大規模点群に基づく干渉判定システム

電気通信大学　○丹羽　健，増田　宏

Collision Detection System based on Large-Scale Point-Clouds

The University of Electro-Communications: Takeru NIWA, Hiroshi MASUDA

Recently, the progress of 3D scanners enables to capture large-scale point-clouds in short time. It is useful for layout planning to simulate collision detection between point-based environment and 3D models. In this research we propose high-precision and real-time collision detection system based on large-scale point-clouds. Collision detection is calculated using depth-maps, which are generated from point-clouds. In this paper, we consider a method to use multiple point-clouds.

Key words: Point-Cloud, Collision Detection,

1.　緒言

近年，3D レーザスキャン技術の進歩により，短時間で高密度かつ高精度な計測点群データを得られるようになった．この計測データを用いることによりコンピュータ上に３次元の大規模仮想環境を構築することができ，多領域での応用が考えられる．シミュレーションや VR 等に応用するためには，点群で表現された仮想環境と仮想物体間での干渉判定が求められることが多い．しかし，大規模環境の計測データは数千から数億点におよび，一般の PC でリアルタイムに処理することは容易ではない．

そこで本研究では，大規模点群に基づいた高精度かつリアルタイムな干渉判定システムの開発を目的とする．さらに，搬入や経路探索といったシミュレーションのためのアルゴリズムを研究する．

前報[1]では，デプスマップを使用した干渉判定アルゴリズムを提案した．しかし，前報では，システムとして求められる判定精度，また，複数の計測データを使用した際の問題について論じられていなかった．そこで本報では，判定精度と処理速度について評価を行い，また，複数データ上での干渉判定手法に関しても論じる．

2.　干渉判定アルゴリズム

2.1　デプスマップ上での干渉判定法

干渉判定は以下のように行う．計測した環境点群を２次元マップ上に展開し，各ピクセルが計測原点との距離を持つデプスマップに変換する．次に，メッシュで表現された仮想物体 X の頂点{P}をデプスマップ上に投影し，さらに，各メッシュの頂点のデプス値からメッシュ内のデプス値を補間する．閉じたモデルは同じピクセル上に少なくとも計測原点から見て表側のデプス値 d_1，裏側のデプス値 d_2 の 2 つが投影される．デプスマップの投影箇所のデプス値を r とすると，d_1，d_2，r の大小関係によってそのピクセルでの干渉の有無を判定することができる．判定の方法を以下に示す．

(1)　$d_2 > r$：干渉なし

(2)　$d_1 \leq r \leq d_2$：干渉あり

(3)　$r < d_1$：干渉不明

以上の判定を物体 X の投影範囲上のすべてのピクセルについて行うことにより，点群との干渉判定が行われる．

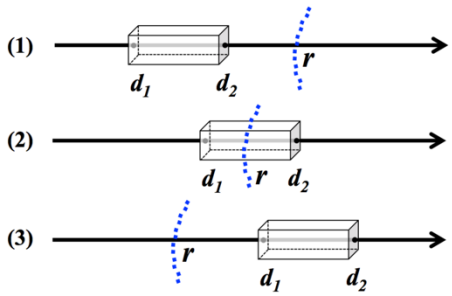

図 1 デプスマップ上での干渉判定法

2.2　干渉判定精度と処理速度の評価

干渉判定法をシステムとして利用するために，精度とリアルタイム性について評価を行った．仮想物体(面:2 千,頂点:1 千)と大学の加工場を計測した 4,000 万の大規模点群を用いて空間上の 1 万箇所について干渉判定処理を行い評価実験とした．デプスマップの解像度は 11,520×5,760 とした．この場合のピクセル間隔は計測原点から 5m 距離で 2.73mm である．

結果を表 1 に示す．評価は，Intel Core i7 2.80 GHz の CPU，16GB の RAM を持つ PC にて行った．干渉ありが最も速くなった理由は，１つでも干渉のあるピクセルがあった場合，それ以降の判定を打ち切るためである．計算時間はどの判定結果においてもリアルタイムなものであり，十分な処理速度をもつアルゴリズムと評価できる．

表 1　判定精度とフレーム数

5m 距離での判定精度	１秒あたりのフレーム数		
	干渉なし	干渉あり	干渉不明
2.73(mm)	140.40 (fps)	173.96 (fps)	167.66(fps)

3.　干渉判定の高速化と判定率向上

3.1　仮想物体の境界ボックスを使用した高速化

仮想物体のデプス値の計算時間はメッシュ内のデプス値補間による部分が大きく，面の数によって変化する．しかし，仮想物体の詳細な凹凸形状を考慮せずとも，判定結果が明らかな場合は多い．そこで仮想物体の Oriented Bounding Box[2]をデプスマップに投影し，詳細な判定の必要性を事前に確認することで，計算時間を削減する．

仮想物体の OBB の構築は以下のように行う．生産設備は平面で表される部分が多く，特に地面のように水平な平面は大きい．

そこで，Z 方向の境界は仮想物体の Z 成分の最小値と最大値によって決定し，残り 2 つの境界は X,Y 成分の主成分分析によって計算する．このようにして構築した OBB を図 2 に示す．

OBB をメッシュモデルとして表現し，同様の手法でデプスマップに投影する．そして，OBB の判定結果が「干渉あり」となった場合にのみ，本来のメッシュモデルを使用して干渉判定を行う．これにより，環境点群と仮想物体が大きく離れていた場合の計算時間が削減される．

