

複数物体による3次元アニメーションからの 4次元メッシュモデル生成システムに関する研究

北海道大学 ○齋藤 洋介, 小野里 雅彦, 田中 文基

要 旨

本研究では、3次元アニメーションから4次元メッシュモデルを生成するのに必要な、隣接フレーム間における複数物体を同一物体同士で対応付ける新手法の提案と、4次元メッシュモデル生成システムの連携によるシステムの高高速化を行う。

1. はじめに

これまで、時空間の構造を記述できる4次元メッシュモデルとそのモデリングシステム[1]が提案されている。4次元メッシュモデルの生成手法として、OpenGLの3DアニメーションからAPIフックによりOpenGLの描画命令を取得[2]し、取得した描画命令からボクセルモデル時系列データの生成[3]を介して4次元メッシュモデルを生成する手法[1] (図1)が提案されている。OpenGLからボクセルモデルを生成する際に、1つのボクセルモデルに複数の物体が存在したまま生成すると、複数物体の運動軌跡が1つの4次元メッシュモデルとして生成され干渉判定などができない。そのため個々の物体を分離し、分離した物体を同一物体同士で対応付けることにより、物体ごとのボクセルモデル時系列データを生成している。先行研究[3]による同一物体の対応付け手法は、物体の運動を考慮していないため対応付けに失敗することがある。本研究では、物体の運動と形状を用いた複数特徴量を過去フレームから推測し、現フレームの複数特徴量と比較した順位を重み付けによる総合順位から対応付ける手法を提案する。また、OpenGLの3Dアニメーションから4次元メッシュモデルを生成するためのボクセルモデル入出力を省くことによって3次元アニメーションから4次元メッシュモデル生成の高高速化を行う。

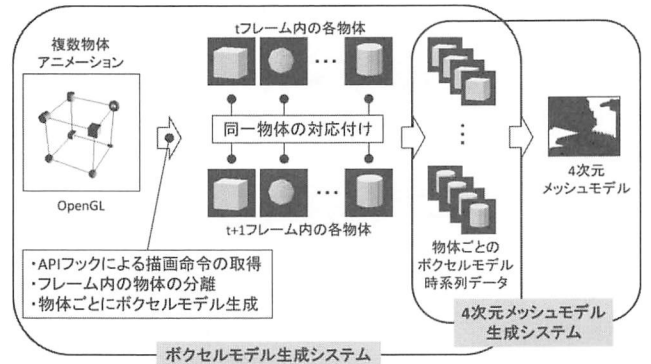


図1 OpenGLから4次元メッシュモデルの生成

2. 先行研究における対応付け手法の課題

先行研究[3]による対応付け手法は、隣接フレーム間において物体の位置変化、体積変化、頂点数の変化が閾値以下のとき対応付ける。そのため同一物体であるにも関わらずこれらの変化が閾値以上のとき対応付けに失敗する。本研究では、これらの失敗を無くすために、物体の運動と形状を用いてそれらを推測する手法を提案する。この結果、先行研究の閾値を満たさなくても対応付けが可能となる。

3. 物体の複数特徴量による同一物体の対応付け

3.1 提案手法の概要

提案手法の流れを図2に示す。過去フレームの物体の頂点を用いて頂点の平均を物体の重心の座標、主成分分析による固有ベクトルを物体の方向、固有値を物体の大きさとし、これらを物体の複数特徴量とする。固有値と固有ベクトルは3つの値と方向を持つので、物体の方向と大きさはそれぞれ3つ存在する。過去フレームの複数特徴量から現フレームの複数特徴量を推測し、推測した複数特徴量と現フレームの複数特徴量をそれぞれ比較し順位付ける。それぞれの順位を重み付けした総合順位から重複を避けて対応付ける。提案した手法の詳細と実装結果を以下に述べる。

3.2 物体の運動による重心の座標と位置推定

tフレーム目における物体iの頂点座標の平均値を物体の重心の座標 x_t^i とする。このときt-1フレーム目との重心の座標変化を物体の速度 \dot{x}_t^i 、速度変化を物体の加速度 \ddot{x}_t^i とする。物体の重心の座標と速度と加速度からt+1フレーム目における物体i

