

非均質弾性体のためのアダプティブな変形シミュレーション

Adaptive Deformation Simulation of Inhomogeneous Elastic Objects

今井 星† 楽 詠こう† 陳 炳宇 ‡,† 西田 友是 ‡
Sei IMAI†, Younghao YUE†, Bing-Yu CHEN‡,† and Tomoyuki NISHITA†

† 東京大学 † University of Tokyo

‡ 国立台湾大学 ‡ National Taiwan University

E-mail: †{etoile,egaku,nis}@nis-lab.is.s.u-tokyo.ac.jp, ‡robin@ntu.edu.tw

1 はじめに

ゼリーのような柔らかいもの（もしくは弾性体）の物理ベースな変形シミュレーションは、動画やテレビゲームなどの分野において重要である。これまで、コンピューターグラフィックスの分野において、弾性体の変形に関する非常に多くの研究がなされてきた。質量バネモデルや粒子ベースモデル [19, 11, 1] は、テレビゲームのようなインタラクティブなアプリケーションに用いられ、より正確な、すなわち連続体力学に基づいた FEM (Finite Element Method, 有限要素法) ベースのシミュレーション手法は、主に正確さが要求される動画に用いられている。FEM ベースのモデルは、計算コストが高いことが知られており、アダプティブ分割法などの多重解像度手法が提案されている [4, 8]。

一般的に多重解像度のアプローチを用いる FEM ベースの手法では、入力の弾性体を階層構造をもった複数の要素（本手法では四面体）に分割する。すなわち、物体を構成する複数の要素は多段階解像度（レベル）からなっており、木構造のような親子関係が作られる。典型的な多重解像度手法では、親要素は子要素の弾性行列を共有しており、物体は均質であるか均質な部分毎に要素を分けておく必要がある。さらに、典型的なアダプティブな手法では、四面体分割は変形中に動的に行われる。これらの手法を直接非均質な物体の変形に適用しようとすると、一つの親に属する子要素の弾性行列（硬さを表す値）が異なる場合には、各レベルにおける弾性行列を変形中に動的に計算するのは難しいことから、適用は困難であると考えられる。

本稿では、我々は与えられた硬さの関数に基づいてボトムアップサンプリングにより、全てのレベルにおけるすべての要素の弾性行列を計算する方法を提案する。提案手法を用いることで、隣接する子要素の弾性行列が連続的に異なるような場合も扱うことが可能となる。さらに分割と弾性行列の計算を前処理で行い、変形中に動的にアダプティブにレベルを変更することで、ランタイムでの計算コストを減らすことが可能である。

我々のアプローチはオフラインで行う前処理とランタイムに行う処理の 2 つからなる。前処理において、入力の弾性体はシミュレーションが安定するように、分割された四面体のクオリティがよくなるように、最初に再帰的に複数の四面体に分割する。次に、硬さを与える関数を元にサンプリングアプローチを用いて、最も細かいレベルから順に最も粗いレベルにわたる全てのレベル四面体要素の弾性行列を計算する。ボトムアップな処理により、我々は全てのレベルにおけるすべての要素の弾性行列を得ることができる。ランタイム時は、ユーザが与えた誤差上限と要素の歪みに応じてシミュレーション要素はアダプティブに選択される。提案手法を用いることにより、均質な変形と同じように非均質な変形をアダプティブに行うことができる。

本稿では粗いレベルの弾性行列のみを計算している。粘性行列（エネルギー散逸に寄与）や塑性（一定以上の外力が加わった時に元の形状に戻らなくなる性質）は考慮しておらず今後の課題である。

2 関連研究

コンピューターグラフィックスの分野において弾性体の変形シミュレーションに関する研究は数多くある [7, 14]。質量バネモデルや粒子ベースモデル [19, 11, 1] は、テレビゲームなどのインタラクティブなアプリケーションに用いられる。より正確な結果をえる方法に連続体力学に基づいた方法がある。このような方法はメッシュレスな手法 [6] とメッシュベースな FEM にわけることができる。本手法では後者に着目する。

一般にメッシュベースな FEM ではシミュレーション空間を四面体 [16, 13] や八面体 [2] などの有限の要素に分割する。これまで FEM を改善するための入力物体の分割に関する研究がなされてきた [18, 17]。また FEM のコストを減らすために、アダプティブな分割アプローチがよく用いられる [4, 8, 5]。これらのアプローチでは入力物体を歪みに応じて変形時に分割する。その一方でこの手法には均質物体にしか適用できないという問題がある。

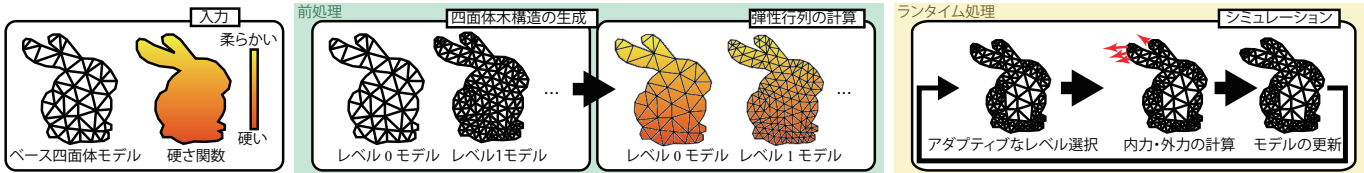


図1 アダプティブシミュレーションの概観. 簡単のため2次元で表示

非均質弾性体を FEM ベースでシミュレートする方法もある [3]. 非均質物体の変形シミュレーションの高速化アプローチに, 細かい要素の集合を粗い要素の集合として数値的に近似する方法がある [10]. この方法は, 非均質物体が粗い四面体で構成されており, 粗い四面体の弾性行列は細かい構造の弾性定数から計算される. そして細かい構造はレンダリング時のみ準静的に計算される. Nesme ら [15] の手法は非均質物体をグリッドに埋め込み, グリッドの弾性行列が計算される. しかし我々の知る限りにおいて, 非均質な物体をアダプティブにシミュレーションする方法はない.

