複雑な断裂を考慮した格子ベースの変形アニメーション

太田　充　　西田　友是
東京大学

1 はじめに

物体の変形シミュレーションはCGの分野において広く研究がなされており、その主な目的のひとつとして、なるべく低い計算コストでシミュレーションを実現することが挙げられる。物体が断裂する時、その断裂面は多くの場合複雑な形状となるため、それをシミュレーションするのは特に計算コストがかかる。そのため既存の有限要素法[1]などでは、複雑な断裂を伴う変形シミュレーションをリアルタイムで行うことは困難であった。

提案法では物体に格子を当てはめる（図１参照）ことで、変形と断裂のシミュレーションを行う。そして、各格子に対して幾何学的に導出される仮想的な力を適用することで、高速かつ数値的に安定に変形と断裂の計算を行う。その際に、物体の変形形状や断裂の状態に応じて、局所的に格子をさらに細かく分割することで、複雑な断裂を伴うインタラクティブな変形シミュレーションの実行を可能にする。

2 本研究の概要

本研究では Rivers ら[2]の均一な格子を用いた変形手法(Lattice Shape Matching; LSM)を発展させ、格子のサイズが均一でない場合も扱えるようにする。これによって格子を形状に合わせて局所的に細かくしたり、変形や断裂の状態に合わせたアダプティブな分割ができる。また、提案法は多角形メッシュで表現された物体の変形を行う。現時点では 2 次元形状のみを扱っており、物体の表示に多角形メッシュの代わりに格子を用いることで、変形、断裂時のメッシュ頂点の位置座標を補間する手間を省き、計算コストを削減している。

アルゴリズムの流れは、「格子のアダプティブな分割」→「断裂」→「変形」を１タイムステップとして繰り返していく。以下、各アルゴリズムについて説明する。

Animations of Lattice-based Shape Deformations with Complex Fracturing
Makoto Ohta and Tomoyuki Nishita
The University of Tokyo

3 変形のアルゴリズム

本稿では物体の変形手法として LSM を用いているので、この節では LSM による変形を説明する。LSM ではまず前計算として物体に格子を当てはめ、変形のための計算点（以下パーティクルと呼ぶ）を各格子点に配置する。さらに、各パーティクルに隣接する八近傍のパーティクルを記憶する。変数 w をユーザが任意に決められる自然数で、物体の変形を制御するパラメタの一つとする。w の値は小さいと変形しやすく、大きいと変形しにくくなる。各パーティクルとその八近傍を含む領域を $w=1$ の領域と呼ぶ（つまり領域はパーティクルと同数作られる）。$w=1$ の領域に、これを囲む 16 のパーティクルを含む領域を合わせて $w=2$ の領域とする。同様に $w=3,4,\cdots$ と任意の w の値の領域が再帰的に作られる（図１）。

続いて変形の各タイムステップでの処理を説明する。現在の各パーティクル i の位置を $\mathbf{x}_i$ とする。また静止状態の領域内の全パーティクルに適切な平行移動・回転を適用して得られる形状を目標形状と呼び、その各パーティクル i の位置を $\mathbf{g}_i$ とする。この移動と回転は各 $\mathbf{x}_i - \mathbf{g}_i$ 間の距離の合計が最小になるように決める。次に各パーティクル i（現在位置 $\mathbf{x}_i$）に対して位置 $\mathbf{g}_i$ に近づけるような力を加える。パーティクル i が複数の領域に含まれている場合、各領域で加わる力の平均を加える（図２）。

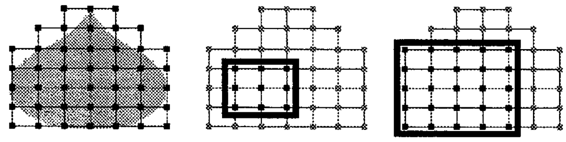

図１．物体に格子を当てはめ、各格子点にパーティクルを発生させたもの（左）と $w=1$ の領域（中央）、$w=2$ の領域（右）を黒枠で囲ったもの。

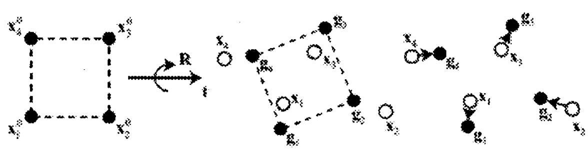

図２．現在の形状（位置 $\mathbf{x}_i$）に最も近づく移動 $\mathbf{t}$ と回転 $\mathbf{R}$ から目標形状（位置 $\mathbf{g}_i$）を求め、各パーティクル i に $\mathbf{g}_i$ へ近づけるような力を加える。

