グラフィクスとCAD 97－2
（1999．12．10）

仮想空間における衝突検出およびオブジェクト探知の一技法

山地秀美　新藤義昭
日本工業大学

仮想空間におけるオブジェクト間の，障害物探知，衝突検出，目標物捕捉等を行う一技法を提案する。障害物を検出するために、一方は移動するオブジェクトから移動方向、他方は逆方向の 2 つの距離画像を、直交投影により作成する。距離画像は、陰面消去のために作成されるデプスバッファから取得する。2 つの距離画像から、移動するオブジェクトと、他のオブジェクトとの間の距離画像を作成する。これを解析する事で、障害物を探知することができる。この技法は、大域的な管理情報を必要とせず、1 回の描画程度の計算コストと記憶量で障害物検出ができる。また、既存の描画処理機能を利用できるので、汎用性が高い。

A Technique of Collision Detection and Object Search in Virtual Reality Space

Hidemi Yamachi, Yoshiaki Shindo
Nippon Institute of Technology

We propose a new technique to detect obstacles, collisions, and to capture objects in the 3-D virtual space named “Cyber Radar”. To detect other objects, this technique uses a Depth-Distance-Array that is generated by rendering process normally. It uses two Distance-Arrays projected by orthogonal view volume, one is from the object toward the moving direction (Range Distance Scope), other is reverse direction to get the object shape (Mask Distance Scope). After that, "LookAt Distance Scope" is calculated from two Distance-Arrays, which has distances between the object and others. The computation cost of this technique would be very low compared with other collision-detection algorithm that costs less than one time rendering process.

1．　はじめに

バーチャルリアリティ（ＶＲとする）における衝突検出およびオブジェクト探知は、ＶＲを構成する上で基本的であるにもかかわらず、十分な精度を保ちつつ、リアルタイムに処理することが困難な課題となっている。

代表的な衝突検出のアルゴリズムとして、OBB(Oriented Bounding Boxes)、AABB(Axis-Aligned Bounding-Boxes)、および Bounding-Spheres などが挙げられる[1]～[6]。これらの手法の中核をなすアルゴリズムは、以下のような共通点を持つ。

- オブジェクトを囲む凸包を定義する。凸包は、全体を含むものから、部分を含むものへ、詳細化して定義され、階層木で管理される。
- オブジェクトの移動に伴い、衝突検出を行うオブジェクト対を決定する。
- 対象となるオブジェクトの階層木の上位から、凸包の重複を検出する。
- 重複が検出されたら、下位の重複する凸包を検出する。
- オブジェクトの動きに合わせ、凸包および階層木を再構成する必要がある。

こうした手法は、微細な衝突を検出できる反面、凸包および階層木作成のための計算コストが非常に大きい。また、オブジェクトそ

" A Technique of Collision Detection and Object Search in Virtual Reality Space "
Hidemi Yamachi : yamachi@nit.ac.jp
Yoshiaki Shindo: shindo@nit.ac.jp
Nippon Institute of Technology

のものの形状が変化する場合は、管理情報を再構成しなければならない。

衝突検出のための計算コストは、インタラクティブなＶＲにおいて要求されるリアルタイム性を実現する上で、大きな障害となっている。

これらのアルゴリズムのさまざまな改良が提案されているが、問題の原因はアルゴリズムそのものの複雑さにあるため、パフォーマンスの大幅な改善は困難である。

本論文で提案する手法「サイバーレーダ(Cyber Radar)」は、次の点を目標とする。

①複雑な形状同士の衝突判定を実時間で行う

②オブジェクトの動きに無関係に、衝突、探知ができる

③形状の記述情報に依存しない方法で行う

サイバーレーダは、描画時に作成される陰面消去用のデプスバッファの距離画像を利用し、オブジェクトの検出を行う。

オブジェクトが移動するとき、オブジェクトの位置から移動方向に描画を行い、距離画像を作成する。距離画像の解析から、他のオブジェクトの有無、距離、衝突時の位置および衝突面の法線を検出することができる。

