　$d_1 > d_2 > r_s$：干渉なし　（図 12(a)）

(2)　$r_s < d_1 < d_2$：干渉不明　（図 12(b)）

仮想物体が投影されるすべてのピクセルにおいて，デプス値が最近値 $r_s$よりも小さい場合，仮想物体は詳細デプスマップにおいても，計測点群の手前にあることが保証されるので，粗いデプスマップのみで「干渉なし」であると判定できる．同様に，粗いデプスマップで，仮想物体の全てのデプス値が最遠値$r_e$より大きい場合は「干渉不明」であると判定できる．

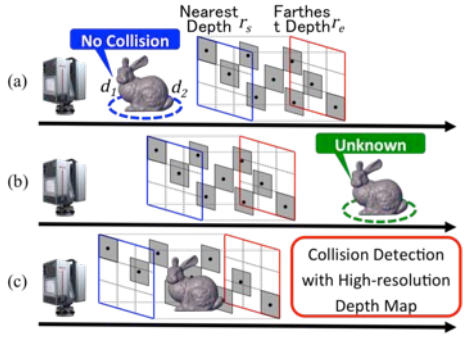
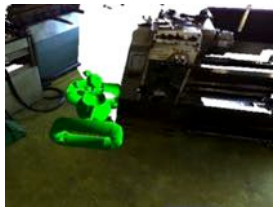



**Fig.11**　Relationships of depth values on multiple depth map

(a) No collisions

(b) Collisions

(c) Unknown



(d) Unknown

**Fig.12**　Results of collision detection

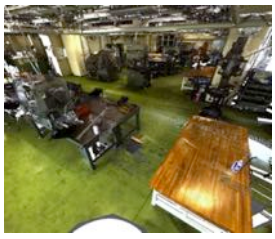


**Fig.13**　Example of point-clouds

それ以外の場合においては，詳細なデプスマップを使用して干渉判定を行なう．その場合でも，粗いデプスマップにおいて「干渉なし」「干渉不明」のいずれでもないピクセスに対してのみ，元の 16×16 のピクセルに展開して干渉の有無を調べればよいので，干渉を調べるべきピクセルの個数の削減が可能である．

仮想物体と計測点群が近接していない場合は，粗いデプスマップのみで判定可能であることが多い．　それ以外の場合には，詳細なデプスマップを使用して干渉判定を行なう．その場合でも，粗いデプスマップにおいて「干渉あり」の三角形に対してのみ詳細デプスマップでの干渉を調べればよいので，高速な判定が可能である．

本論文で提案した手法は非常に高速であり，5000 万点程度の点群であれば，仮想物体をマウスでドラッグする速度に追従して干渉判定を行なうことができる．干渉判定結果の例を**図 12** に示す．

## 5.　評価実験

### 5.1 評価条件

計算機は 16GB のメモリをもつ 64bitPC を用いた．CPU のクロック数は 2.80GB である．本手法の評価のために，大学

の加工場を計測したデータを用いた．計測データを**図 13** に示す．計測点群は 4359 万点 で構成されている．使用したデプスマップの最大解像度は縦 5760，横 11520 とした．この最大解像度は，計測で得られた大規模点群をすべて保持したものである．点群のピッチは，計測原点から 5m の距離 3.1m，10m の距離で 6.3mm であるため．点間距離が大きいことによる干渉の見逃しが生じにくく，高解像度での干渉判定が可能である．評価実験では，計測原点からの距離が 5m 以内の点を 1 万箇所選択し，その位置に仮想物体を置いたときの干渉判定を行った．

### 5.2 デプスマップの階層数による計算速度の評価

本研究で提案した手法を評価するために，複数の階層デプスマップを用いて，計算時間を評価した．評価においては，16×16 ピクセルを 1 ピクセルに圧縮した粗いデプスマップを用いた．また，比較のために，4×4 ピクセルを 1 ピクセルに圧縮した中間的な解像度のデプスマップも用意した．これらを用いて，**表 1** に示すように，詳細デプスマップのみを用いた干渉判定，詳細デプスマップと粗いデプスマップによる干渉判定，3 階層のデプスマップを用いた干渉判定の計算時間を評価した．3 階層デプスマップの干渉判定では，干渉の可能性があるときに，順次，解像度の高いデプスマップを使用する．

計算時間を**表 2** に示す．ここでは，1 万箇所で干渉判定を行なった場合の平均を示している．詳細デプスマップのみを用いた場合では，非常に多くのピクセルを調べる必要があるため，他に比べて処理が遅くなる．特に，「干渉なし」「干渉不明」の場合に処理速度が低下する．この 2 つの判定の場合，仮想物体が投影された全ピクセルで干渉判定を行なっているためである．我々の提案した 2 階層デプスマップを用いた場合には，計算が大幅に高速化され，単一デプスマップの場合の約 22 倍となった．

一方で，3 階層デプスマップでは，2 階層の場合よりも処理速度が遅くなった．干渉する場合には最大解像度でのデプスマップが必要となるため，中間解像度での処理によって干渉判定できるケースが少なかったためと考えられる．

