

# 位置ベース物理

Matthias Müller Bruno Heidelberger Marcus Hennix John Ratcliff

AGEIA

## 概要

コンピュータグラフィックスにおける物理シミュレーションのための最も主要なアプローチは、力ベースのものです。ニュートンの運動の第2法則にもとづいて計算された加速度から、内部と外部の力は合成されます。時間積分により速度が更新され最終的にオブジェクトの位置が更新されます。2,3 のシミュレーション手法(多くは剛体のシミュレーションですが)は、力積ベースの手法を用いて直接速度を更新します。この論文では、速度更新段階を取り除き、直接的に位置を更新できる手法を提案します。位置ベースの手法の主要な長所はその可制御性です。力ベースのシステムを陽的積分手法で解いた場合に発生する発散する問題を回避することができます。それに加えて、衝突の制約は、簡単に扱うことが出来ますし、貫通した際も正確な位置へ射影することで完全に解決が可能です。我々は、この手法を実時間での衣服シミュレータへ用いてきました。これは、ゲーム用の物理ソフトウェアライブラリの一部で、本手法の強力さと利点を示すものとなります。

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object ModelingPhysically Based Modeling; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and RealismAnimation and Virtual Reality

## 1. 導入

コンピュータ・グラフィックス分野での物理ベースアニメーション分野の研究においては、剛体や変形物体や流体の流れの運動のような物理現象の新しいシミュレーション手法を探すことに関心があります。計算科学においては正確さに重きが置かれるのは対照的に、ここでの主要な議題は、安定性、堅牢さ、視覚的にもっともらしさを残した結果における計算速度とします。それ故、既存の計算科学手法は一対一には対応しません。実際のところ、コンピュータ・グラフィックスにおける物理シミュレーションの研究を行う際には、特殊な手法を思いついたり、特定用途に特化した手法になることが正当化されます。我々が提案する手法は、このような分類になるかと思います。

動的なオブジェクトをシミュレーションするための従来の手法は、力ベースのものです。時間刻みの最初で、内部と外部の力が累積されます。内力の例としては、弾性力や粘性、流体内部の圧力などがあります。重力や衝突力は外力の例です。ニュートンの運動の第2法則は、質量により加速する力と関係があります。

したがって、密度か質量が一点に集中した(=質点)頂点群を用い、力を加速度に転化します。どの積分方法でも、最初に加速度から速度を求め、速度から位置を計算します。いくつかの手法では力の代わりに力積を用いアニメーションの制御を行います。したがって、力積が速度を直接変更します。積分の段階が一つスキップされているわけです。

コンピュータ・グラフィックス分野、とりわけビデオゲームでは、オブジェクトの位置やメッシュの頂点を直接制御することが望ましいことがあります。ユーザーは、運動するオブジェクトへ頂点をアタッチしたいかもしれないですし、頂点が常に衝突オブジェクトの外側へ確実に維持されるようにしたいわけです。ここで我々が提案する手法は、位置へ直接働きますし、そのような操作を簡単に実現します。加えて、位置ベースの手法を用いることで、積分を直接行うことが出来ます。そのため陽積分手法で生じるオーバーシュート問題やエネルギーが増大してしまう問題を回避することが出来ます。したがって、位置ベース物理の主要な機能と長所は下記の通りです。

- ・位置ベースのシミュレーションは、陽積分を超えたコントロール性を与え、典型的な不安定性問題を解決します。

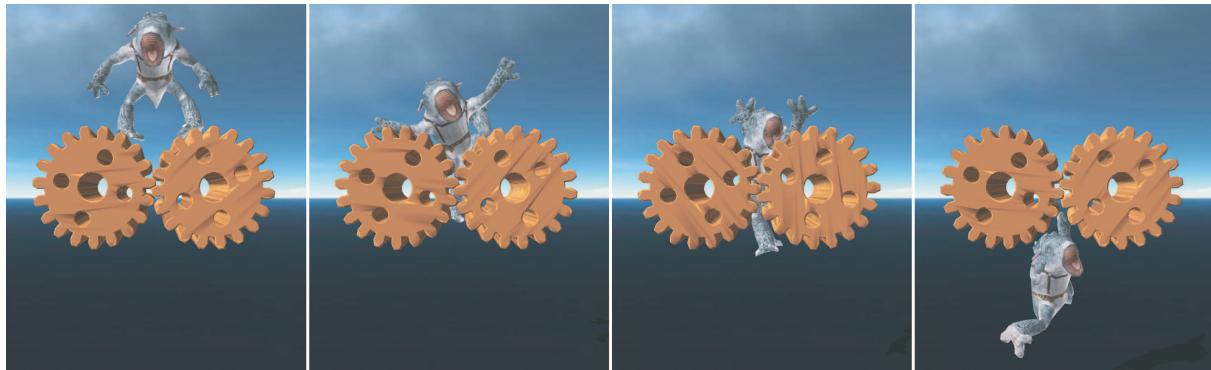


図1:ある知られた変形物質のベンチマークテスト。ここでは、衣服をきた人物が圧縮されます。

- ・頂点位置とオブジェクト位置はシミュレーションの間、直接的に操作が可能です。
- ・我々が提案する定式化手法は、位置ベースの設定で一般的な制約条件の制御を可能とします。
- ・陽的な位置ベースのソルバーは、理解しやすく実装も容易です。

## 2. 関連研究

[NMK\*05]によって報告された最新報告は、コンピュータ・グラフィックスで用いられる手法のよい導入資料となります。例えば、質点バネシステムや有限要素法や有限差分法が扱われています。[MHTG05]の引用から離れて、位置ベース物理は、この調査には入っていません。しかしながら、位置ベースの手法は、明示的にそう呼ばれることなく、完全なフレームワークとして定義されることなく様々な論文すでに記述されていました。

Jakobsen [Jak01]は、彼の Fysix engine を位置ベース物理の手法で作りました。彼の中心となるアイデアは、Verlet 積分手法を用い、位置を直接操作することです。しかしながら、速度は現在と1フレーム前の位置を保存することで明示的に保存されています。一方彼は、主として距離の制約条件に集中しています。彼が、どのようにより一般的な制約条件が制御できるのかのぼんやりとしたヒントを与えました。我々が提案するこの論文で、一般的な製薬を制御するための完全に一般的な手法を提案します。我々は、位置の射影によって並進運動と回転運動の重要な保存問題に集中して記述します。我々の手法では、1フレーム前の座標を保持する代わりに陽的な速度を用います。このことがダンピングと摩擦のシミュレーションをより簡単にします。

Desbrun [DSB99] と Provot [Pro95] は、質点バネシステムにおいて制約条件の射影を用いることでバネが発散する問題を回避しました。その一方で、完全な位置ベースの手法は、そのような伸びすぎたバネの処理を洗練させることのみに用いられ、基本的なシミュレーション手法としては使われていません。

Bridson らは、衣服のシミュレーション [BFA02] に

従来の力ベースの手法を用いて、幾何的な衝突と組み合わせて位置に基づいて解決するアルゴリズムを用いています。このアルゴリズムは、力積に基づいて衝突解決を確実なものにします。運動する物体の衝突の訂正するアイデアは、Volino ら [VCMT95] によっても提案されています。

Clavet ら [CBP05] によって用いられた位置ベースの手法は、粘弹性流体のシミュレーションに用いられました。彼らの手法は、完全に位置ベースの手法ではありません。なぜなら、時間刻みが様々な位置の射影で用いられているからです。したがって、積分は、通常の陽的な積分として限定的に安定しています。