本手法は、オブジェクトの形状にほとんど影響されない。また、周囲に存在するオブジェクトの数や、それらが動くかどうかにも依存しない。

さらに、計算コストは、移動オブジェクトの数Ｎに対し、Ｏ(Ｎ)程度となる。

また、多くの場合、既存のグラフィックスAPI およびハードウェアの機能を利用するため、低コストで実装することができる。

本論文では、２章および３章でサイバーレーダによるオブジェクト探知、衝突検出の手法を提案する。４章では、サイバーレーダの実装と評価を示す。５章でサイバーレーダの問題点および課題を述べる。

２．サイバーレーダによるオブジェクト探知

2.1 オブジェクト探知の手順

従来の手法は、空間分割を繰り返してオブジェクトの存在する領域を特定するのに対し、サイバーレーダは、対象となる空間を直接「見る」ことにより検出するという特徴を持つ。

本手法では、まず判定する領域の一端から、領域を包含するビューボリュームを定義して描画を行なう。この時作成されるデプスバッファから距離画像（**探査距離画像 Range Distance**）を取得する。距離画像に最大値未満の点があれば、そこにオブジェクトが存在すると判断できる。

以下に、その手順を示す（Fig.1)。

(1) 検出の起点位置（視点）に移動する

(2) ビューボリュームの解像度、幅、高さ、奥行き、直行・透視投影の別を定義する

(3) 描画する

(4) 探査距離画像を取得する

(5) 探査距離画像を解析し、奥行き値以下のピクセルの有無を検出する

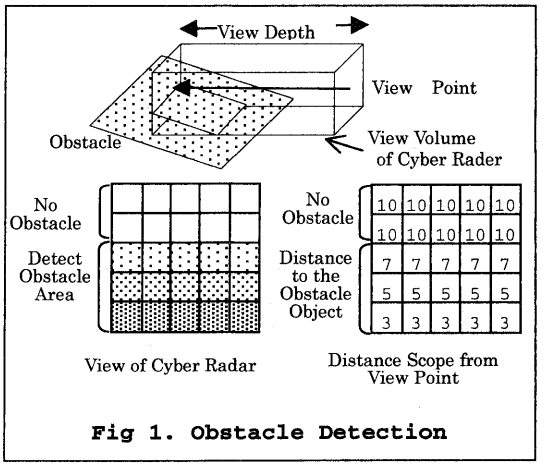


Fig 1. Obstacle Detection

2.2 ビューボリュームの定義

ビューボリュームの解像度は、描画時間に大きく影響する。そのため、パフォーマンスを向上させるには、可能な限り低解像度で描画する必要があが、解像度以下のオブジェク

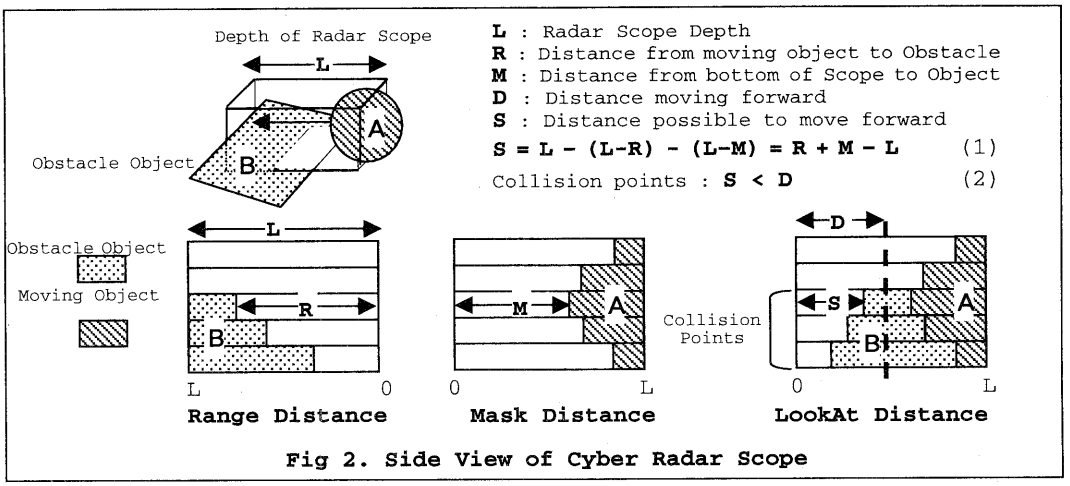


Fig 2. Side View of Cyber Radar Scope

トは認識できなくなる。

そこで、検出対象の状態によって、解像度を動的に変化させることが重要になる。

最初は低解像度で奥行きの長いビューボリュームで検出を行う。オブジェクトが検出されたら、その距離と大きさに応じて、最適なビューボリュームを定義する。

探査距離画像を解析する際、直行投影法では、補正なしにその値を使用することができるのに対し、透視投影法では、位置情報の補正が必要になる。また、検出起点（視点）をビューボリュームの近接面上に設定できないため、検出起点近傍に死角ができる。そのため、直行投影法のみを使用する。

2.3 探査距離画像の解析