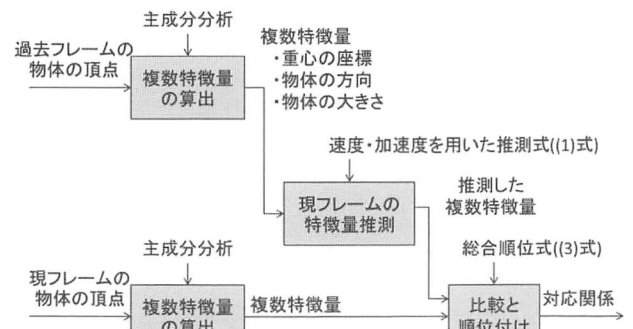


図2 本研究による同一物体の対応付け提案手法

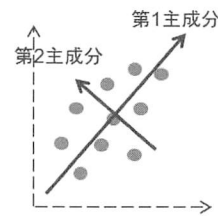


図3 主成分分析

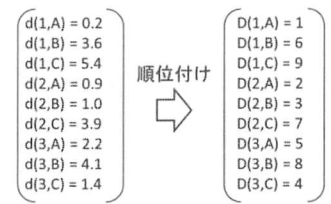


図4 順位付け

の重心の推定座標 \hat{x}_{t+1}^i を(1)式のテイラー展開より求める。

$$\hat{x}_{t+1}^i = x_t^i + \dot{x}_t^i + \frac{1}{2} \ddot{x}_t^i \quad (1)$$

t+1フレーム目における物体jの重心の座標 x_{t+1}^j と推定座標 \hat{x}_{t+1}^i の距離d(i,j)を求める。

3.3 主成分分析による物体の方向と大きさ

物体の頂点分布に対して、相関のある複数の変数を相関の少ない変数に変換する主成分分析を行う。変数の分散が最大となる軸から図3のように順に第1主成分、第2主成分、第3主成分と呼ぶ。頂点数mの物体の頂点座標を $(x_{1,i}, x_{2,i}, x_{3,i})$ ($i=1,2,\dots,m$)、頂点の平均を $(\bar{x}_1, \bar{x}_2, \bar{x}_3)$ とすると、これらの軸は、(2)式の共分散行列Cの固有ベクトルより求められる。

$$C = \begin{bmatrix} c_{11} & c_{21} & c_{31} \\ c_{12} & c_{22} & c_{32} \\ c_{13} & c_{23} & c_{33} \end{bmatrix} \quad (2)$$

$$c_{j,k} = \frac{1}{m} \sum_{i=1}^m (x_{j,i} - \bar{x}_j)(x_{k,i} - \bar{x}_k)$$

(2)式の固有ベクトルを複数特徴量である物体の方向とし、それぞれの固有値を物体の大きさとする。(1)式を用いて固有ベクトルと固有値を推測し比較する。物体*i*と物体*j*を比較する時、固有ベクトルは差ベクトルの大きさ $v_n(i, j)$ ($n=1,2,3$), 固有値は固有値の差 $s_n(i, j)$ とする。

3.4 順位付けと総合順位による対応付け

重心の座標の距離差 $d(i, j)$, 固有ベクトルの差ベクトルの大きさ $v_n(i, j)$, 固有値の差 $s_n(i, j)$ をそれぞれ値の小さい順に並べ、順位 $D(i, j)$, $V_n(i, j)$, $S_n(i, j)$ を決定する (図4)。各順位から重み w_n ($n=1,2,3$) を用いた総合順位 $E(i, j)$ を (3)式より求める。

$$E(i, j) = w_1 D(i, j) + w_2 \sum_{n=1}^3 V_n(i, j) + w_3 \sum_{n=1}^3 S_n(i, j) \quad (3)$$

$(w_1 : w_2 : w_3 = 3 : 1 : 1)$

重心の座標による順位 $D(i, j)$ と、物体の3方向による順位 $V_n(i, j)$ の平均と、物体の3つの大きさによる順位 $S_n(i, j)$ の平均とは同程度の重要性があると考えたので重みを $w_1 : w_2 : w_3 = 1 : 1/3 : 1/3$ つまり(3)式のようにした。同一物体の対応付けは総合順位の低い順位から順々に対応付ける。そのときすでに対応付けられた物体を対応付けの対象から除外することで、重複を避けた対応付けが可能である。

3.5 実装結果

使用したアニメーションを図5に示す。8つの物体はそれぞれ異なり、円柱(物体E, F, G)は大きさや頂点数が異なる。アニメーションに存在する線は空間を把握するために配置しているため、対応付けの対象としない。各物体の初期状態は一边12の空間の各頂点に配置し、空間の対角線方向に速さ $0.05\sqrt{2}t$ (t はフレーム数)の等加速度直線運動をしている。16フレーム目は物体が干渉している。先行研究と本研究による対応付けの結果を図6に示す。先行研究による対応関係を灰色の線、本研究による対応関係を黒色の線で表現している。先行研究による対応付けは、14フレーム目の物体EとGが15フレーム目の物体と対応付けされず14フレーム目で消滅と判断されている。また、15フレーム目と16フレーム目において物体AとB、物体CとDがそれぞれ対応付けに失敗している。一方、本研究による対応付けは正しい対応関係になった。今回使用したアニメーションの物体は速度と加速度の変化が微小な運動をしている。また、想定しているアニメーションやシミュレーションは速度と加速度の変化が微小な運動をしている。そのため本研究が提案する物体の運動を用いて推測する手法は有効な手法であると言える。しかし(3)式の重み付けは必ずしも最適な重みではない。この重みをより最適な値にすることでさらなる改善が見込まれる。

