計測自動制御学会東北支部 第231回研究集会 (2006.10.31)
資料番号 231-8

DirectXによる校舎内移動シミュレーションの開発

Development of the Moving Simulation in a School Building by DirectX

○大木崇嗣*，大久保重範*, 及川一美*, 高橋達也*

Takashi Oki*, Sigenori Okubo*, Kazumi Oikawa*, Tatsuya Takahashi*

*山形大学

*Yamagata University

キーワード： DirectX(DirectX)，衝突判定(Collision Judgment), インタラクティブアニメーション(Interactive Animation)

連絡先： 〒992-8510　山形県米沢市城南4-3-16 山形大学 工学部 機械システム工学科 大久保研究室
大久保重範，Tel.: (0238)26-3245, E-mail: sokubo@yz.yamagata-u.ac.jp

1. 緒言

近年の3D技術は目覚しい進化を遂げ、本物により近い表現が可能となった。この3Dを利用することで過去の建造物をリアルにディスプレイ上に表示し、ユーザーが自由に中を歩き回り、見渡すことが出来れば写真や動画では伝わらない臨場感が出せるのではないかと思われる。そこで本研究では、DirectXを用いて旧米沢高等工業学校の中を自由に歩きまわれるリアルインタラクティブアニメーションの製作を行い、リアルさを出すために人物と壁との衝突の表現を第一の目的とする。

2. DirectX

DirectXとは、Microsoft社が同社のWindowsシリーズのマルチメディア機能を強化するために提供している拡張API群の事であり、DirectGraphics、DirectMusic、DirectInput、DirectPlay、DirectShowの5つからなる。

3. 製作環境

校舎のモデリングには98年升澤大志氏製作の旧米沢高等工業学校(以下『旧校舎』)モデルを、人物にはDirectXのサンプル『Tiny』を使用する。[1] プログラムにはVisual Studio.NET、DirectXにはDirectX SDK(December 2004)を使用する。

4. 衝突判定

4.1 以前の判定方法

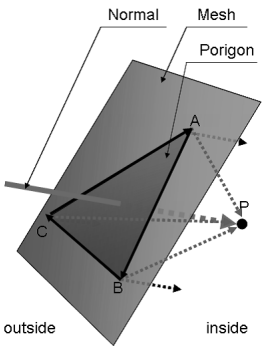

Fig.1:Ray Tracing.

これまで使用していた方法は、レイと呼ばれる線分を4方向に飛ばし、旧校舎モデルのポリゴンの表裏判定を行うことで衝突判定を行っていたが、この方法はモデルのポリゴンを1フレーム毎に全部調べているので大変非効率であると共に大変重い処理であった。

4.2 新しい判定方法

新しく考えた方法は、バウンディング球を使用した判定方法である。バウンディング球という球状のポリゴンを使用し、球体同士で判定を行う方法である。球体同士の交差判定は半径の大小だけで判断できるので高速な処理方法として利用されているが、球体でモデルを囲った場合、モデルの存在しない部分にも範囲が及んでしまい、判定が甘くなるという欠点がある。

その欠点を補うには球の半径を小さくし、モデルの形に沿って沢山配置することで解消できるのではないかと考えた。しかし、モデルの形毎に手作業で球を配置するのは大変なので、モデルの形に沿って自動で球を並べる方法を考えた。

5. 自動生成

モデルの形を調べる為に、まずモデルを囲む最小の大きさの直方体を作る。これは、モデルの右上と左下の座標から作成することが出来る。

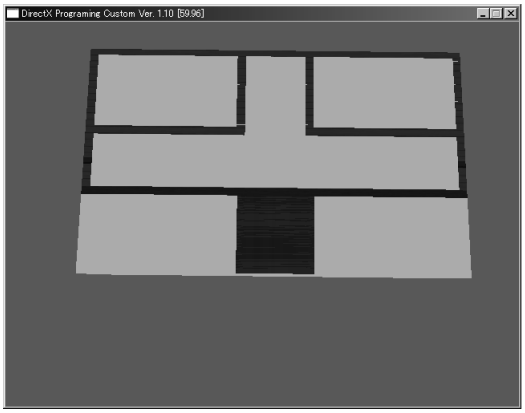

Fig.2:Create Bounding Box.