3 アダプティブなシミュレーション手法

我々の手法はアダプティブなアプローチであり, シミュレーションは FEM ベース [16] で行う. 我々のアダプティブな手法では図 1 にあるように, 分割処理 (3.1 節) を弾性行列の計算 (4 節) とともにオフライン時に行い, ランタイム時の計算コストを軽減する (3.2 節). このように, 我々の手法はオフラインの前処理と変形シミュレーション時の処理の二つからなる.

3.1 四面体の木構造の生成

オフラインの前処理においてまずアダプティブなシミュレーションのために四面体の木構造を生成する. この時入力物体はあらかじめ粗い四面体に分割されているとする. 入力の四面体に基づいて再帰的に分割し木構造を生成する. ここで分割数に応じて各レベルを粗い方から順にレベル $0, 1, \dots, n$ と呼ぶ. 分割の時, 全ての四面体の形状をできる限り“ハイクオリティ”なままで分割する必要がある. さもないとシミュレーションが不安定になってしまう. 木構造を構成するための四面体分割法にはいくつかあるが, 我々は文献 [12] のような分割により四面体のみが生成されることが保証された方法を用いた. (詳細は 付録 A を参照.)

3.2 四面体フロントを用いたシミュレーション

四面体の木構造に対して, まず図 2 のようなメッシュ簡略化 [9] にもよく利用される「四面体フロント」を定義する.

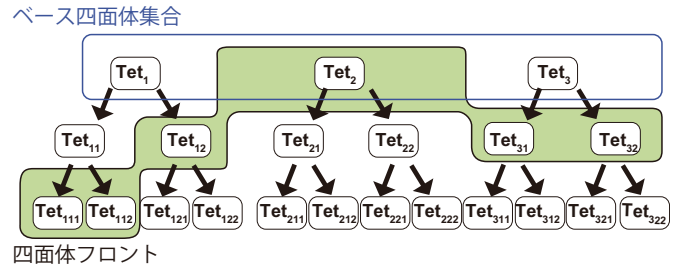


図2 四面体木構造のイラスト. 根を構成する「ベース四面体集合」は入力の四面体の集合でありそれと同時に最も粗いレベルである. 中間的な四面体の集合はシミュレーションに適切なレベルをとっている.

「四面体フロント」とはシミュレーションする際に適切なレベルの四面体の集合を指す. もし「四面体フロント」が全ての末端の四面体 (最も細かい四面体) のみから構成されていれば, シミュレーションは最も細かいレベルの四面体により正確なシミュレーションが行われる. 一方で「四面体フロント」が図 2 のような「ベース四面体集合」のみを含んでいる場合には, シミュレーションは入力の四面体の集合 (最も粗いレベルに属す) により最も高速にシミュレーションが行われる.

変形シミュレーションを行う前に, シミュレーションを精度を維持したままなるべく高速にシミュレーションできるように, 四面体木構造の「四面体フロント」を決定する必要がある (詳しくは 5.2 節と 5.1 節を参照).

「四面体フロント」を決定した後, 物体は次式に従って変形する.

$$\sigma = C\epsilon \quad (1)$$

この式は三次元に拡張されたフックの法則で, $\sigma, \epsilon \in \mathbb{R}^6$ はそれぞれ歪ベクトルと応力ベクトルを表し, $C \in \mathbb{R}^{6 \times 6}$ は硬さを表す値である弾性行列を表す.

変形シミュレーション時には, 次に述べるプロセスを毎タイムステップ繰り返す. まず, 各要素 (四面体) の歪ベクトル ϵ を計算する. 次に, 応力ベクトル σ を弾性行列 C を用いてフックの法則式 (1) に従って変形する. そして, そのタイムステップにおける各頂点の加速度, 速度, 座標を計算す

る。最後に 5.3 節で述べる T-ジャンクションに関する処理を行う。

4 弾性行列の計算と歪ベクトルの条件

四面体の分割により我々の木構造は四面体のみで構成される。入力した弾性物体を分割すると、最も細かいレベルにおけるすべての四面体は、ほとんど等分割されているとみなすことができ、これらの四面体の弾性行列 C は硬さを表す関数により得ることができる。次に粗いレベル (レベル $0 \sim n-1$) の要素 (四面体) の弾性行列 C をボトムアップサンプリングにより計算する。

ここで弾性行列 C とは式 (1) にあるように、応力ベクトル σ と歪ベクトル ϵ の線型な関係を表す。すなわち、もし適切な応力ベクトル σ と歪ベクトル ϵ が与えられれば、弾性行列 C を線型な計算式を解くことで弾性行列 C を計算できることを意味している。我々のボトムアップアプローチの基本的な考え方は、まず歪ベクトル ϵ に従って粗いレベルの四面体を動かす、細かいレベルにおける変形のシミュレーションをおこない、細かいレベルの応力ベクトル σ を計算、そして一段階粗い四面体の応力ベクトル σ として計算する。最後に我々は弾性行列 C を粗いレベルでの σ や ϵ の線型関係から計算する。詳細を次に述べる。