取得された探査距離画像は、ビューボリュームの奥行きを最大値（L）とする、0 以上の値を取る。1 画素の左右の大きさは、設定されたビューボリュームのサイズを解像度で割った値となる。

L 未満の値のピクセルを検出することで、オブジェクトの存在を判定できるだけでなく、オブジェクトまでの距離、ビューボリューム内の座標系に対する位置情報が得られる。

ただし、存在するオブジェクトの数は判明しない。

3．サイバーレーダによる衝突処理

3.1 距離画像の作成

衝突処理には、衝突の検出、衝突位置の特定、衝突後の運動定義の 3 つのフェーズがある。いずれの場合も、衝突するオブジェクト相互の衝突面の情報が不可欠となる。

そこで、以下のように距離画像の作成を行い、衝突面の情報を取得する(Fig.2)。

(1)オブジェクト（A）の位置から移動方向に、直交投影法で移動量に対応する奥行きのビューボリュームを作り、オブジェクト以外（B）を描画した探査距離画像を作る。

(2)移動先の位置からAの方向を向いたビューボリュームを作り、Aだけを描画した距離画像を取得する（**マスク距離画像 Mask Distance Scope**）。

(3)探査距離画像とマスク距離画像から、移動可能な範囲を求める（**照準距離画像 LookAt Distance Scope**）。

(4)照準距離画像から衝突位置（要求された移動量未満の点）を求め、衝突判定を行う。

ビューボリュームは、Aに外接する幅と高さ、移動によってオブジェクトが掃き出す空間を含む奥行きを持つように定義する。

マスク距離画像の内、奥行きと等しい値を持つ点は、何もない領域なので、衝突判定から除外しなければならない。これによって、運動方向に直交する方向の凹凸への対応が可能になる。

照準距離画像は、ＡＢ間の距離をあらわし、探査距離画像とマスク距離画像から、Fig.2の式(1)によって求められる値Ｓから作成される。移動しようとする距離Ｄに対し、Ｓ＜Ｄとなる点があれば、衝突が発生する。このうちの最小値が、移動可能な距離を表わす。

照準距離画像により、ＡＢ両方の運動方向の凹凸を正しく処理することができる。

3.2 衝突位置と衝突面の法線の推定

衝突の中心位置を推定するには、距離画像の画素の最小値と等しい画素（衝突点）を調べる。衝突点の数は、1点、2点、3点以上の、3つの場合に区分される。

1点の場合は、その点の近傍画素の距離情報から、法線を推定する。

2点の場合は、2点の中点を衝突中心位置とし、2点を結ぶ直線と、いずれかの点の近傍画素で構成される三角形から、法線を求める。

3点以上の場合は、全ての点を含む、外接多角形を求め、その重心を衝突中心とする。法線は、任意の3点を利用して求めることができる（Fig.3）。

衝突位置および法線ベクトルの精度は、距離画像の解像度に依存する。低解像度で衝突を検出し、衝突位置に対してサイズや解像度を変化させて、高精度の解析を行うこともできる。