以上の結果から，我々の提案した詳細デプスマップと粗いデプスマップの 2 階層を用いた手法によって，干渉判定の速度が著しく向上できることがわかった．

**Table.1** Three cases of depth maps

| | Low | Middle | High |
|---|---|---|---|
| 1 Layer | - | - | 5760×11520 |
| 2 Layers | 360×720 | - | 5760×11520 |
| 3 Layers | 360×720 | 1440×2880 | 5760×11520 |

**Table.2** Performance evaluation (frame per second)

| | No Collision | Collision | Unknown | Average |
|---|---|---|---|---|
| 1 Layer | 25 | 34 | 26 | 26 |
| 2 Layers | 578 | 548 | 544 | 562 |
| 3 Layers | 289 | 284 | 287 | 287 |

### 5.3 仮想物体のメッシュ数の影響評価

次に，仮想物体のメッシュ数を変えて，手法の評価を行った．ここでは，仮想物体として，1 辺の長さが 50cm の立方体を用いた．**表 3** に計算時間を示す．この結果から，メッシュ数を多くすると，デプスマップに投影する面の個数が増加することが示されたが，処理時間の増大は緩やかであり，三角形の個数が 2 万程度であれば，リアルタイム処理が可能であることがわかる．

次に，各メッシュ数において，1 万箇所での処理速度のワーストケースを**表 4** に示す．ワーストケースは，粗いデプスマップでは干渉の有無が判定できず，詳細デプスマップにおいて「干渉あり」「干渉なし」「干渉不明」が判明した場合である．「干渉あり」が最も速い理由は，1 つのピクセルでも干渉があれば処理が終了するためである．表 4 の結果から，この評価実験においては，ワーストケースにおいてもリアルタイム性が確保されていることがわかる．

**Table.3** Performance evaluation (frame per second)

| Triangles | No Collision | Collision | Unknown | Average |
|---|---|---|---|---|
| 4800 | 245 | 247 | 247 | 246 |
| 10800 | 153 | 158 | 153 | 156 |
| 14700 | 125 | 126 | 129 | 127 |
| 19200 | 102 | 103 | 104 | 103 |

**Table.4** Worst cases of processing time (frame per second)

| Triangles | No Collision | Collision | Unknown |
|---|---|---|---|
| 4800 | 147 | 154 | 138 |
| 10800 | 93 | 112 | 94 |
| 14700 | 73 | 87 | 90 |
| 19200 | 60 | 63 | 62 |

**Table.5** Performance evaluation (frame per second)

| Cube size | No Collision | Collision | Unknown | Average |
|---|---|---|---|---|
| 25cm | 254 | 256 | 259 | 256 |
| 50cm | 245 | 247 | 247 | 246 |
| 75cm | 221 | 231 | 229 | 227 |
| 100cm | 213 | 229 | 216 | 222 |

### 5.4 仮想物体の大きさの影響評価

次に，様々な大きさの立方体での計算時間を評価した．メッシュ数は 4800 に固定した．面が大きくなると，三角形をデプスマップに投影したときに補間するピクセル数が増大する．評価結果を**表 5** に示す．この結果から，ピクセルの補間処理に要する時間は相対的に小さく，干渉判定の計算時間に対する影響が小さいことがわかる．

## 6. 結　　言

本研究では，大規模点群と仮想物体との干渉判定を高速かつ高解像度で行なう手法を提案した．提案手法では，大規模点群から詳細デプスマップと粗いデプスマップを生成し，仮想物体をデプスマップ上に投影することで，「干渉なし」「干渉あり」「干渉不明」の 3 通りの判定を行なう．また，実データを用いた評価実験の結果，本手法を適用することで非常に高速な干渉判定が実現できることを示した．

ただし，提案手法では，単一のスキャナ位置からの計測で得られた点群を対象としている．複数地点で計測された点群を用いることで「干渉不明」の場合が減少すると考えられるが，複数のデプスマップを保持するとデータ量が大きくなるという問題がある．今後は，複数視点からの計測点群による干渉判定を実用的なメモリサイズで効率的に行なう手法について検討していく予定である．また，工業設備でのレイアウト検討などの手法についても考えていく予定である．

## 参 考 文 献


1) M. Figueiredo, J. Oliveira, B. Araújo, J. Pereira: An efficient collision detection algorithm for point cloud models, 20th International conference on Computer Graphics and Vision, 43 (2010) 44
2) J. Pan, I. A. Sucan, S. Chitta, D. Manocha: Real-time collision detection and distance computation on point cloud sensor data, In Robotics and Automation (ICRA), 2013 IEEE International Conference on, (2013) 3593-3599
3) J. Klein, G. Zachmann: Point cloud collision detection, Computer Graphics Forum, 23 (2004) 567-576
4) M. Radwan, S. Ohrhallinger, M. Wimmer: Efficient collision detection while rendering dynamic point clouds, In Proceeding of 2014 Graphics Interface Conference, Canadian Information Processing Society, (2014) 25-33
5) 平岡美那子，増田宏: 大規模点群における衝突判定法，精密工学会春季大会 学術講演会講演論文集，(2013) M67
6) H. Masuda, I. Tanaka, M. Enomoto: Reliable surface extraction from point-clouds using scanner-dependent parameters, Computer-Aided Design and Applications, 10(2) (2013) 265-277