4.1 歪ベクトルの分解

線型式を簡単に解くために、まず歪ベクトル ϵ を次のように 6 つの基底に分解する。

$$\epsilon = \sum_{m=1}^6 \alpha^{(m)} \epsilon^{(m)} \quad (2)$$

ここで、 $m = 1, \dots, 6$ は基底のインデックスであり、 $\alpha^m \in \mathbb{R}$ は $-1 < \alpha^{(m)} < 1$ の値をとるスカラーであり、 $\epsilon^{(m)}$ は m 番目の要素が 1 で他が 0 となるような線型独立なベクトルの基底の一つである。

次に、フックの法則は線型なので、我々は式 (1) を $\sigma = \sum_{m=1}^6 \alpha^{(m)} C \epsilon^{(m)}$ と分解し、応力ベクトル σ の j 番目の要素 σ_j もまた $\sigma_j = \sum_{m=1}^6 \alpha^{(m)} \sum_{k=1}^6 C_{jk} \epsilon_k^{(m)} = \sum_{m=1}^6 \alpha^{(m)} C_{jm}$ と表す。ここで C_{jk} は j 行 k 列の要素を表す。計算すると

$$\sigma = \sum_{m=1}^6 \alpha^{(m)} C_m \quad (3)$$

となる。ただし C_m は C の m 列目を表す。

4.2 弾性行列の計算法

式 (3) に基づいて C の要素を決定する。各 m 行について $\alpha^{(m)}$ を 0 でないとし、弾性行列 C_m を $C_m = \sigma / \alpha^{(m)}$ として計算する。しかし、正確に C_m を計算するためには $\alpha^{(m)}$ 一つのみを用いて計算するには次の二つの理由から不十分である。まず、 σ は数値誤差を含む。次に、線型関係 $C_m = \sigma / \alpha^{(m)}$ は変形が小さい、すなわち $|\alpha^{(m)}|$ が十分小さい時のみ成立する。以上の理由から複数のスカラー $\alpha^{(m)}$ について、歪ベクトル $\epsilon^{(m)}$ と応力ベクトル $\sigma^{(m)}$ を計算する。そして、歪ベクトル $\epsilon^{(m)}$ と応力ベクトル $\sigma^{(m)}$ の関係から弾性行列 C_m を計算する (図 3)。この計算手法から、計算に線形性が成り立つ $\alpha^{(m)}$ の領域を得ることができる (詳細は 4.3 節を参照)。

レベル $n-1$ に属す未知の弾性行列をレベル n に属す既知の弾性行列を用いて計算する方法を次に示す。まず基底 m について $\alpha^{(m)}$ を決定し、歪ベクトル $\epsilon = \alpha^{(m)} \epsilon^{(m)}$ を決定する。次に、レベル $n-1$ の四面体の頂点を動かす、四面体がこの歪ベクトル ϵ となるようにする。これらのレベル $n-1$ の四面体の頂点は、レベル n の四面体により共有されている。レベル n の四面体のシミュレーションでは、このような共有された頂点は固定しておき、レベル n の四面体のみ属する残りの頂点を動かす。具体的には、まず、これらのレベル n の四面体のみ属する頂点の位置を、レベル $n-1$ の四面体の頂点座標から補間することにより初期化する。次に、レベル n の四面体を、レベル n の弾性行列 (既知) を用いて式 (1) に従ってシミュレーションする。なお、このシミュレーションにおいて、レベル n の四面体のみ属する頂点だけが更新される。応力が釣り合った後、レベル n の応力ベクトルを用いてレベル $n-1$ の応力ベクトル σ を計算する。そして歪ベクトルと応力の関係を図 3 のようにプロットする。 $\alpha^{(m)}$ の大きさを断続的に変えることで複数のサンプルを得ることができる。

最後に、大きく外れた点を除いた点について直線をフィッティングし、その傾きをレベル $n-1$ の弾性行列の値とする。

4.3 歪ベクトルの適用範囲

前節で述べたとおり弾性行列の計算に用いられる関係は小さな変形についてのみである。そのため弾性行列の要素 C_{jm} は次で述べる $\alpha^{(m)}$ が条件を満たすときのみ適切であるとすると

$$\frac{|\alpha^{(m)} C_{jm} - \sigma_j(\alpha^{(m)})|}{|\alpha^{(m)} C_{jm}|} < T_1 \quad (4)$$

ここで T_1 はユーザが指定した閾値であり C_{jm} は図 3 でフィッティングした直線の傾き、そして $\sigma_j(\alpha^{(m)})$ はシミュ

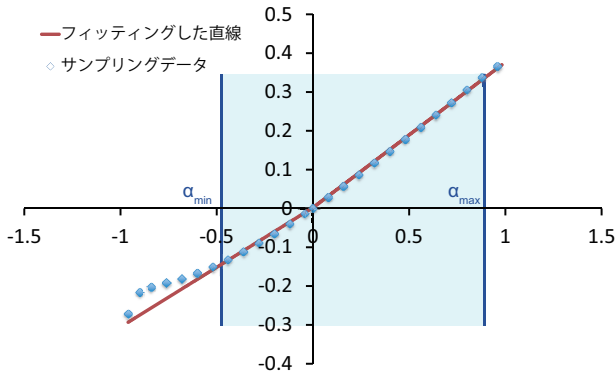


図3 $\alpha^{(m)}$ (x 軸) と応力ベクトル σ の要素 (y 軸) の関係。水色の領域は歪ベクトルの適用範囲。

レーションから得た値で、図3で青い点で表されている。

しかし、上記の条件だけで分母が小さい時に誤差が出てしまう。そこで、さらに次の条件を追加する。

$$|\alpha^{(m)} C_{jm} - \sigma_j(\alpha^{(m)})| < T_2 \quad (5)$$

ここで、 T_2 は別のユーザーが与えた閾値である。もし二つの条件のうちいずれかが満たされた場合に、線型条件が満たされたと判定する。閾値の上限 $\alpha_{max}^{(m)}$ と下限 $\alpha_{min}^{(m)}$ を求めるのに実験では値を $T_1 = 1.0 \times 10^{-2}$ 、 $T_2 = 1.0 \times 10^{-2}$ とした。