この直方体の縦と横の長さを、作りたい球の半径で割る事で球が半径分重なった状態で並んだ個数が割り出せるので、その数分だけ球を配置する。この状態からモデルの形に沿って球の座標を移動させるが、ここで以前の判定方法で使用していたレイを使用して形をトレースする。

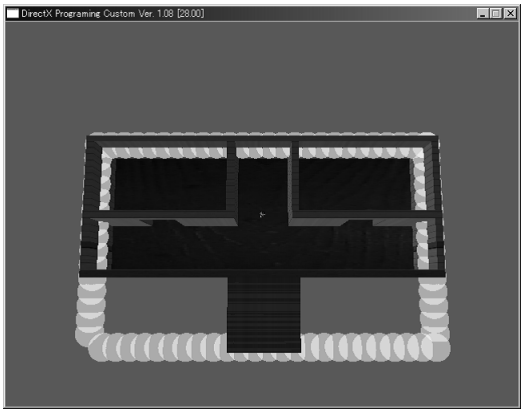

Fig.3:Setting Bounding Sphere.

球の中心から4方向にレイを飛ばし、レイがモデルと接触した場合、モデルと球との距離を測定し、その距離だけ球を移動させることでモデルに沿って配置することが出来る。それを全ての球に対して行なう事でFig.3の様にする事が出来た。

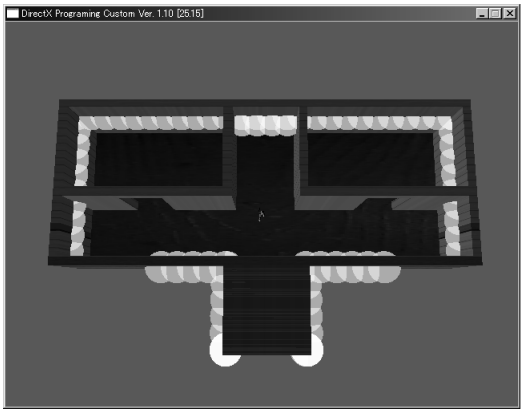

Fig.4:Moved Bounding Sphere.

現在は2方向以上レイがモデルと衝突を検出した場合、距離の長い方向に移動させている為、同じ場所に固まってしまうという現象が生じた。そこで、次に球体同士が重なっているかを調べる方法を考えた。

6. 3種類の検索方法

どの球体と重なっているかを調べる為に次に記す3種類の検索方法を実装し、それぞれの検索時間を比較した。[2]

6.1 総当り法

この方法は全部の球の座標を調べ、指定した範囲内に別の球の座標が存在した場合に検出とみなす方法である。しかし、調べる数は球の全体数の二乗オーダで増加していく為、数が多い場合には不向きである。

6.2 原点距離比較法

この方法は原点と球との距離を調べ、調べたい球の距離と近い距離の球だけを抜き出して検索する方法である。アルゴリズムは以下の通りである。

1) 球体数をn個で$O_i(i = 0 \cdots n)$とし、球の半径をrとする。
2) 原点からの距離D_iを測定する。
3) D_iを基準に、物体を昇順に並べ替える。
4) ある物体O_iについて以下の処理を行う。
   (a) $O_{i+h}(h = 1, 2, 3 \cdots)$との距離の差を$D_i$と$D_{i+h}$から求める。
   (b) 差がr以下なら衝突判定を行う。

6.3 XYZ法

3つ目の方法は、3軸毎に順次調べていく方法である。アルゴリズムは以下の通りである。

1) 球体数をn個で$O_i(i = 0 \cdots n)$とし、球の半径をrとする。
2) (球の中心座標-r)をS_{xyz}、(球の中心座標+r)をE_{xyz}を設定する。
3) 長さ2nの配列Xを用意し、そこにS_{xyz}、E_{xyz}を格納する。
4) 配列Xを昇順に並び替える。
5) X_jについて次の処理を行う。
   (a) $X_j = (S_x)_i$、$X_{j+1} = (E_x)_i$であればO_iは衝突判定のリストに加えない。
   (b) $X_j = (S_x)_i$、$X_{j+h} = (E_x)_i$の時、X_{j+s}に格納されている物体を判定リストに加える。$(0 < s < h)$
6) リストに加えられた物体だけで今度はZ軸について2の方法を使用する。
7) 残った物体のみで衝突判定を行う。

7. 測定結果

Table.1に生成個数と生成時間の平均を、Fig.4にXYZ法と原点比較法のグラフを記す。

Table.1:Processing time.