3.3 マスク距離画像とビューボリュームサイズの動的設定

運動方向から見たオブジェクトの形状が変化しない場合は、マスク距離画像は更新する必要がない。球の場合は、初期処理でマスク距離画像を作成し、繰り返し利用することができる。しかし、変化する場合は、変化に応じてマスク距離画像を更新しなければならない。

ビューボリュームのサイズも、同様に変化させる必要がある。

オブジェクトに外接する直方体を正確に定義するコストを回避するために、距離画像を利用して、以下のように処理する。

(1) オブジェクト定義時に、オブジェクトの最大長（D）を定義する。

(2) 運動方向から見たオブジェクトの形状が変化するとき、ビューボリュームの幅と高さをDとし、マスク距離画像を作る。

(3) マスク距離画像から、上下左右のオブジェクトの存在しない領域の幅を調べ、ビューボリュームの幅と高さに設定する。

(4) マスク距離画像を作成し直す。

実験では、オブジェクトの最大長は、XYZ方向の長さから求めた対角線の長さを設定し

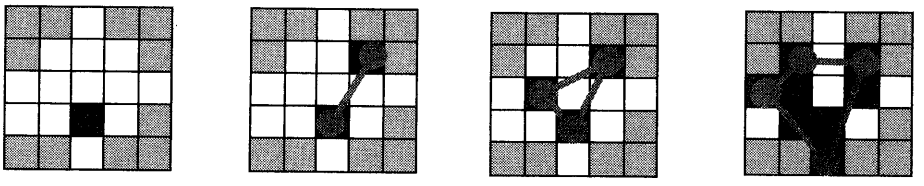
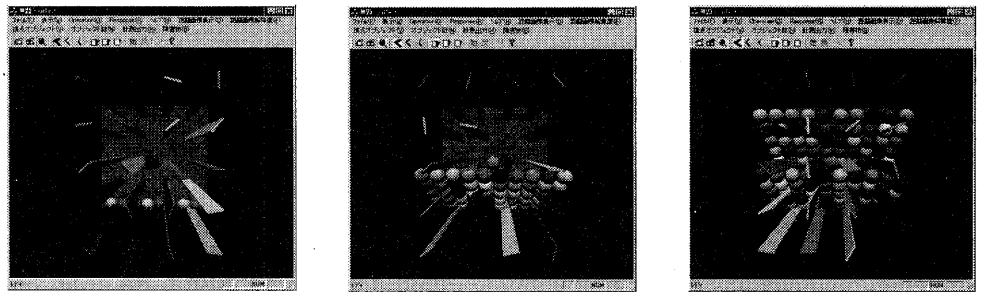


Fig.3 Estimation of collision range



10 Move Objects　　50 Move Objects　　100 Move Objects

20 Plate Obstacles and Move Spherical Objects in Cube

WindowsNT4.0(SP5) PentiumⅢ450MHz

Fig.4 Estimation of Performance

ている。

これにより、複数のオブジェクトの接合や分離が発生しても、適切なビューボリュームのサイズを得ることができる。

4．サイバーレーダの実装と評価

4.1 実行時間の計測

サイバーレーダは、描画することによりオブジェクトを検出するため、計算コストは1フレームの描画コストに依存するが、通常の描画とは異なり、より低解像度で、照光計算も不要となる。また、ビューボリュームに含まれるオブジェクト数も少なくなるため、通常の描画より低コストで済む。衝突検出を行ないながら移動する複数の球を作り、実行時間の計測を行なった（Fig.4）。