これら二つの閾値は誤差をコントロールするパラメータである。もしパラメータが小さいと、誤差は小さくなりより正確になる。このように範囲が小さくなるとシミュレーションは細かいレベルにより頻繁に変更される。

5 アダプティブなレベル選択

ランタイムステップは次の三つのステップからなる。1) 四面体フロントの更新、2) 四面体の応力の計算、外力の計算、そして各頂点の速度・座標の更新、そして3) T-ジャンクションにおける頂点の修正である。ステップ1)では、四面体フロントに含まれる全ての四面体について子の四面体を親の四面体に変更するか、親四面体の子の四面体に変更するか、もしくは変更をしないかについてチェックする。これらの変更条件については5.1節と5.2節で示す。ステップ2)については[16]を参照のこと。ステップ3)の詳細は5.3節で述べる。

5.1 親の四面体から子の四面体への変更条件

まず4.3節で述べた歪ベクトルの変更条件を考慮してより細かいレベルに変更する必要があるかを判定する。各四面体について歪ベクトル ϵ を式(2)に従って各基底 $\epsilon^{(m)}$ につ

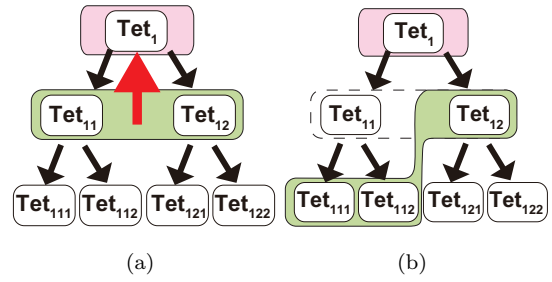


図4 (a) 粗い階層を細かい階層に変更できる場合と (b) できない場合。緑色の領域は四面体フロントを示しており、ピンク色の領域は変更する四面体の候補。

いて分解する。もしすべての m について $\alpha^{(m)}$ が次の条件

$$\alpha_{min}^{(m)} < \alpha^{(m)} < \alpha_{max}^{(m)} \quad (6)$$

を満たすならば選択されているレベルは適切であると判定する。逆に一つでも満たさない m がある場合には子の四面体に変更する。

5.2 子の四面体から親の四面体への変更条件

細かい四面体を親の四面体(粗い四面体)として扱うと細かい四面体として変形する場合に比べて誤差が生じてしまう。そのため細かいレベルを粗いレベルに変更するためには注意が必要であり、実際細かいレベルに変更する条件に比べて条件が多い。我々の手法において親階層に変更する条件は次に述べる3つの条件を全て満たす必要がある。

1. 親の歪ベクトルが式(6)の条件を満たし親の四面体が四面体フロントとして適切であると判定できる。
2. 全ての子の四面体が四面体フロントに含まれている。
3. 全ての子の四面体の歪ベクトルが全て親の四面体の歪ベクトルと同じである。

最初の条件は、粗いレベルを扱う際、歪ベクトルと応力ベクトルの関係が線型であると保証するのに必要な条件である。2つ目の条件は細かいレベルの四面体のいくつかより細かい階層になっている場合である。この条件は、子の四面体を選ばれていないときに、一気に親に移ると誤差が大きくなるからである。3つ目の条件は、子の四面体の実際の応力を考慮した条件である。もし四面体の応力が大きく異なる場合には、親階層の四面体を選択すると応力の近似がなりたたなくなってしまう。