生成個数(個)	140	550	1832	5492
総当り法	9.77	199.3	2252.2	20237.2
原点比較法	0.27	1.20	3.60	11.10
XYZ法	0.33	1.33	3.77	11.30

単位(ms)

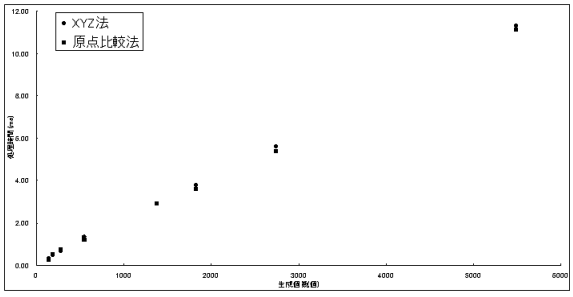

Fig.5:Compare Collision Judgment.

グラフの通り、XYZ法と原点比較法はほぼ同じ検索速度となった。しかし、次に示す検索の為の準備時間で差が現れた。

Table.2:Setting time.

生成個数(個)	140	550	1832	5492
原点比較法	141.5	2314.1	27411.6	263469.9
XYZ法	178.7	3058.0	34541.0	350150.4

単位(ms)

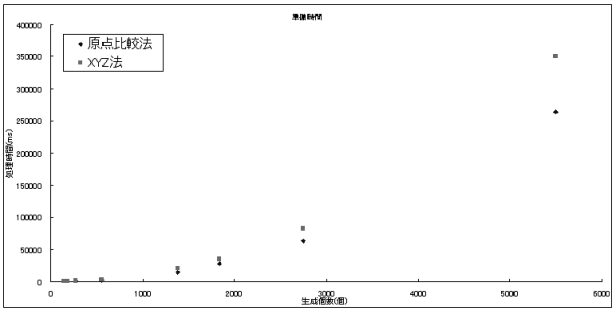

Fig.6:Compare Setting time.

準備時間は、球の数が多くなるに連れてXYZ法の時間が増え始めた。XYZ法は配列に球の全体数の2倍用意する必要がある為、数が増えるにつれて差が顕著に現れたのではないかと考えられる。この結果より、全体的に一番速度の速かった原点距離比較法を使用する事にした。

8. 現在の問題点

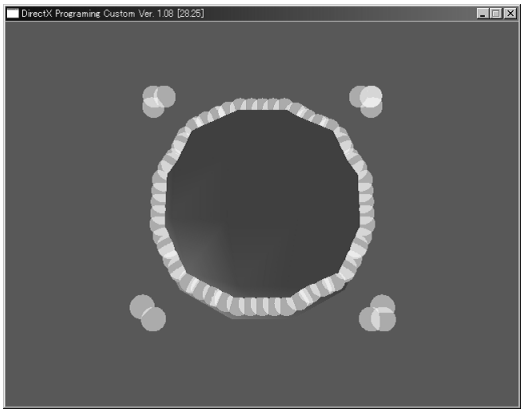

Fig.7:Alrady-known Ploblem(1).

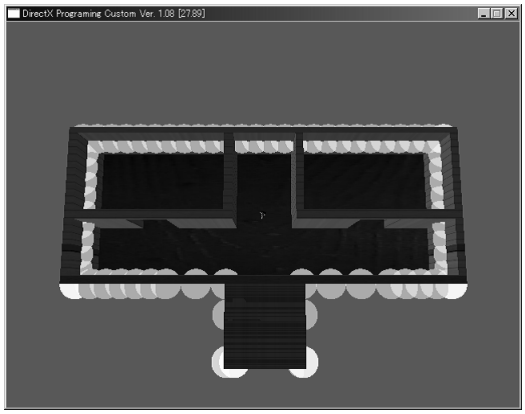

Fig.8:Alrady-known Ploblem(2).

Fig.7の様に、レイがモデルと触れない部分の球はその場から動かないようにしている為、無駄な球が出来てしまう。また、Fig.8の凸型モデルの場合、2方向以上壁を検出する場合は距離の遠い方へ移動させているのでそこの部分は球の密度が薄くなってしまう。今後は、レイがモデルと触れない球を再利用し、2方向以上検出した場合は片方に移動させるのではなく、球を一つ増やして両方へ移動させる等の方法を考えている。

9. 結言

今回は新しく実装した衝突判定の方法について述べた。今後は現在わかっている問題点の解消をしていきたい。

参考文献

1) 升澤大志: コンピュータグラフィックスに関する研究,山形大学大学院修士論文 (1998)

2) 大森祐:複数基準によるシリアライズを利用した衝突判定の高速化アルゴリズムに関する研究,東京工科大学卒業論文 (2002)