ここでは、1つのマスク距離画像を1度だけ作成し、全てのオブジェクトで共用している。また、解像度は、10×10で計測した。

計測結果は、表1にある通り、衝突検出時間は、衝突検出を行なうオブジェクトの数に比例することがわかる。

4.2 解像度による実行時間の変化

距離画像の解像度による実行時間の変化を、Fig.4と同様の条件で調べたものが、表2である。解像度は、10×10、25×25、50×50の3種類で計測した。

解像度による実行時間は、画素数Nに対しO(N)より小さいことがわかる。

サイバーレーダは、描画機能を直接利用すため、実行時間は、ジオメトリエンジンをはじめとする、ハードウェアのグラフィックス機能により、大きく変化することが予想される。しかし、描画処理コストが、サイバーレーダの実質的な実行時間の大半を占めている。

	Number of Move Objects		
	10	*50*	*100*
No Detection	16.5	16.0	31.0
Perform Detection	66.6	266.5	702.3
Detection Time Per Object	6.7	5.3	7.0

Table.1 Performance Time of Collision Detection (msec)

	Number of Object		
Resolution	*10*	*50*	*100*
10x10	66.6	266.5	702.3
25x25	82.3	300.2	785.4
50x50	101.2	417.2	1035.3

Table.2 Performance Time Under Different Resolutions (msec)

4.3 実装例

サイバーレーダの有効性を確認するために、以下のような実装例を示す。

(1) サーチと接触

長距離のサイバーレーダを使い、オブジェクトが検出されるまで、サイバーレーダを回転させながら探査する。

サイバーレーダでオブジェクト描画するときに、オブジェクトに付与されたＩＤ番号を色として描画する。距離画像からオブジェクトが検出されたら、カラーバッファからその画素の色を取得し、オブジェクトを識別する。

(2) 木の成長モデル(Fig.5)

木の枝を成長させるときに、成長点にサイバーレーダを置き、成長する方向に他の枝が存在するかどうかを探査する。他の枝が存在するときは、樹相に応じた成長方向の選択を行い、再び探査する。成長可能な方向が見つかるまで、一定回数の試行を行うが、見つからない場合は成長を中止する。

Fig.5 Growth Model of Tree

(3) オブジェクトをつかむ

手のひらおよび、指のそれぞれの部分を構成する基本立体ごとにサイバーレーダを設定する。指の付け根にある基本立体の回転による移動量を、指先に向かって累積させながら、オブジェクトまでの回転を制御する。

5．サイバーレーダの課題

サイバーレーダには、以下のような問題が存在する。

(1) サーフェースモデルでは平面の厚さが無いため、視線方向に平行な面は検出できない。

(2) サイバーレーダによる衝突検出は、運動方向から見たマスク距離画像を利用するため、トーラスの穴のように、マスク距離画像に反映されない部分の衝突も検出できない。

(3) 回転時の衝突検出は、オブジェクトが掃き出す空間が円柱状になるため、直交投影によるビューボリュームから得られた情報をそのまま利用することはできない。

(4) フレーム間のオブジェクトの移動距離（速度）によって、衝突検出ができない場合がある。

今後、これらの問題に対する解決策を検討して行く。

参考文献

[1] "OBBTree: A Hierarchical Structure for Rapid Interference Detection", Gottschalk, S., Lin, M.C., Manocha, D., Proc. of ACM Siggraph '96, 173.

[2] Collision Detection between Geometric Models: A Survey M. Lin and S. Gottschalk. Appeared in the Proceedings of IMA Conference on Mathematics of Surfaces 1998.

[3] "I-COLLIDE: An Interactive and Exact Collision Detection System for Large-Scaled Environments" J. Cohen, M. Lin, D. Manocha and K. Ponamgi, Proceedings of ACM Int. 3D Graphics Conference, pp. 189-196, 1995

[4] "V-COLLIDE: Accelerated Collision Detection for VRML ", Hudson, M. Lin, J. Cohen, S. Gottschalk and D. Manocha. Appeared in Proc. of VRML'97

[5] "Collision Detection: Algorithms and Applications" Ming C.Lin, Dinesh Manocha, Jon Cohen, Stefan Gottscalk http://WWW.cs.unc.edu/~geom/collide.html

[6] "ストリーミングSIMD拡張命令による衝突検出バージョン1.1",インテル株式会社資料販売センター 1999年1月