5.3 T-ジャンクションの扱い方

異なるレベルの四面体同士が面した面上では図5にあるようなTジャンクション問題が生じる。Tジャンクション

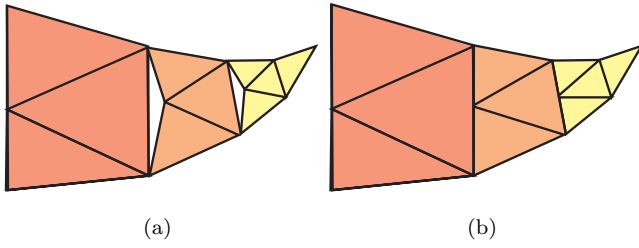


図 5 (a)T-ジャンクション問題により穴が生じた例。(b)T-ジャンクション上にある細かい四面体の頂点を粗い四面体の辺上の中点に固定して連続性を保証。

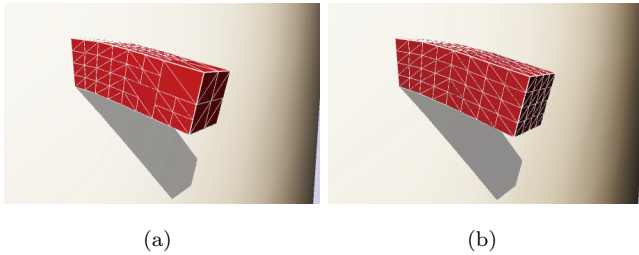


図 6 我々の手法 (a) と既存手法 (b) の比較

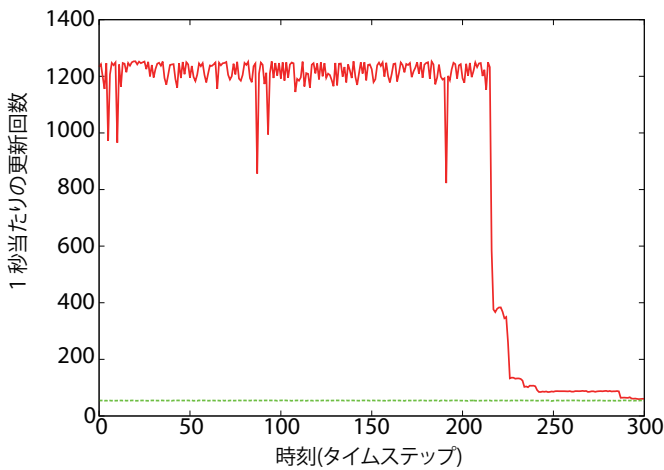


図 7 本手法 (赤) と既存手法 (緑) の 1 秒あたりの更新回数の比較 (値が大きいと計算時間が早い)。

上にある、細かいレベルの四面体の頂点でかつ粗い四面体には属さない頂点は、穴や重なりを起こしてしまう。この条件を避けるために、他のアダプティブな手法と同様に T-ジャンクション上の点を粗い四面体の辺の中点に移動修正する。この手続を粗いレベルから細かいレベルについて順に行うことで T-ジャンクション問題を解決できる。

6 結果

シミュレーションは Intel Core X980 3.33 GHz の CPU, 8GB のメモリを搭載したコンピューターで行った。

まず、直方体の片方を壁に固定し重力によって変形するシ

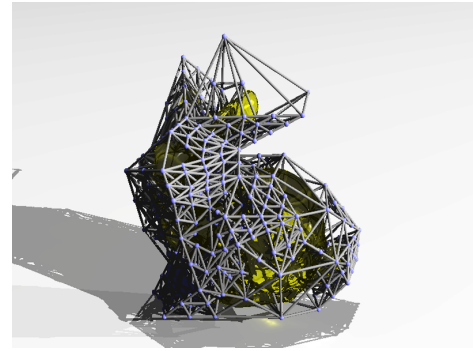


図 8 うさぎ型ゼリーのメッシュが四面体構造に埋め込まれている。レンダリング時にメッシュの全ての頂点が四面体構造をもとに補間された座標に移動する。

ミュレーションを我々の手法と既存手法 [16] とで比較した (図 6)。シミュレーション開始時には、我々の手法ではほとんどのパーツの変形が十分小さいため、直方体のほとんどが粗い四面体で構成されている。数秒後、直方体のいくつかの部分が細かいレベルに変化するが、残りの部分は変形がそれほど大きくないため元のレベルのままとなっている。変形が平衡状態に達した時、ほとんどの部分が最も細かいレベルに達している。我々のシミュレーションの結果は既存手法とほとんど同じ精度を示している。この変形シミュレーションにおける更新毎の必要な処理時間を図 7 に示した。シミュレーションの開始時にはほとんどの四面体が粗いレベルに属しているため、我々の処理速度は既存手法に比べて十分早い。シミュレーションが進むに連れて、レベルはより細かいレベルへと変わっていき、それと同時により多くの処理時間が必要となる。変形が十分大きくなり、すべての部分が最も細かいレベルになると、計算処理時間は既存手法と (おおよそ) 同じになる

レンダリング用の滑らかなメッシュは図 8 のように四面体構造に埋め込まれる。初期化の段階で埋め込まれた滑らかなメッシュがどの最も細かい四面体に対応するかを決定する。レンダリングのステップでは、埋め込まれたスムーズなメッシュのすべての頂点を補間された座標に移動する。

より複雑な結果を示すために重力により変形した均質に硬いゼリー (上段) と硬さが連続的に異なる非均質なゼリー (下段) の比較シミュレーションの結果を示す (図 9)。連続的に硬さがことなるウサギと均等に硬いウサギを比較すると、足元は均質なウサギと同じように変形している一方で頭のほうが柔らかくなっているウサギの方が耳が大きくなっているのがわかる。

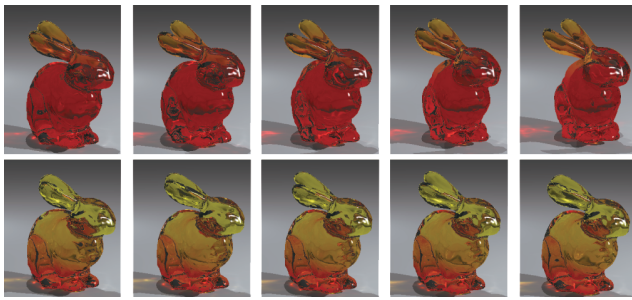


図9 均質なうさぎ型ゼリー（上段）と非均質なうさぎ型ゼリー（下段）の変形シミュレーションの比較。均質なうさぎは全体が柔らかく、非均質なうさぎ型ゼリーの硬さは底面から上面に従って連続的に異なる。足元が硬く（赤）耳の付近は柔らかい（黄）。