Müller ら [MHTG05] は、変形オブジェクトを、ある目標位置へ向かう点としてのシミュレーションしました。これは、静止状態と現在のオブジェクトの状態とのマッチングを用いる手法です。その積分手法は、我々がここで提案するものに最も近いものです。それらは、一つの特別な目標制約を用いるだけなのです。従って、位置のソルバが必要ではありません。

Fedor [Fed05] は、Jakobsen の手法を用いゲームの人物のシミュレーションを行いました。彼の手法は、人物のシミュレーションの特定の問題向けに調整されています。彼は様々な骨の代理表現を行い、射影を通じて同期を保ちました。

Faure [Fau98] は、Verlet 積分手法を速度ではなく位置の修正に用いました。更新された位置は、制約の線形化法で計算されます。その一方我々は、非線形の制約関数を直接用います。

我々は、一般的な制約条件を [BW98] と [THMG04] で示される制約関数として定義します。力を計算する代わりに、制約関数のエネルギーの微分として、我々は平衡配位を直接解き位置を射影します。我々の手法では、衣服のシミュレーションのための曲げ項は、[GHDS03] と [BMF03] で提案されたものと似ていますが、位置ベースの手法を採用しています。

節4 では、衣服のシミュレーションに位置ベースの手法を用います。衣服のシミュレーションは、近年コンピュータグラフィックスの分野では活発に研究されている分野です。この分野の主要な論文を引用する代わりに、我々は読者に [NMK\*5] を完全なサーベイとして薦めます。

### 3. 位置ベースのシミュレーション

このセクションでは、一般的な位置ベース手法を定式化していきます。衣服のシミュレーションでは、結果のセクションで一連の手法の特別な応用例を示します。我々は3次元空間を考慮しています。しかしながら、手法は2次元でも同様に適用可能です。

#### 3.1. アルゴリズムの概要

我々は、動的なオブジェクトを  $N$  頂点の集合と制約条件  $M$  として表現します。頂点  $i \in [1, \dots, N]$  で、質量  $m_i$ 、位置  $\mathbf{x}_i$  と速度  $\mathbf{v}_i$  を持ります。

制約条件  $j \in [1, \dots, M]$  は、下記で構成されます。

- 基数  $n_j$ ,
- 関数  $C_j: \mathbb{R}^{3n_j} \rightarrow \mathbb{R}$ ,
- インデックス  $\{i_1, \dots, i_{n_j}\}$ ,  $i_k \in [1, \dots, N]$
- 硬さパラメータ  $k_j \in [0, \dots, 1]$  と
- 等式か不等式化のどちらかを示すタイプ

等式タイプの制約条件  $j$  は  $C_j(\mathbf{x}_{i_1}, \dots, \mathbf{x}_{i_{n_j}}) = 0$  の時に条件を満たします。不等式タイプの場合は

$C_j(\mathbf{x}_{i_1}, \dots, \mathbf{x}_{i_{n_j}}) \geq 0$  の時に条件を満たします。硬さパラメータ  $k_j$  は、制約条件の強さを 0 から 1 の範囲で定義しています。

このデータ構造とタイムステップ  $\Delta t$  に基づいて、動的なオブジェクトは以下のようにシミュレーションされます。

```
(1) forall vertices  $i$ 
(2)   initialize  $\mathbf{x}_i = \mathbf{x}_i^0, \mathbf{v}_i = \mathbf{v}_i^0, w_i = 1/m_i$ 
(3) endfor
(4) loop
(5)   forall vertices  $i$  do  $\mathbf{v}_i \leftarrow \mathbf{v}_i + \Delta t w_i \mathbf{f}_{ext}(\mathbf{x}_i)$ 
(6)   dampVelocities( $\mathbf{v}_1, \dots, \mathbf{v}_N$ )
(7)   forall vertices  $i$  do  $\mathbf{p}_i \leftarrow \mathbf{x}_i + \Delta t \mathbf{v}_i$ 
(8)   forall vertices  $i$  do generateCollisionConstraints( $\mathbf{x}_i \rightarrow \mathbf{p}_i$ )
(9)   loop solverIterations times
(10)  projectConstraints( $C_1, \dots, C_{M+M_{coll}}, \mathbf{p}_1, \dots, \mathbf{p}_N$ )
(11)  endloop
(12)  forall vertices  $i$ 
(13)     $\mathbf{v}_i \leftarrow (\mathbf{p}_i - \mathbf{x}_i) / \Delta t$ 
(14)     $\mathbf{x}_i \leftarrow \mathbf{p}_i$ 
(15)  endfor
(16)  velocityUpdate( $\mathbf{v}_1, \dots, \mathbf{v}_N$ )
(17) endloop
```

行(1)-(3) は、ただ変数の値を初期化するだけです。位置ベース物理の主要なアイデアは行 (7), (9)-(11) と (13)-(14) で示されます。行 (7), では、陽的にオイラー積分ステップを用いることで新しい頂点位置  $\mathbf{p}_i$  を推定します。(9)-(11) 行の反復ソルバーにより、制約条件を満たすようにこれらの位置の操作を行います。これは、それぞれの制約条件を繰り返しガウスサイデル法のように射影する事で実現します。(節 3.2 を参照ください)。ステップ (13) と (14) で、頂点の位置はより正しく推定され速度もそれに応じて更新されます。

これは、[Jak01] で示されている、Verlet 積分のステップと現在の位置の修正部分と正確に一致します。なぜなら Verlet 手法は速度を最後の位置と現在の位置の間の距離として陰的に保持しているからです。しかしながら、速度を持って動かすことにより操作を直感的に扱うことが可能となります。

速度は、行 (5), (6) と (16) で操作されます。行 (5) は、なんの力も位置の制約へと変換できない時に外力をシステムに伝えます。我々は、それを重力をシステムに伝えるためにのみ伝えます。その場合、行は  $\mathbf{v}_i \leftarrow \mathbf{v}_i + \Delta t \mathbf{g}$  となります。ここで  $\mathbf{g}$  は重力加速度です。行 (6) では、速度は必要に応じてダンピングされます。節 3.5 では、オブジェクトの剛体の様式に影響をあたえることなくグローバルなダンピングをどのように加えるかを示します。最終的に、行 (16) では、衝突した頂点が、摩擦と慣却係数とともに修正されます。

与えられた制約条件  $C_1, \dots, C_M$  は、シミュレーションの間を通して固定です。これらの制約に加えて、行 (8) は、衝突による制約条件  $M_{coll}$  を生成し、これはタイムステップ毎に変化します。行 (10) の射影ステップは、固定と衝突の両方の制約を考慮に入れわれます。

この仕組は、無条件に安定しています。これは、積分ステップ (13) と (14) は、従来の手法が行うような、未来の時間まで無分別に外挿しないからです。しかしながら、頂点は制約条件のソルバーによって  $\mathbf{p}_i$  が物理的に正しく動きます。唯一、不安定な可能性があるのは、ソルバ自身で、ニュートン・ラフソン法を用いるで適切な位置を求める場合です。(節 3.3 を参照ください)。しかしながら、その安定性はタイムステップサイズに依存しているんではなく、制約条件の関数の形に依存しています。