4 システムの連携による高速化

4.1 4次元メッシュモデル生成の現状

図1に示すように、ボクセルモデル生成システムと4次元メッシュモデル生成システムにより3次元アニメーションから4次元メッシュモデルを生成する。その際にボクセルモデルを入出力している。ボクセルモデルのデータサイズはボクセルモデルに比例し、ボクセルサイズが 512^3 の場合、データサイズは128(MB)である。4次元メッシュモデル生成のためのボクセルモデル時系列データの総データサイズはアニメーションのフレーム数に比例するため、膨大なデータ量となる。このような膨大なデータ量は扱いにくいいため本研究では、ボ

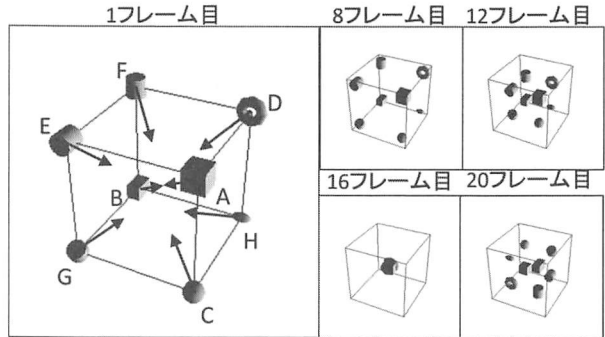


図5 使用したアニメーション

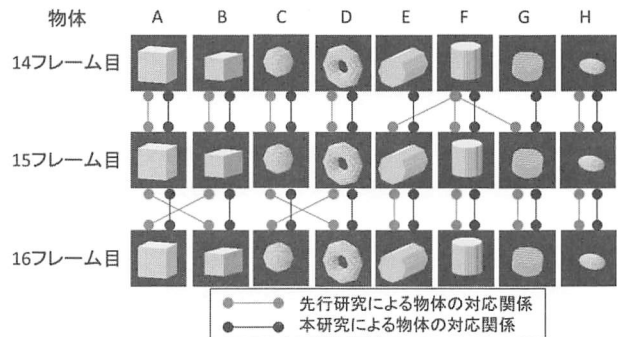


図6 対応付けの結果

クセルモデル生成システムと4次元メッシュモデル生成システムを連携することで、ボクセルモデルを入出力せず、3次元アニメーションから4次元メッシュモデルを生成する方法を提案する。

4.2 システムの連携結果と更なる高速化の実現方法

連携後のシステムはボクセルを生成しているが、それらをファイルとして入出力していない。そのためファイルの入出力の時間、システム連携前の約5%の時間だけOpenGLから4次元メッシュモデル生成の高速化を実現した。しかし、システム連携後のシステムにおいては、ボクセルから4次元メッシュモデル生成に時間を費やしている。そのためボクセルから4次元メッシュモデル生成の高速化によりシステムの更なる高速化が可能である。例えば、4次元メッシュモデル生成の際にボクセルの解像度分だけ繰り返し処理をしているが、これらの処理をGPGPUにより並列化することで高速化が可能である。

5. 結論

本研究では、OpenGLの3Dアニメーションから4次元メッシュモデルを生成するための隣接フレーム間における同一物体の対応付け手法として、物体の運動と形状を用いた新たな手法の提案と、4次元メッシュモデル生成システムの連携によるシステムの高速化を行った。

今後の課題として、GPGPU使用による4次元メッシュモデル生成の高速化が挙げられる。

参考文献

- [1] 川岸良次他：サイバーフィールドのための4次元形状モデリングに関する研究(第2報), 2008年度精密工学会秋季大会学術講演会講演論文集, J03, 2008
- [2] D Trebilco : GL Interceptor, <http://glinterceptor.nutty.org/>
- [3] 加藤航平他：サイバーフィールドのための4次元形状モデリングに関する研究(第4報), 2009年度精密工学会春季大会学術講演会講演論文集, O03, 2009