7 まとめと今後の課題

本論文では FEM ベースな均質および非均質な物体の弾性変形をアダプティブにシミュレーションする手法を提案した。

我々は、子の既知の弾性行列を用いて粗い階層の弾性行列の値をボトムアップなサンプリングアプローチにより計算する方法を提案した。また、既存のアダプティブな手法のようにランタイム中に動的に分割するのではなく、分割処理を前処理に行うことでランタイムの計算コストを軽減した。さらに、歪ベクトルに応じてユーザーが与えた誤差閾値による動的な階層選択によるアダプティブシミュレーション手法を提案した。

我々の手法は、粗い四面体が選択されている時には各タイムステップの処理時間は既存手法より速くなる。逆に最も悪い時においても、処理時間は既存手法とほとんど同じか少し遅くなるくらいである。サンプリングアプローチにより粗いレベルの弾性行列の計算をしたが、粘性定数も同様な手法により計算が可能だと考えられる。将来的には手法を拡張し、塑性や破壊も実現したいと考えている。

謝辞

本研究は科研費（基盤研究（C）23500112）の助成を受けたものである。

参考文献

- [1] D. Baraff and A. Witkin. Large steps in cloth simulation. In *ACM SIGGRAPH 1998 Conference Proceedings*, pp. 43–54, 1998.
- [2] S. Capell, S. Green, B. Curless, T. Duchamp, and

- Z. Popović. A multiresolution framework for dynamic deformations. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 41–47, 2002.
- [3] N. Chentanez, R. Alterovitz, D. Ritchie, L. Cho, K. K. Hauser, K. Goldberg, J. R. Shewchuk, and J. F. O'Brien. Interactive simulation of surgical needle insertion and steering. *ACM Transactions on Graphics*, Vol. 28, No. 3, pp. 88:1–88:10, 2009.
- [4] G. Debunne, M. Desbrun, M.-P. Cani, and A. H. Barr. Dynamic real-time deformations using space & time adaptive sampling. In *ACM SIGGRAPH 2001 Conference Proceedings*, pp. 31–36, 2001.
- [5] J. Dequidt, D. Marchal, and L. Grisoni. Time-critical animation of deformable solids. *Computer Animation and Virtual Worlds*, Vol. 16, No. 3-4, pp. 177–187, 2005.
- [6] F. Faure, B. Gilles, G. Bousquet, and D. Pai. Sparse meshless models of complex deformable solids. *ACM Transactions on Graphics*, Vol. 30, No. 4, pp. 73:1–73:10, 2011.
- [7] S. F. F. Gibson and B. Mirtich. A survey of deformable modeling in computer graphics. Technical Report TR97-19, Mitsubishi Electric Research Laboratories, 1997.
- [8] E. Grinspun, P. Krysl, and P. Schröder. CHARMS: a simple framework for adaptive simulation. *ACM Transactions on Graphics*, Vol. 21, No. 3, pp. 281–290, 2002.
- [9] H. Hoppe. Progressive meshes. In *ACM SIGGRAPH 1996 Conference Proceedings*, pp. 99–108, 1996.
- [10] Lily Kharevych, Patrick Mullen, Houman Owhadi, and Mathieu Desbrun. Numerical coarsening of inhomogeneous elastic materials. *ACM Transactions on Graphics*, Vol. 28, No. 51, pp. 51:1–51:8, 2009.
- [11] Y. Lee, D. Terzopoulos, and K. Waters. Realistic modeling for facial animation. In *ACM SIGGRAPH 1995 Conference Proceedings*, pp. 55–62, 1995.
- [12] Anwei Liu and Barry Joe. Quality local refinement of tetrahedral meshes based on 8-subtetrahedron subdivision. *SIAM Journal on Scientific Computing*, Vol. 16, No. 6, pp. 1269–1291, 1995.
- [13] Matthias Müller, Leonard McMillan, Julie Dorsey, and Robert Jagnow. Real-time simulation of deformation and fracture of stiff materials. In *Proceedings*

of the 2001 Eurographic Workshop on Computer Animation and Simulation, pp. 113–124, 2001.

- [14] A. Nealen, M. Muller, R. Keiser, E. Boxerman, and M. Carlson. Physically based deformable models in computer graphics. *Computer Graphics Forum*, Vol. 25, No. 4, pp. 809–836, 2006.
- [15] Matthieu Nesme, Paul G. Kry, Lenka Jeřábková, and François Faure. Preserving topology and elasticity for embedded deformable models. *ACM Transactions on Graphics*, Vol. 28, No. 3, pp. 52:1–52:9, 2009.
- [16] J. F. O’Brien and J. K. Hodgins. Graphical modeling and animation of brittle fracture. In *ACM SIGGRAPH 1999 Conference Proceedings*, pp. 137–146, 1999.
- [17] S. Schaefer, J. Hakenberg, and J. Warren. Smooth subdivision of tetrahedral meshes. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry processing*, pp. 147–154, 2004.
- [18] J. R. Shewchuk. Tetrahedral mesh generation by Delaunay refinement. In *Proceedings of the 14th Annual Symposium on Computational Geometry*, pp. 86–95, 1998.
- [19] X. Tu and D. Terzopoulos. Artificial fishes: physics, locomotion, perception, behavior. In *ACM SIGGRAPH 1994 Conference Proceedings*, pp. 43–50, 1994.