積分は、明確に陰的手法か陽的手法かのカテゴリーに分かれるわけではありません。もし、一つのソルバーの繰り返しがタイムステップごとに行われる場合は、陽的な積分手法のように見えるでしょう。しかしながら、繰り返し階数が増加するに連れて、制約システムは任意の硬さを持ちアルゴリズムは陰的な手法のように振る舞います。繰り返し回数が増加するに連れて、衝突検出とソルバーの計算コストが増加します。

#### 3.2. ソルバー

ソルバーへの入力は、 $M + M_{coll}$  の制約条件と推定位置  $\mathbf{p}_1, \dots, \mathbf{p}_N$  を頂点の新しい位置として用います。ソルバーは、全ての制約条件を満たすように推定位置を修正していきます。結果として得られる式の系は非線形となります。単純な距離の制約条件  $ii(\mathbf{p}_1, \mathbf{p}_2) (-\mathbf{p}_1 - \mathbf{p}_2) - ii$  でさえ、非線形な式となります。加えて、不等式の制約条件は不等式となります。そのような等式や不等式のような一般的な式を解くために、ガウスサイデル法のような繰り返し手法を用います。オリジナルのガウスサイデル法のアルゴリズム ÖGS) は、線形のシステムのみを扱います。GS から、一部借りてきたアイデアは、それぞれの制約条件を独立して解くアイデアです。しかしながら GS とは異なり、

制約条件を解くのは非線形な工程となります。繰り返し全ての制約条件を適用して粒子に射影し与えられた制約条件単独に関して適切な位置に修正します。ヤコビ法の繰り返しとは対照的に、粒子位置の修正は直ちに行われプロセスが可視化されます。これは、収束を劇的に早めます。なぜなら、圧力波が、単一のソルバステップで素材に広がり、制約条件を解く順番で効果があるからです。過剰に制約条件がある状況で、もし計算順序が一定でない場合は、このプロセスで、運動を引き起こすことがあります。

### 3.3. 制約条件の射影

制約条件にしたがって頂点集合の射影することとは、制約条件を満たすために頂点を動かすことを意味します。シミュレーションループの中で頂点を直接動かすことに関して最も重要な点は、並進運動量と回転運動量の保存です。 $\Delta\mathbf{p}_i$  を頂点  $i$  の射影による変異とします。並進運動量は、下記の際に保存されます。

$$\sum_i m_i \Delta\mathbf{p}_i = \mathbf{0}, \quad (1)$$

これは、質点中心の保存に相当します。回転運動量は、下記で保存されます。

$$\sum_i \mathbf{r}_i \times m_i \Delta\mathbf{p}_i = \mathbf{0}, \quad (2)$$

$\mathbf{r}_i$  は、 $\mathbf{p}_i$  の任意の共通回転中心からの距離となります。もし、射影がこれらの制約条件の1つに違反した場合、ゴースト力と呼ばれる、外力のようにオブジェクトを引っ張り回転させる力を導入します。しかしながら、内部の制約のみが運動量保存するために必要となります。衝突や、接続の制約条件は、オブジェクトへのグローバルな効果を持つことを許します。

我々が提案する制約条件を射影する手法では、内部の制約条件で両方の運動量を保存します。繰り返しますが、位置ベース手法では、エネルギー項([BW98, THMG04]を参照ください)経由の力を使う手法に対して、制約条件の関数を直接使うことでより直接的な操作が可能となります。制約条件を見てみましょう。頂点  $\mathbf{p}_1, \dots, \mathbf{p}_n$  上の濃度  $n$  とし、制約条件関数  $C$  と硬さ  $k$  とします。  $\mathbf{p}$  を

$[\mathbf{p}_1^T, \dots, \mathbf{p}_n^T]^T$  の連結と捉えます。内部の制約条件として、 $C$  は剛体の状態つまり移動と回転変換とは独立しています。これが意味することは、頂点が移動したり回転したりしても制約条件の関数の値は不变だということです。それ故

勾配  $\nabla_{\mathbf{p}} C$  は、剛体の状態とは直行しています。なぜなら勾配は最大の変異の方向だからです。もし、修正量  $\Delta\mathbf{p}$  が  $\nabla C_{\mathbf{p}}$  によって選ばれると両方の運動量は、自動的に保存されます。この場合全ての質量が均一であるとします。(後ほど異なる質量のケースを扱います)  $\mathbf{p}$  が与えられれば、修正量  $\Delta\mathbf{p}$  は、 $C(\mathbf{p} + \Delta\mathbf{p}) = 0$  より求められます。この式は下記で近似されます。

$$C(\mathbf{p} + \Delta\mathbf{p}) \approx C(\mathbf{p}) + \nabla_{\mathbf{p}} C(\mathbf{p}) \cdot \Delta\mathbf{p} = 0. \quad (3)$$

$\Delta\mathbf{p}$  を  $\nabla_{\mathbf{p}} C$  の方向になるように制約する事はつまりあるスカラー値  $\lambda$  で下記のように示される事を意味します。

$$\Delta\mathbf{p} = \lambda \nabla_{\mathbf{p}} C(\mathbf{p}). \quad (4)$$

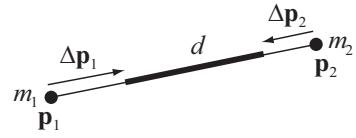


図 2 下記制約条件の射影は  $ii(\mathbf{p}_1, \mathbf{p}_2) (|\mathbf{p}_1 - \mathbf{p}_2| - d)^2$  となります。修正量  $\Delta\mathbf{p}_i$  は、質量の逆数で重み付けされます  $ii(1/m_i)$

式(4)を式(3)に代入し  $\lambda$  を解き、再度式(4)に代入することで最終的な  $\Delta\mathbf{p}$  に関する式が求まります。

$$\Delta\mathbf{p} = -\frac{C(\mathbf{p})}{|\nabla_{\mathbf{p}} C(\mathbf{p})|^2} \nabla_{\mathbf{p}} C(\mathbf{p}) \quad (5)$$

これは、単一の制約条件を構成する非線形形式の繰り返し解法である通常のニュートン・ラフソン法のステップとなります。個々の位置である  $\mathbf{p}_i$  の修正値として、下記を用います。

$$\Delta\mathbf{p}_i = -s \nabla_{\mathbf{p}_i} C(\mathbf{p}_1, \dots, \mathbf{p}_n), \quad (6)$$

ここでスケール要素として

$$s = \frac{C(\mathbf{p}_1, \dots, \mathbf{p}_n)}{\sum_j |\nabla_{\mathbf{p}_j} C(\mathbf{p}_1, \dots, \mathbf{p}_n)|^2} \quad (7)$$

が、全ての頂点で同じ値として用いられます。もし、頂点ごとに異なる質量をもたせる場合、修正量  $\Delta\mathbf{p}_i$  を質量の逆数  $w_i = 1/m_i$  で重み付けします。無限の質量を持つ場合、つまり、望むなら例えば  $w_i = 0$  の場合に動かなくすることも可能です。したがって式(4)は下記のように置き換え可能です。

$$\Delta\mathbf{p}_i = \lambda w_i \nabla_{\mathbf{p}_i} C(\mathbf{p}) \text{ yielding}$$

$$s = \frac{C(\mathbf{p}_1, \dots, \mathbf{p}_n)}{\sum_j w_j |\nabla_{\mathbf{p}_j} C(\mathbf{p}_1, \dots, \mathbf{p}_n)|^2} \quad (8)$$