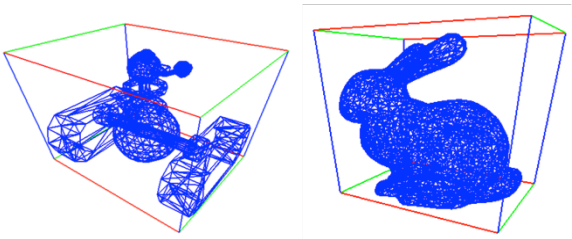

図 2 メッシュモデルに本手法を適用した OBB

3.2 複数の計測点群データの使用

1 つの計測点からのデータではオクルージョン領域が多く，正確な判定ができず「干渉不明」となってしまうことが多い．オクルージョン領域を減らすために，複数視点からの計測データを使用する．仮想物体の位置はワールド座標系で表し，各デプスマップはワールド座標系から計測原点座標系への変換行列 M を持つ．そして，各デプスマップ上での干渉判定のために，仮想物体の座標を変換行列 M で各計測座標系へと変換する．また，判定を行うデプスマップの順番は，水平面の二次元座標での仮想物体と各計測原点との距離が近い順とした．各マップでの判定では，「干渉あり」の場合は以降のマップでの判定を打ち切る，「干渉なし」と「干渉不明」の場合は次のマップでの判定も行う．「干渉あり」の判定がなかったものに対しては，1 つでも「干渉なし」がある場合，これを判定結果とする．

3.3 仮想物体の凸形状分割

複数の計測点群データを使用することによって，「干渉不明」となる領域を減らすことができる．しかし，図 3 左図では総合的には干渉が無いにもかかわらず，計測原点 1,2 のいずれからも仮想物体の一部が隠れるために「干渉不明」となってしまう．また，処理速度の高速化の為に，仮想物体のピクセルは投影されるデプス値のうち最遠と最近の値を持つこととしている．このとき，図 3 右図のように凹のある仮想物体では，計測原点を考慮すると橙色で表された形状として処理されてしまい，誤って「干渉あり」と判定されてしまう．

そこで，仮想物体を凸形状に分割して，各々に干渉判定を行うことで判定可能領域を増やすことを考える．分割される形状

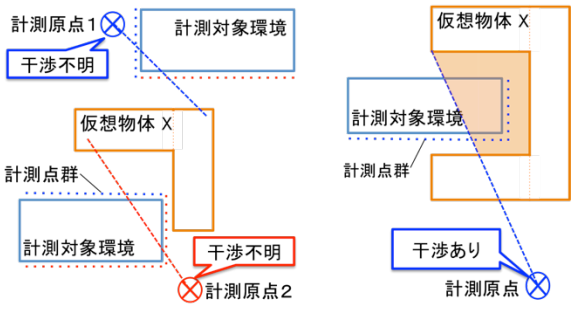

図 3 オクルージョン領域と仮想物体の関係

が凸形状になるようにすることによって，デプスマップに投影されるメッシュ中でのデプス値の最遠と最近のみで判定することが可能となる．また，仮想物体を細分化することで判定性能を高めることができる．

3.4 提案手法の評価

仮想物体(面:1 千，頂点:2 千)と大学の加工場を 4 箇所から計測した各 4,000 万の大規模点群を使用し，2.2 節の実験と同様の方法にて処理を行い，処理速度の評価を行った．1 計測のデプスマップあたりの RAM 使用量は 265MB である．

結果を表 2 に示す．OBB のみで判定した場合の処理速度は表 1 での通常の処理速度よりも速い．また OBB のみで判定可能なケースは 83%であったことから，計算時間の削減が確認できる．また，複数計測点群を使用した場合でも，リアルタイムな干渉判定となっている．

表 2　OBB の使用率とフレーム数

	干渉なし	干渉あり	干渉不明
OBB のみ	110.66 (fps)	なし	30.97 (fps)
通常判定	75.55 (fps)	54.10 (fps)	28.86 (fps)
平均	107.12 (fps)	54.10 (fps)	30.81 (fps)

次に，仮想物体の凸形状分割によって「干渉不明」から「干渉なし」に変化した判定結果を図 4 右図に示す．各計測原点と仮想物体の配置は図 4 左図のようになっている．ロボットの各クローラ部を別の仮想物体として分割し，各々に干渉判定を行うことによって「干渉なし」という結果が得られた．

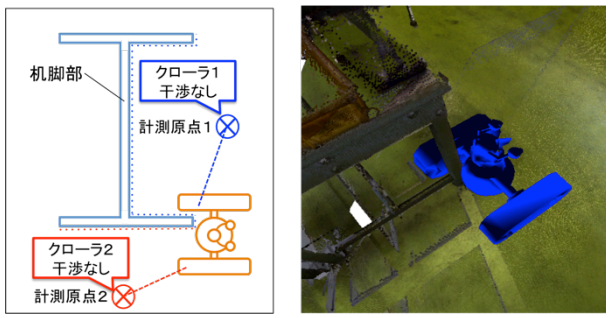

図 4 凸形状分割による判定の変化

4. 結論

本研究では，提案したアルゴリズムが大規模点群に基づく干渉判定システムに必要な精度と処理速度を持つことを示した．OBB を使用することによって干渉判定の処理時間を削減することができた．複数計測点群データを使用することによって干渉判定可能領域を増やす手法を提案した． 仮想物体を凸形状に分割することによって判定性能を高める手法を提案した．

今後は，干渉判定システムを実際のユーザが使用することを想定し，近接性の判定や，改修・搬入シミュレーションといった機能を開発していきたい．

参　考　文　献

1) 丹羽健，増田宏：大規模点群における衝突判定法（第 3 報），精密工学会春季講演会 2014
2) 丹羽健，増田宏：大規模点群における衝突判定法（第 2 報），精密工学会春季講演会 2013
3) S.Gottshalk， M.C.Lin and D.Manocha：OBBTree: A Hierarchical Structure for Rapid Interference Detection，Proceedings of ACM Siggraph '96