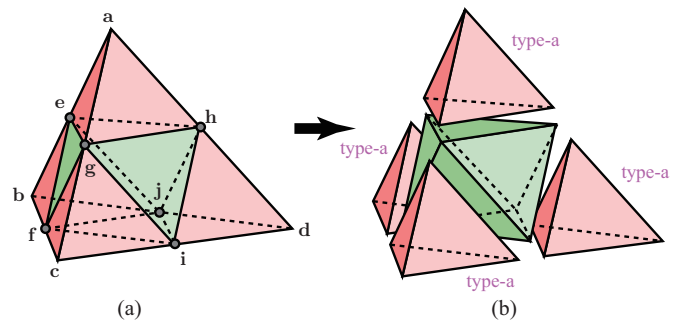


図 10 (a) タイプ-a の四面体 $abcd$. 辺 ab, bc, ac, ad, cd, bd の中点をそれぞれ e, f, g, h, i, j とする. (b) 4 つの四面体 $aegh, ebfj, gfci, hjid$ は $abcd$ を分割してできた四面体. これらの 4 つの四面体は $abcd$ に相似である. この分割により八面体が残る.

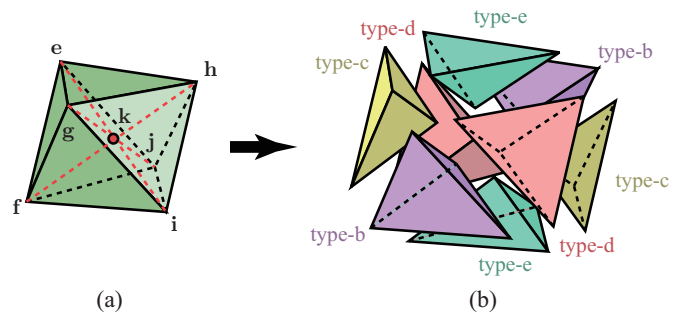


図 11 (a) 八面体 $efghij$. 線分 ei, gj, fh は k で交わり, この点はそれぞれの辺の中点である. (b) この八面体を頂点 k と八面体の面を結んで出来る 8 つの四面体に分割する. これら 8 つの四面体は合同性から 4 つのタイプ (タイプ-b, タイプ-c, タイプ-d, タイプ-e) に分けることができる.

法とその証明を示す.

入力の四面体, そしてその四面体に相似な四面体をタイプ-a と呼ぶ. まず, タイプ-a の四面体の分割法について説明する. タイプ-a 上の頂点をそれぞれ a, b, c, d とする (図 10(a)). 四面体をこれらの頂点をつかって $abcd$ のように表す. 辺 ab, bc, ac, ad, cd, bd の中点をそれぞれ e, f, g, h, i, j とする. $e = \frac{a+b}{2}, f = \frac{b+c}{2}, g = \frac{a+c}{2}, h = \frac{a+d}{2}, i = \frac{c+d}{2}, j = \frac{b+d}{2}$.

四面体 $abcd$ は 4 つの四面体 $aegh, ebfj, gfci, hjid$ に分割される (図 10(b)). これらの 4 つの四面体は四面体 $abcd$ に相似となる. なぜならこれらの四面体の辺は全て四面体 $abcd$ の辺の半分の長さであるからである. これら 4 つの四面体を四面体 $abcd$ から取り除くと八面体 $efghij$ が残る (図 11(a)). 線分 ei, gj, fh は頂点 k で交わり, この点は各線分の中点である. $k = \frac{e+i}{2} = \frac{g+j}{2} = \frac{h+f}{2} = \frac{a+b+c+d}{4}$.

次に図 11(b) のようにこの八面体を頂点 k と八面体の面を結んで出来る 8 つの四面体 $fgik, efgk, efjk, fijk, ehjk, hijk, ghik, eghk$ に分割する. これら 8 つの四面体は, それ

付録 A 四面体分割

四面体分割手法は [17] を参考にした. 再帰的に分割された四面体の集合で安定なシミュレーションを実現するためには, 1) 四面体の辺の比率, 2) 四面体の辺の長さのうち最も短い辺, そして 3) 四面体の内角のうち最大の角 の三つの基準から定義されるクオリティが十分高い必要がある. 我々の分割手法はこの条件を充たす必要がある. 本手法の特徴は, 再帰的に四面体を分割しても 5 種類のみ四面体のみから構成される点である (それぞれのタイプの四面体はお互いに相似, もしくは合同である). この特徴により, 生成された四面体の辺の比率の組み合わせはレベルにかかわらず 5 種類のみ抑えることが可能である. さらに四面体の内角のうち最大となる角もまたレベルにかかわらず 5 種類のみ抑えることができる. 最後に同じタイプ, レベルの四面体はお互いに合同である. このように四面体の最も短い辺はレベルの深さと四面体のタイプから計算できる. 次に具体的な分割

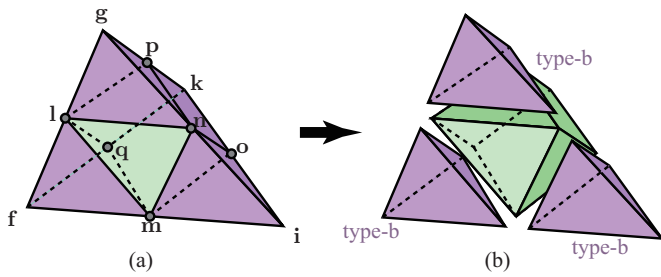


図 12 (a) タイプ-b の四面体 fgik. 辺 fg, fi, gi, ik, gk, fk の中点をそれぞれ l, m, n, o, p, q と表す. (b) 3 つの四面体 glnp, lfmq, nmio は fgik を分割してできた四面体. これら 3 つの四面体は四面体 fgik に相似である. 四面体 fgik からこれら 3 つの四面体を取り除くと 7 面体 lmnopqk が残る.