スケール要素を考慮にいれると最終的な修正量はこうなります。

$$\Delta\mathbf{p}_i = -s w_i \nabla_{\mathbf{p}_i} C(\mathbf{p}_1, \dots, \mathbf{p}_n). \quad (9)$$

例を挙げると、距離の制約条件の関数

$$C(\mathbf{p}_1, \mathbf{p}_2) = |\mathbf{p}_1 - \mathbf{p}_2| - d \text{ となります。頂点ごとの勾配は、 } e \nabla_{\mathbf{p}_1} C(\mathbf{p}_1, \mathbf{p}_2) = \mathbf{n} \text{ と } \nabla_{\mathbf{p}_2} C(\mathbf{p}_1, \mathbf{p}_2) =$$

$-\mathbf{n}$  で  $\mathbf{n} = \frac{\mathbf{p}_1 - \mathbf{p}_2}{|\mathbf{p}_1 - \mathbf{p}_2|}$  となり、スケール要素  $s$  は  $s = \frac{w_1 + w_2}{|\mathbf{p}_1 - \mathbf{p}_2|}$  となり最終的な修正量は下記となります。

$$\Delta\mathbf{p}_1 = -\frac{w_1}{w_1 + w_2} (|\mathbf{p}_1 - \mathbf{p}_2| - d) \frac{\mathbf{p}_1 - \mathbf{p}_2}{|\mathbf{p}_1 - \mathbf{p}_2|} \quad (10)$$

$$\Delta\mathbf{p}_2 = +\frac{w_2}{w_1 + w_2} (|\mathbf{p}_1 - \mathbf{p}_2| - d) \frac{\mathbf{p}_1 - \mathbf{p}_2}{|\mathbf{p}_1 - \mathbf{p}_2|} \quad (11)$$

これらは、距離の制約条件の射影のために [Jak01] で提案されていた式です。(図2を参照ください)。これらは、一般的な制約条件の射影手法の特別なケースが現れています。

これまで、制約条件タイプと硬さ  $k$  に関して考慮されていませんでした。タイプを考慮することは、ごく簡単なことです。もし、タイプが等式の場合、常に射影を実行します。もし、タイプが不等式の場合、射影は、 $C(\mathbf{p}_1, \dots, \mathbf{p}_n) < 0$  の時のみ実行します。様々に包含される硬さパラメータがあります。最も単純な変形は、修正量  $\Delta\mathbf{p}$  に  $k \in [0..1]$  を乗算することです。しかしながら、ソルバーの中の繰り返しループで乗算することで  $k$  の効果は非線形となります。単一の距離の制約条件の残っている誤り値は、 $n_s$  ソルバーの繰り返し後に  $\Delta\mathbf{p}(1-k)^{n_s}$  となります。線形の関係を得るために、修正量に  $k$  を直接乗算するのではなく、 $k' = 1 - (1-k)^{1/n_s}$  を乗算します。この変換で、望み通りエラー修正量は  $\Delta\mathbf{p}(1-k')^{n_s} = \Delta\mathbf{p}(1-k)$  となり、それ故  $k$  に対して線形になります。 $n_s$  とは独立します。しかしながら、結果として得られる素材の硬さはまだシミュレーションのタイムステップとは独立したままであります。典型的な実時間環境では、固定のタイムステップを用い、その場合この依存は問題となりません。

### 3.4. 衝突検出と応答

位置ベースのアプローチの1つの利点は、シンプルに衝突応答を実現できることです。シミュレーションアルゴリズムの行 Ö8) では、衝突の制約条件  $\bar{u}_{coll}$  が生成されます。最初の制約条件  $\bar{u}$  がオブジェクト表現によって与えられシミュレーションの間中固定な一方で、追加の制約条件  $\bar{u}_{coll}$  がそれぞれのタイムステップでスクラッチから生成されます。衝突の制約条件  $\bar{u}_{coll}$  の数は様々で、衝突している頂点の数によって依存します。連続的、静的な衝突の両方を扱うことが出来ます。連続的な衝突を扱うために、我々はそれぞれの頂点  $i$  と  $i-1$  のレイとの交差判定を行います。もし、このレイがオブジェクトと交差するなら、エンタリーポイント  $c$  とサーフェイスの法線  $\mathbf{n}_c$  をこの位置で計算します。不等式の制約条件  $\bar{u}(\mathbf{p}) ((\mathbf{p} - c) \cdot n_c \text{ と硬さ } \bar{u} (\text{Ö} \text{は、制約条件のリストに追加されます。もしレイ } i-1 \text{ が完全にオブジェクトの中にある場合、連続的な衝突検出はどこかで失敗しています。この場合、我々は静的な衝突処理に戻ります。})$  に最も近いサーフェイス上の点  $s$  とこの位置でのサーフェイスの法線  $\mathbf{n}_s$  を計算します。不等式関数  $\bar{u}(\mathbf{p}) ((\mathbf{p} - \mathbf{q}_s) \cdot \mathbf{n}_s$  の制約条件と硬さ  $k = 1$  が、制約条件のリストに追加されます。衝突の制約条件は、ソルバーのループの外で実行されます。これは、シミュレーションをさらに高速化させます。しかしながら、固定の衝突制約条件のセットでソルバーが動く場合に衝突を取り逃すケースが有ります。幸運なことに、我々の経験ではこのアーティファクトは無視できるものです。

摩擦とその回復は、アルゴリズムのステップ (16) で衝突頂点の速度を操作することで扱うことが可能です。衝突の制約条件が生成した、それぞれの頂点の速度は、衝突の法線と垂直にダンピングされ、衝突法線の向きを反映します。

これまで議論してきた衝突応答は、静的なオブジェクトとの衝突のみ正確です。なぜなら衝突したペアの相手に力積が転送されないからです。2つの動的な衝突オブジェクトの

正確な応答は、我々のシミュレータで両方のオブジェクトをシミュレートすることで実現可能です。つまり  $N$  頂点と  $M$  制約条件を我々のアルゴリズムに入力することで2つかそれ以上の独立したオブジェクトをシンプルに表現することが可能となります。従ってもし、オブジェクトの頂点  $\mathbf{q}$  が、別のオブジェクトの三角形  $i-1, \mathbf{p}_2, \mathbf{p}_3$  の中へ動いた場合、不等式の制約条件を制約条件関数  $\bar{u}(\mathbf{q}, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3) ((\mathbf{q} - \mathbf{p}_1) \cdot ((\mathbf{p}_2 - i) - (\mathbf{p}_3 - i))$  に挿入します。この式が、頂点  $i$  を三角形の正確な向きへ維持させます。この制約条件関数は剛体の状態には依存しませんので、正確に並進運動量と回転運動量を保存します。衝突検出は、4つの頂点がレイ  $i-1, i$  で表現されるため僅かに複雑になります。それ故、移動している三角形に対する移動する点の衝突が検出される必要があります。(衣服との事故衝突のセクションを参照ください)

### 3.5. 減衰

新しい位置の推定に速度が使われる前にシミュレーションアルゴリズムの行 (6) で速度が減衰されます。減衰のどんな形であろうと利用可能で文献([NMK\*05]を参照ください)の中で多くの減衰手法が提案されています。ここで、興味深い特徴をいくつか持った新しい手法を提案します。