4 断裂のアルゴリズム

4.1 前計算

提案手法では、格子を構成する各四角形において、隣接する２つの四角形の重心間距離の変化率でアダプティブな分割や断裂が起こるかどうかを決める（詳細は 4.2 節以降）。そのため前計算として、各四角形について隣接する四角形をリストに登録する。

また、外力によって断裂が起こる場合、最初に断裂の起きる位置は物体の境界部分になる。最初の断裂位置はその後の断裂に大きく影響するので、より正確に断裂の始まる位置を把握するため、境界付近の格子のうち特に形状に沿わない格子はより細かく分割し、詳細に表現する（図３）。

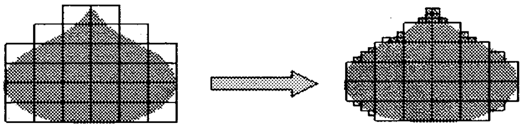

図３．境界付近の一部の格子を分割、不要な格子を除外することで、格子全体の形がより物体に近づく。

4.2 格子のアダプティブな分割

アニメーションの各タイムステップではまず、隣接する２つの四角形の現在の重心間距離の、静止状態からの変化率を計算する。この変化率の値が大きいほど、この２つの四角形の間に大きな力がかかっていると考えられる。そこでこの変化率の値がある閾値を超えていたら断裂が起こる可能性が高いとみなし、断裂面付近を詳細に表示するためにこの隣接する２つの四角形を半分のサイズに分割する（図４）。

新しい格子にもパーティクルを発生させる。また分割することによって新しい四角形が作られるので、新しい四角形とこれに隣り合う四角形の、隣接する四角形のリスト、さらに各領域に含まれるパーティクルの一覧を更新する。

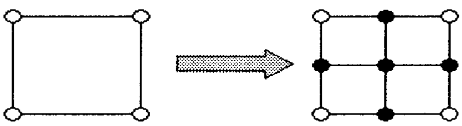

図４．新しいパーティクル（黒丸）を追加し、１つの四角形を４つに分割する。

4.3 断裂

4.2 節と同様に、各四角形に対して、隣接する四角形との重心間距離の静止状態からの変化率を計算し、これがある閾値（4.2 節で用いた閾値より大きな値）を超えていたらこの２つの四角形の間に断裂が起きるほどの力がかかっているとみなし、２つを切り離す。隣接する四角形は互いに接する部分のパーティクルを共有しているので、これを２つのパーティクルに分け、隣接する四角形のリストから互いを除く。

5 結果

提案法を用いて、Star 形の物体を左右に引きちぎった際に、アダプティブに格子を細分割することで断裂部を詳細に表現した結果を図５に示す。また、アダプティブな分割を行わず一様な格子を用いた場合と比較すると、計算コストを大きく削減することができた。C++および OpenGL を用いて実装を行い、Intel Core 2 2GHz、メモリ 1GB のマシンで比較実験を行った結果が表１である。図には示されていないが、モデル名”Teapot”はティーポット形状の２次元板を用いた。

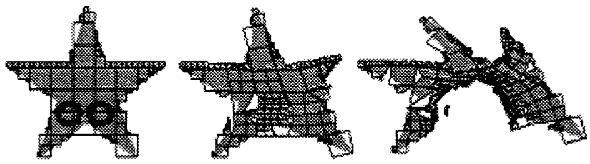

図５．格子（黒）のアダプティブな細分割による詳細な断裂面の表示の例。物体の左右（黒い太丸）を引っ張ってちぎっている。

表１．各変形、断裂方法での計算コストの比較。

モデル名	メッシュ頂点数	一様分割	アダプティブ分割
Star	1361	1.1fps	15.5fps
Teapot	4358	0.4fps	12.3fps

6 まとめと今後の課題

本稿では２次元の物体を格子状に分割し、さらにその格子に対してアダプティブな細分割を適用することで、複雑な断裂を伴う物体の変形を高速に計算する手法を提案した。

今後の課題としては、分割したが断裂しなかった格子を再統合することによる計算コストの削減、３次元の物体への拡張などが挙げられる。

参考文献

[1] J. F. O'Brien, A. W. Bargteil and J. K. Hodgins, "Graphical Modeling and Animation of Ductile Fracture", ACM SIGGRAPH 2002, pp.291-294.
[2] A. R. Rivers and D. L. James, "FastLSM: Fast Lattice Shape Matching for Robust Real-Time Deformation", ACM SIGGRAPH 2007, Article No.82.