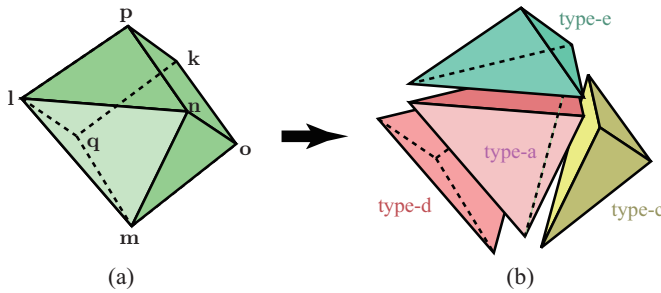


図 13 (a) 七面体 lmnopqk. (b) この七面体を 4 つの四面体 lmnk, nmok, lmqk, lnpk に分割する. この四面体はそれぞれタイプ-a, タイプ-c, タイプ-d, タイプ-e の四面体となる.

それ fgik, efgk, efjk, fijk と ehjk, hijk, ghik, eghk とが辺の長さが等しくなるため互いに合同となる. このようにして 8 つの四面体は 4 つのタイプ, タイプ-b, タイプ-c, タイプ-d, タイプ-e に分けることができる.

次にタイプ-b の四面体の分割法について説明する. タイプ-b の四面体 fgik を例にとって説明する (図 12(a)). 辺 fg, fi, gi, ik, gk, fk の中点をそれぞれ l, m, n, o, p, q とする. $l = \frac{f+g}{2} = \frac{a+b+2c}{4}$, $m = \frac{f+i}{2} = \frac{b+2c+d}{4}$, $n = \frac{g+i}{2} = \frac{a+2c+d}{4}$, $o = \frac{i+k}{2} = \frac{a+b+3c+3d}{8}$, $p = \frac{g+k}{2} = \frac{3a+b+3c+d}{8}$, $q = \frac{f+k}{2} = \frac{a+3b+3c+d}{8}$. まず四面体 fgik を三つの四面体 glnp, lfmq, nmio に分割する. これら三つの四面体は四面体 fgik に相似である. 何故なら三つの四面体の辺の長さは fgik の辺の長さの半分だからである. 四面体 fgik からこの 3 つの四面体を取り除くと七面体 lmnopqk が残る (図 13(a)). 次に, 七面体を 4 つの四面体 lmnk, nmok, lmqk, lnpk に分ける (図 13(b)). これら 4 つの四面体がそれぞれタイプ-a, タイプ-c, タイプ-d, タイプ-e に属することを次に示す.

まず四面体 lmnk がタイプ-a に属することを示す. この四面体の各辺のベクトルは $l - m = \frac{a-d}{4}$, $m - n = \frac{b-a}{4}$,

$n - l = \frac{d-b}{4}$, $l - k = \frac{c-d}{4}$, $m - k = \frac{c-a}{4}$, $n - k = \frac{c-b}{4}$. のように計算される. このように四面体 lmnk のすべての辺がタイプ-a の四面体 aegh の辺の長さの $\frac{1}{2}$ となっている. (もしくはタイプ-a の四面体 abcd の辺の長さの $\frac{1}{4}$ となっている.) よって四面体 lmnk は四面体 abcd に相似であることがわかり, タイプ-a に属することが示された.

次に四面体 nmok がタイプ-c (つまり efgk に相似) であることを示す. まずタイプ-c の四面体 efgk の辺のベクトルは $e - f = \frac{a-c}{2}$, $e - g = \frac{b-c}{2}$, $e - k = \frac{a+b-c-d}{4}$, $f - g = \frac{b-a}{2}$, $f - k = \frac{-a+b+c-d}{4}$, $g - k = \frac{a-b+c-d}{4}$ となる. 同様に四面体 nmok の辺のベクトルは $m - n = \frac{b-a}{4} = \frac{f-g}{2}$, $n - o = \frac{a-b+c-d}{8} = \frac{g-k}{2}$, $k - n = \frac{b-c}{4} = \frac{e-g}{2}$, $m - o = \frac{-a+b+c-d}{8} = \frac{f-k}{2}$, $k - m = \frac{a-c}{4} = \frac{e-f}{2}$, $k - o = \frac{a+b-c-d}{8} = \frac{e-k}{2}$ と計算できる. このように四面体 nmok のすべての辺がタイプ-c の四面体 efgk の $\frac{1}{2}$ になっている. 以上から四面体 nmok が四面体 efgk に相似でありタイプ-c に属することが示された.

上記の証明と同様にして四面体 lmqk, lnpk がそれぞれ efjk, fijk に相似であることが示せる. このようにして lmqk, lnpk がタイプ-d, タイプ-e に属することを示せる.

タイプ-c, タイプ-d, タイプ-e の分割法は上記に述べたタイプ-b の四面体の分割法と同じである. このようにして与えられた入力四面体から分割された四面体はたったの 5 種類に分類することができる.

最後に同じタイプ, レベルの四面体は互いに合同であることを示す. 同じタイプの四面体は互いに相似であることは既に示しており体積が同じことを示せば十分である. タイプ-a の四面体の体積を V として表す. この時レベル n のタイプ-a の四面体の体積は $(1/8^n)V$ となる. おなじようにレベル 1 のタイプ-b, タイプ-c, タイプ-d, タイプ-e の四面体の体積を V とするとレベル $n+1$ のそれぞれの四面体は $(1/2)(1/8^n)V$ となる. このようにして最も短い辺の長さは四面体のレベルとタイプにより制限されることがわかる.