- (1)  $\mathbf{x}_{cm} = (\sum_i \mathbf{x}_i m_i) / (\sum_i m_i)$
- (2)  $\mathbf{v}_{cm} = (\sum_i \mathbf{v}_i m_i) / (\sum_i m_i)$
- (3)  $\mathbf{L} = \sum_i \mathbf{r}_i \times (m_i \mathbf{v}_i)$
- (4)  $\mathbf{I} = \sum_i \tilde{\mathbf{r}}_i \tilde{\mathbf{r}}_i^T m_i$
- (5)  $\boldsymbol{\omega} = \mathbf{I}^{-1} \mathbf{L}$
- (6) **forall** vertices  $i$
- (7)  $\Delta\mathbf{v}_i = \mathbf{v}_{cm} + \boldsymbol{\omega} \times \mathbf{r}_i - \mathbf{v}_i$
- (8)  $\mathbf{v}_i \leftarrow \mathbf{v}_i + k_{damping} \Delta\mathbf{v}_i$
- (9) **endfor**

ここで  $\mathbf{r}_i = \mathbf{x}_i - \mathbf{x}_{cm}$ ,  $\tilde{\mathbf{r}}_i$  は、3行3列の行列となり、 $\tilde{\mathbf{r}}_i \mathbf{v} = \mathbf{r}_i \times \mathbf{v}$  と  $k_{damping} \in [0..1]$  は、減衰係数となります。行 (1)-(5) で、系全体のグローバルな並進速度  $\mathbf{x}_{cm}$  と角速度  $\boldsymbol{\omega}$  を計算します。行(6)-(9) でのみグローバルの動き  $\mathbf{v}_{cm} + \boldsymbol{\omega} \times \mathbf{r}_i$  から求めた速度  $\mathbf{v}_i$  の個々の差  $\Delta\mathbf{v}_i$  のみを減衰させます。したがって、極端なケースは  $k_{damping} = 1$  でグローバルな動きのみが生き残り頂点集合は剛体のようにふるまいます。任意の  $k_{damping}$  の値の場合、速度はグローバルに減衰されますが頂点のグローバルな動きの影響はありません。

### 3.6. 連結

位置ベースの手法の場合、静的あるいは運動しているオブジェクトに頂点を連結することは極めてシンプルに実現できます。頂点の位置を単純に静的な目標位置に設定するか毎タイムステップに運動するオブジェクトの位置に更新するだけです。この頂点を含む他の制約条件を確実に満たすために、質量の逆数  $w_i$  は 0 に設定します。

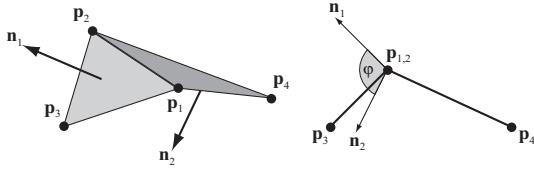


図 4: 曲げた時の抵抗のために、制約条件関数  $\bar{u}(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4) = \arccos(\mathbf{n}_1 \cdot \mathbf{n}_2) - \varphi_0$  が使われます。実際の二面角  $\varphi$  は、2つの三角形の法線の間の角度を測定します。

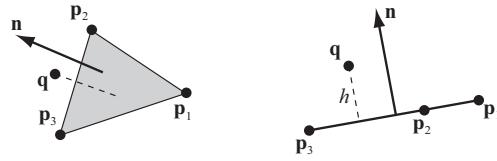


図 5: 制約条件関数  $\bar{u}(\mathbf{q}, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3)$  ( $(\mathbf{q} - \mathbf{p}_1) \cdot \mathbf{n} - \bar{u}$  は、が三角形  $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3$  の布の厚み  $h$  だけ上にあることを保証します。

#### 4. クロス・シミュレーション

我々は、位置ベース物理フレームワークをビデオゲーム向きのリアルタイム・クロス・シミュレーションへと実装してきました。このセクションでは、クロス固有の問題について議論します。それゆえ前のセクションで紹介した一般的な概念の具体的な例を示します。

##### 4.1. 布の表現

我々のクロスシミュレーションは、任意の三角形メッシュを用いることができます。入力メッシュに課する唯一の制限は、メッシュがマニホールドであることで、これは、すべての辺が最大でも2つの三角形で共有されていることです。メッシュのそれぞれのノードがシミュレーションされる頂点となります。ユーザーは、体積当たりの質量 [ $kg/m^2$ ] として提供される密度  $\rho$  を提供します。頂点の質量は、それぞれの隣接三角形の質量の1/3の合計として設定されます。それぞれの辺で、下記の制約条件関数を用いた伸び制約条件を生成します。

$$C_{stretch}(\mathbf{p}_1, \mathbf{p}_2) = |\mathbf{p}_1 - \mathbf{p}_2| - l_0,$$

堅さ  $k_{stretch}$  と等式のタイプです。スカラー値  $l_0$  は、辺の初期長で、 $k_{stretch}$  は、ユーザー指定のグローバルパラメータです。これは、布の伸びやすさを定義しています。

隣接三角形  $(\mathbf{p}_1, \mathbf{p}_3, \mathbf{p}_2)$  と  $(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_4)$  のそれぞれのペアに対して、曲げ制約条件を下記制約条件関数を定義します。

$$C_{bend}(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4) = \text{acos} \left( \frac{(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_3 - \mathbf{p}_1)}{|(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_3 - \mathbf{p}_1)|} \cdot \frac{(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_4 - \mathbf{p}_1)}{|(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_4 - \mathbf{p}_1)|} \right) - \varphi_0,$$

$k_{bend}$  は硬さで、タイプは等式です。スカラー値  $\varphi_0$  は、二つの三角形の間の初期二面角で  $k_{bend}$  は、布の曲げやすさを定義するユーザー指定のグローバルパラメーターです。（図4 参照ください）頂点  $\mathbf{p}_3$  と  $\mathbf{p}_4$  の間の距離の制約条件に加えてこの曲げ項を追加するか、あるいは[GHDS03] によって提案された曲げ項を追加する利点は、引き伸ばし量とは独立していることです。これは項が辺の長さから独立しているからです。このようにして、例えば、ユーザーは低い伸びにくさを持っているが高い曲げにくさを持つ特別な布を作ることができます。（図3を参照ください）

式 (10) と (11) は、伸び制約条件のための射影として定義されます。付録A で、曲げ制約条件の射影の式を導出します。

#### 4.2. 剛体との衝突

剛体との衝突を扱うために、セクション3.4で記述した処理を行います。双方向の応答を行うために、力積  $m_i \Delta \mathbf{p}_i / \Delta t$  を剛体との接觸点へ適用します。それぞれの時間で頂点  $i$  は、剛体との衝突のために投影されます。衝突判定のために布の頂点を使うだけでは十分ではありません。なぜなら小さな剛体は大きな布の三角形をすり抜けることがあるからです。それゆえ、布の三角形と接する剛体の角の凸面の衝突もテストされます。

#### 4.3. 自己衝突

すべての三角形が同じサイズだと仮定し、頂点と三角形の衝突を見つけるために空間ハッシュングを使います。[THM\*03] 頂点  $\mathbf{q}$  が三角形  $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3$  の中を動くとき、我々は下記制約条件関数を用いることができます。

$$C(\mathbf{q}, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3) = (\mathbf{q} - \mathbf{p}_1) \cdot \frac{(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_3 - \mathbf{p}_1)}{|(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_3 - \mathbf{p}_1)|} - h, \quad (12)$$

ここで  $h$  は布の厚さです。（図5を参照ください）もし三角形の法線に対して下側に頂点が入った場合、制約条件の関数は下記のようになります。

$$C(\mathbf{q}, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3) = (\mathbf{q} - \mathbf{p}_1) \cdot \frac{(\mathbf{p}_3 - \mathbf{p}_1) \times (\mathbf{p}_2 - \mathbf{p}_1)}{|(\mathbf{p}_3 - \mathbf{p}_1) \times (\mathbf{p}_2 - \mathbf{p}_1)|} - h \quad (13)$$

頂点がオリジナルな側を維持します。これらの制約条件の射影により、並進運動および角運動量の保存が行われ、この内部の処理により布の自己衝突が正しく行われます。図6は、布が自己衝突した際の状態です。もし布が折りたたまれた状態に入った場合、連続的に衝突をテストすることは非効率です。ですから [BWK03] によって提案されたような手法が適用されます。

#### 4.4. 布風船

閉じた三角形メッシュの場合、メッシュの内側に圧力を加えるようなモデルは簡単に実現できます。（図7をご覧下さい。）我々は、メッシュの全ての頂点  $N$  に関わる等式の制約条件を加えます。

$$C(\mathbf{p}_1, \dots, \mathbf{p}_N) = \left( \sum_{i=1}^{n_{\text{triangles}}} (\mathbf{p}_{t_1^i} \times \mathbf{p}_{t_2^i}) \cdot \mathbf{p}_{t_3^i} \right) - k_{\text{pressure}} V_0 \quad (14)$$

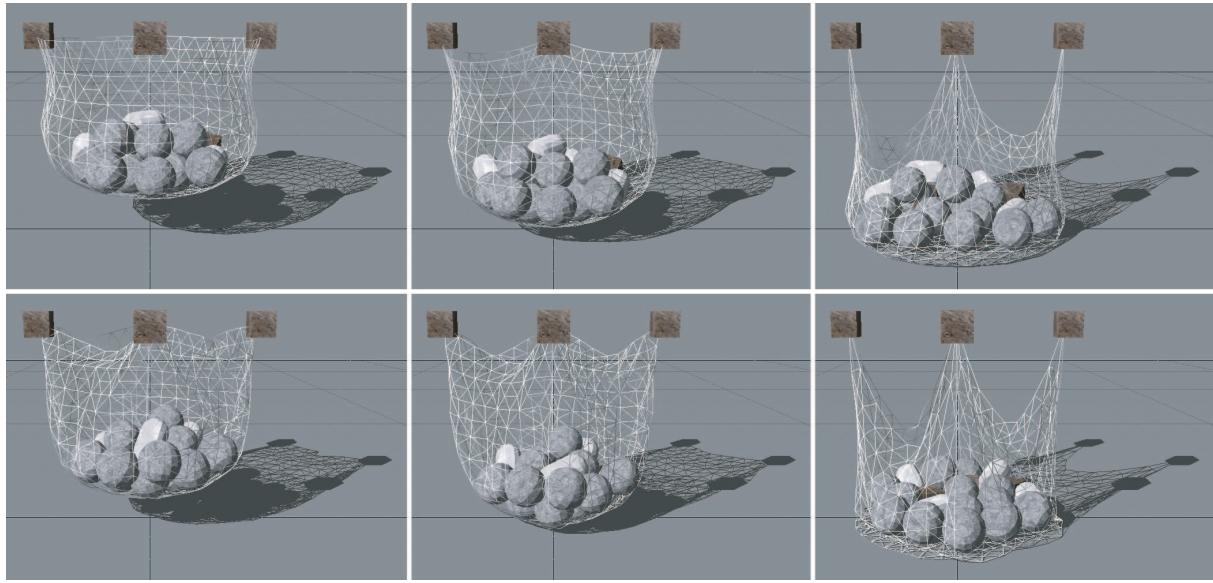


図3: 提案する曲げ項を用いたもの、曲げと伸びは独立したパラメータとなります。上の列が示すのは  $(k_{stretching}, k_{bending}) = (1,1)$ ,  $(1,2)$  と  $(100,1)$  です。下の列が示すのは  $(k_{stretching}, k_{bending}) = (1,0)$ ,  $(1,2)$  と  $(100,0)$  です。

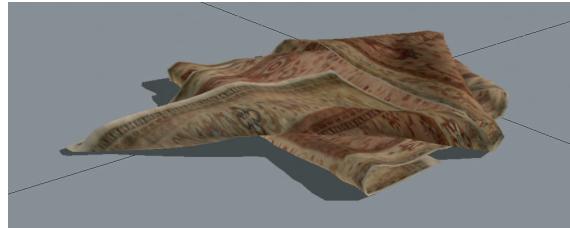


図6: この折りたたまれた状態は、自己衝突と応答の状況を示しています。

そして堅さ  $k = 1$  を制約条件集合に適用します。ここで、 $t_1^i, t_2^i$  と  $t_3^i$  は、三角形  $i$  に属する3つの頂点インデックスです。合計を計算すると閉じたメッシュの実際の体積を計算することができますオリジナルの体積  $V_0$  と比較することで圧力超過項  $k_{pressure}$  を求めます。この制約条件の関数は、下記勾配となります。

$$\nabla_{\mathbf{p}} C = \sum_{j: t_1^j = i} (\mathbf{p}_{t_2^j} \times \mathbf{p}_{t_3^j}) + \sum_{j: t_2^j = i} (\mathbf{p}_{t_3^j} \times \mathbf{p}_{t_1^j}) + \sum_{j: t_3^j = i} (\mathbf{p}_{t_1^j} \times \mathbf{p}_{t_2^j}) \quad (15)$$

この勾配は、式(7)で与えられるスケール項でスケールされ、式(9)に沿った質量でウエイトされ、最終的な射影オフセット  $\Delta \mathbf{p}_i$  となります。

## 5. 結果

我々は、手法を、ゲームのような物理シミュレーション環境であるRocket [Rat04]に統合しました。様々な実験



図7: キャラクターの中に圧力を加えたシミュレーション

が、提案手法の特徴とパフォーマンスを分析するために実施されました。このセクションで提示されるすべてのテストケースでは、Pentium 4, 3 GHz の PC でテストされました。

**独立した曲げ制約条件と伸び制約条件。**我々の曲げ項は、辺の長さではなく隣接三角形の2面角にのみ依存しています。ですから曲げと伸び抵抗は独立して選ぶことができます。図3は、様々な伸びやすさを持つ布の袋を示しています。最初のものは曲げ抵抗を有効にしたものでその後のものは無効にしたものです。上の段が示すのは、曲げ抵抗は伸びの抵抗に影響しないという事です。

**双方向のインタラクションを持った連結。**我々の手法は、一方向および双方向の連結制約条件をシミュレーションすることができます。図8の布の帯は、1方向の制約条件で上の静的剛体と連結されています。加えて、帯と底の剛体との間の双方向の相互作用が有効となっています。この構成で、帯が揺れたり、ねじれたりする現実的な見た目の結果が得られました。6つの剛体と3切れの布のシミュレーションのシーンは380fpsでレンダリングされています。



図 8: 布の帯が上の静的な剛体と連結していて、下の剛体とは双方の制約条件で連結されています。

**実時間での自己衝突。**図6で示す布切れは、1364頂点と2562の三角形で構成されています。シミュレーションは平均30fpsで動きますし、この中には自己衝突の検出を含む衝突応答とレンダリングを含みます。摩擦の効果は図9で示されており、布の同じ部分が、回転するドラムの中でひっくり返っています。

**引き裂きと安定性。**図10は、4264頂点と8262の三角形で構成される布が連結された資格で破られ最終的に投げられた球によって2つに分断されるシーンを示しています。シーンは平均47fpsでシミュレーション・レンダリングされます。引き裂きは、簡単なプロセスでシミュレーションされます:辺の伸長が特定の閾値を超えた場合はいつでも辺の隣接頂点の1つを選ぶことができます。その後、辺の方向に垂直な頂点を通る分割平面を通して頂点を分割します。分割平面の上側の全ての頂点はオリジナルの頂点側に割り当てられ、下側は複製されたものとして割り当てられます。我々の手法は、図1に示されるように [ITF04] に影響を受けた極端な状況でさえ安定しています。膨らませたキャラクターモデルは、回転する歯車を通して複数の制約条件で圧迫され、衝突と自己衝突が单一の布の頂点に働きます。

**複雑なシミュレーションのシナリオ。**提案手法は、複雑なシミュレーション環境特に適しています。(図12を参照ください。)アニメーションするキャラクターとの高価な相互作用や幾何的に複雑なゲームのレベルにも関わらず、複数の布のシミュレーションとレンダリングがインタラクティブな速度で可能です。

## 6. 結論

我々は、位置ベース物理フレームワークを提案しました。これは、制約条件閾数を通じて定式化された一般的な制約条件を扱うことができます。この位置ベースの手法を用いることで、シミュレーションの間オブジェクトを直接的に操作することが可能となります。この手法は、衝突の応答を極めて単純にし、連結制約や陽的な積分を可能にします。そしてアニメーションされたシーンの直接的で即時の制御を可能とします。

我々は、このフレームワークを用い堅牢な布のシミュレーションの実装を行いました。このシミュレーションは、布と剛体との双方のインタラクションや布の自己衝突などの応答や布に動的な剛体に連結した布のシミュレーションが可能となります。

## 7. 今後の課題

この論文で扱わなかったトピックとして剛体のシミュレーションがあります。しかしながら、提案手法は、簡単に剛体のオブジェクトを同様に簡単に扱えるように拡張できます。通常の剛体のソルバが典型的に行うように衝突の解像度での並進および回転運動の力積の計算を行う代わりに、移動と回転は衝突点で剛体に対して適用されソルバが終了した段階で修正されます。



図9:衝突の影響、自己衝突と摩擦、布が回転するドラムの中でひっくり返っています。

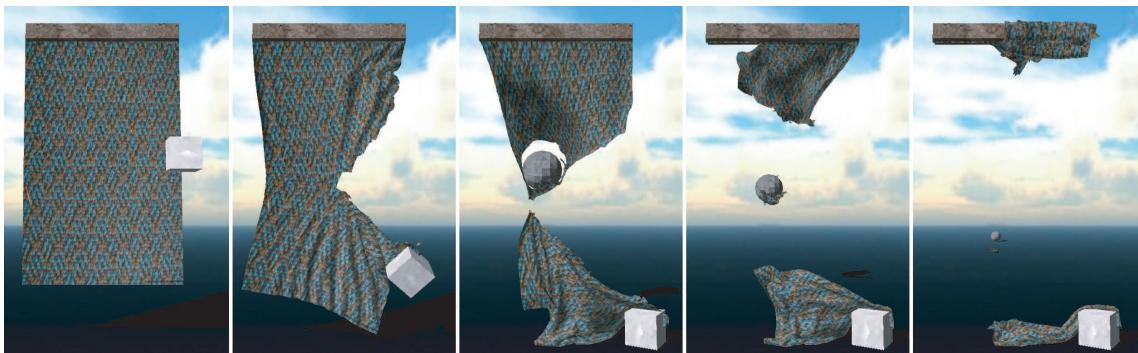


図10:布が連結された四角で引きちぎられ、投げられた球で2つに引き裂かれる様子です。

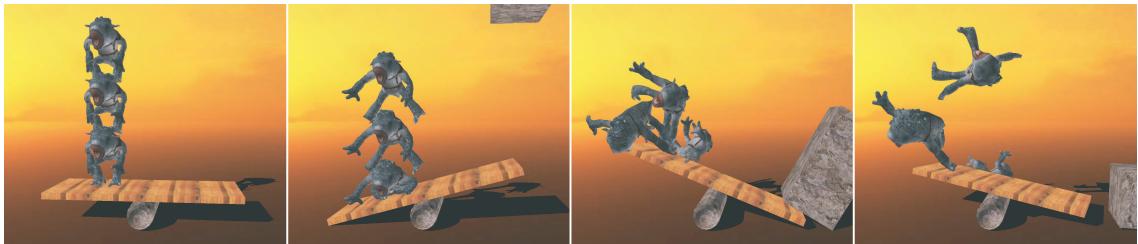


図11:3つの膨らませたキャラクターは、複数の衝突判定と自己衝突の影響を受けます。



図12:布とアニメーションするゲームキャラクター間の大規模な相互作用(左)、幾何的に複雑なゲームレベル(中)、多数の植物のシミュレーション(右)

## 参考文献

- [BFA02] BRIDSON R., FEDIKIW R., ANDERSON J.: Robust treatment of collisions, contact and friction for cloth animation. *Proceedings of ACM Siggraph* (2002), 594–603.
- [BMF03] BRIDSON R., MARINO S., FEDIKIW R.: Simulation of clothing with folds and wrinkles. In *ACM SIGGRAPH Symposium on Computer Animation* (2003), pp. 28–36.
- [BW98] BARAFF D., WITKIN A.: Large steps in cloth simulation. *Proceedings of ACM Siggraph* (1998), 43–54.
- [BWK03] BARAFF D., WITKIN A., KASS M.: Untangling cloth. In *Proceedings of the ACM SIGGRAPH* (2003), pp. 862–870.
- [CBP05] CLAVET S., BEAUDOIN P., POULIN P.: Particle-based viscoelastic fluid simulation. *Proceedings of the ACM SIGGRAPH Symposium on Computer Animation* (2005), 219–228.
- [DSB99] DESBRUN M., SCHRODER P., BARR A.: Interactive animation of structured deformable objects. In *Proceedings of Graphics Interface '99* (1999), pp. 1–8.
- [Fau98] FAURE F.: Interactive solid animation using linearized displacement constraints. In *Eurographics Workshop on Computer Animation and Simulation (EGCAS)* (1998), pp. 61–72.
- [Fed05] FEDOR M.: Fast character animation using particle dynamics. *Proceedings of International Conference on Graphics, Vision and Image Processing, GVIP05* (2005).
- [GHDS03] GRINSPUN E., HIRANI A., DESBRUN M., SCHRODER P.: Discrete shells. In *Proceedings of the ACM SIGGRAPH Symposium on Computer Animation* (2003).
- [ITF04] IRVING G., TERAN J., FEDIKIW R.: Invertible finite elements for robust simulation of large deformation. In *Proceedings of the ACM SIGGRAPH Symposium on Computer Animation* (2004), pp. 131–140.
- [Jak01] JAKOBSEN T.: Advanced character physics Ú the fysix engine. [www.gamasutra.com](http://www.gamasutra.com) (2001).
- [MHTG05] MÜLLER M., HEIDELBERGER B., TESCHNER M., GROSS M.: Meshless deformations based on shape matching. *Proceedings of ACM Siggraph* (2005), 471–478.
- [NMK\*05] NEALEN A., MÜLLER M., KEISER R., BOXERMAN E., CARLSON M.: Physically based deformable models in computer graphics. *Eurographics 2005 state of the art report* (2005).
- [Pro95] PROVOT X.: Deformation constraints in a mass-spring model to describe rigid cloth behavior. *Proceedings of Graphics Interface* (1995), 147–154.
- [Rat04] RATCLIFF J.: Rocket - a viewer for real-time physics simulations. [www.physicstools.org](http://www.physicstools.org) (2004).
- [THM\*03] TESCHNER M., HEIDELBERGER B., MÜLLER M., POMERANERTS D., GROSS M.: Optimized spatial hashing for collision detection of deformable objects. *Proc. Vision, Modeling, Visualization VMV 2003* (2003), 47–54.
- [THMG04] TESCHNER M., HEIDELBERGER B., MÜLLER M., GROSS M.: A versatile and robust model for geometrically complex deformable solids. *Proceedings of Computer Graphics International (CGI)* (2004), 312–319.
- [VCMT95] VOLINO P., COURCHESNE M., MAGNENAT-THALMANN N.: Versatile and efficient techniques for simulating cloth and other deformable objects. *Proceedings of ACM Siggraph* (1995), 137–144.

補足 A:

## 正規化された外積の勾配

制約条件関数は、たびたび正規化された外積を含みます。射影の修正を求めるために、制約条件の関数勾配が必要となります。それゆえに、両方の議論に関する正規化された外積の勾配を知っておくことが有益となります。与えられた正規化済外積の値  $\mathbf{n} = \frac{\mathbf{p}_1 \times \mathbf{p}_2}{\|\mathbf{p}_1 \times \mathbf{p}_2\|}$ , が与えられると、最初のベクトルに相当する導関数は下記となります。

$$\frac{\partial \mathbf{n}}{\partial \mathbf{p}_1} = \begin{bmatrix} \frac{\partial n_x}{\partial p_{1x}} & \frac{\partial n_x}{\partial p_{1y}} & \frac{\partial n_x}{\partial p_{1z}} \\ \frac{\partial n_y}{\partial p_{1x}} & \frac{\partial n_y}{\partial p_{1y}} & \frac{\partial n_y}{\partial p_{1z}} \\ \frac{\partial n_z}{\partial p_{1x}} & \frac{\partial n_z}{\partial p_{1y}} & \frac{\partial n_z}{\partial p_{1z}} \end{bmatrix} \quad (16)$$

$$= \frac{1}{\|\mathbf{p}_1 \times \mathbf{p}_2\|} \left( \begin{bmatrix} 0 & p_{2z} & -p_{2y} \\ -p_{2z} & 0 & p_{2x} \\ p_{2y} & -p_{2x} & 0 \end{bmatrix} + \mathbf{n}(\mathbf{n} \times \mathbf{p}_2)^T \right) \quad (17)$$

短くまとめて下記の式となります。

$$\frac{\partial \mathbf{n}}{\partial \mathbf{p}_1} = \frac{1}{\|\mathbf{p}_1 \times \mathbf{p}_2\|} (-\tilde{\mathbf{p}}_2 + \mathbf{n}(\mathbf{n} \times \mathbf{p}_2)^T) \quad (18)$$

$$\frac{\partial \mathbf{n}}{\partial \mathbf{p}_2} = -\frac{1}{\|\mathbf{p}_1 \times \mathbf{p}_2\|} (-\tilde{\mathbf{p}}_1 + \mathbf{n}(\mathbf{n} \times \mathbf{p}_1)^T) \quad (19)$$

(20)

ここで  $\tilde{\mathbf{p}}$  は、 $\tilde{\mathbf{p}}\mathbf{x} = \mathbf{p} \times \mathbf{x}$  となる行列です。

## 曲げ制約条件の射影

The constraint function for bending is  $C = \arccos(d) - \varphi_0$ , where  $d = \mathbf{n}_1 \cdot \mathbf{n}_2 = \mathbf{n}_1^T \mathbf{n}_2$ . Without loss of generality we set  $\mathbf{p}_1 = \mathbf{0}$  and get for the normals  $\mathbf{n}_1 = \frac{\mathbf{p}_2 \times \mathbf{p}_3}{\|\mathbf{p}_2 \times \mathbf{p}_3\|}$  and  $\mathbf{n}_2 = \frac{\mathbf{p}_2 \times \mathbf{p}_4}{\|\mathbf{p}_2 \times \mathbf{p}_4\|}$ . With  $\frac{d}{dx} \arccos(x) = -\frac{1}{\sqrt{1-x^2}}$  we get the following gradients:

$$\nabla_{\mathbf{p}_3} C = -\frac{1}{\sqrt{1-d^2}} \left( \left( \frac{\partial \mathbf{n}_1}{\partial \mathbf{p}_3} \right)^T \mathbf{n}_2 \right) \quad (21)$$

$$\nabla_{\mathbf{p}_4} C = -\frac{1}{\sqrt{1-d^2}} \left( \left( \frac{\partial \mathbf{n}_2}{\partial \mathbf{p}_4} \right)^T \mathbf{n}_1 \right) \quad (22)$$

$$\nabla_{\mathbf{p}_2} C = -\frac{1}{\sqrt{1-d^2}} \left( \left( \frac{\partial \mathbf{n}_1}{\partial \mathbf{p}_2} \right)^T \mathbf{n}_2 + \left( \frac{\partial \mathbf{n}_2}{\partial \mathbf{p}_2} \right)^T \mathbf{n}_1 \right) \quad (23)$$

$$\nabla_{\mathbf{p}_1} C = -\nabla_{\mathbf{p}_2} C - \nabla_{\mathbf{p}_3} C - \nabla_{\mathbf{p}_4} C \quad (24)$$

Using the gradients of normalized cross products, first compute

$$\mathbf{q}_3 = \frac{\mathbf{p}_2 \times \mathbf{n}_2 + (\mathbf{n}_1 \times \mathbf{p}_2)d}{\|\mathbf{p}_2 \times \mathbf{p}_3\|} \quad (25)$$

$$\mathbf{q}_4 = \frac{\mathbf{p}_2 \times \mathbf{n}_1 + (\mathbf{n}_2 \times \mathbf{p}_2)d}{\|\mathbf{p}_2 \times \mathbf{p}_4\|} \quad (26)$$

$$\mathbf{q}_2 = -\frac{\mathbf{p}_3 \times \mathbf{n}_2 + (\mathbf{n}_1 \times \mathbf{p}_3)d}{\|\mathbf{p}_2 \times \mathbf{p}_3\|} - \frac{\mathbf{p}_4 \times \mathbf{n}_1 + (\mathbf{n}_2 \times \mathbf{p}_4)d}{\|\mathbf{p}_2 \times \mathbf{p}_4\|} \quad (27)$$

$$\mathbf{q}_1 = -\mathbf{q}_2 - \mathbf{q}_3 - \mathbf{q}_4 \quad (28)$$

Then the final correction is

$$\Delta \mathbf{p}_i = -\frac{w_i \sqrt{1-d^2} (\arccos(d) - \varphi_0)}{\sum_j w_j |\mathbf{q}_j|^2} \mathbf{q}_i \quad (29)$$