柔軟な物体の能動的な動作の表現手法

-A Method for Rig-less Character Animation-

井尻 敬 [1, 2]　　高山建志 [1]　　横田秀夫 [2]　　五十嵐健夫 [1, 3]

Takashi IJIRI[1, 2], Kenshi TAKAYAMA[1], Hideo YOKOTA[2], and Takeo IGARASHI[1, 3]

1, 東京大学,　2, 理化学研究所　3, JST/ERATO

E-mail: takashi.ijiri @ riken.jp

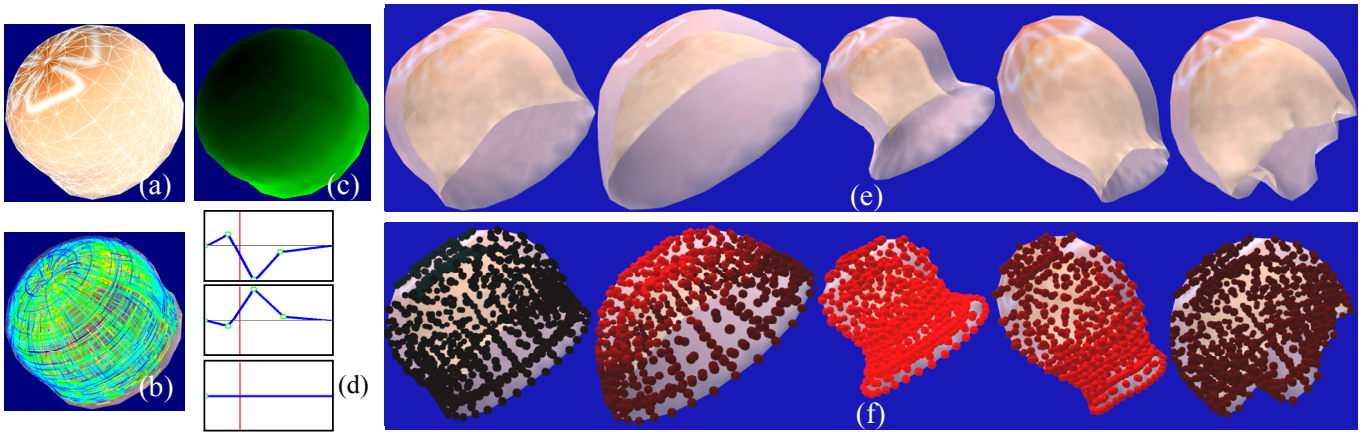

図 1: ***クラゲのアニメーション:*** *ユーザは，四面体モデル(a)内に tensor field(b) と phase-shift field(c) を作り，また，変形グラフ(d)を編集することで，局所部分の変形の様子を指定する．システムは，局所部分の変形をなるべく満たすように大域的な変形を合成する(e)．(f) では，変形の大きい領域が赤く可視化されている．*

概要

本研究では，従来の関節角を制御する手法では表現が難しい，柔軟物体のアニメーションを生成する手法を提案する．柔軟物体において，物体全体の変形は局所部分の変形の積み重ねで引き起こされる．この局所的な変形に駆動される全体像の変形に関して，我々は幾何制約に基づく高速かつロバストなアルゴリズムを提案する．また，効率的に局所変形の様子をデザインするためのユーザインタフェースも提案する．

1. はじめに

クラゲ，タコ，腸や心臓など，骨格構造が不明確な柔軟物体のアニメーションは，埋め込まれた骨格の関節角を指定する従来手法では表現し難い．キーフレームを指定する手法もあるが，多数のキーフレームを指定するのは手間のかかる作業である．また，キーフレーム法では，障害物との接触などの外力に動的に反応する動きはデザインできない．バネモデルや shape matching 法[8][10]など，柔軟物体の動的な変形手法も知られるが，これらは外力による受動的な変形のみを扱うもので，能動的に変形する物体を扱っていない．そこで我々は，従来の関節角やキーフレーム指定では表現が困難であった能動的に動く柔軟物体のための，手続き的な変形手法を提案する．提案手法は，Virtual Reality 空間やゲームなどの，インタラクティブな環境で利用されることを想定している．

軟体動物や臓器などの柔軟な物体が動く際，その大域的な変形は局所部分の変形の積み重ねで引き起こされる，という知見がある．これはつまり、柔軟物体の変形時に，まず物体の各局所部分が何らかの刺激を受け，独立的にその形状を変形させ，その局所的な変形の積み重ねが大域的な動きを駆動するというものである．たとえば，心臓は筋線維が渦状に密集した臓器であるが，心臓が拍動するとき，洞房結節で生じた電気的な刺激が心筋線維に伝わり，各心筋線維がその方向に沿って局所的に収縮する．この局所的な収縮が互いに影響しあいながら，心臓の力強く捻る動きを生む．我々はこの知見に基づいて，柔軟物体の変形フレームワークを設計する．我々のフレームワークでは，ユーザが物体の局所部分の 3 次元的な伸縮を指定すると，システムが指定された局所部分の変形をなるべく維持する様に，大域的な動きを合成する(図 1).

提案フレームワークでは，柔軟物体の動きをデザインするために，局所変形の様子を制御する必要がある．複雑な局所変形パターンを効率的に制御できるよう，我々は 3 個の volumetric field と変形グラフからなるイ

ンタフェースを提案する．ユーザは，tensor field により局所変形の方向を，amplitude field により局所変形の大きさを，phase-shift field により局所変形のタイミングを指定できる．また、ユーザは，変形グラフを編集することで，伸縮のパターンを制御できる．本研究では，3 個の volumetric field を直観的にデザインできるよう，スケッチインタフェースを提供する．

提案フレームワークでは，各アニメーションプロセスにおいて，局所部分の変形がなるべく維持される様に大域的な形状が合成される．我々は，高速かつロバストに大域的な形状を計算するため，shape matching dynamics に基づいたアルゴリズムを提案する．我々のアルゴリズムでは．変形の物理的な正しさを犠牲にする代わりに，前進オイラー法と同程度のコストで，無条件に安定に，大域的な変形を計算することができる．

2. 関連研究

局所変形が引き起こす大域的な変形：生物学関連の分野で，局所変形の積み重ねが引き起こす大域的な形状の変形に関する研究がなされている．Rolland らは[11]，金魚草の花弁の非対称な成長をシミュレートした．彼らは，2 次元の弾性メッシュで花弁を表現し，各エッジの自然長を伸ばすことで、局所的な成長をエミュレートした． Ijiri らは，この手法を 3 次元に拡張した[6]．Combaz らは，有機的な形状をモデリングするための半自動的なインタフェースを提案した[3]．彼らも弾性メッシュでモデルを表現し，ユーザの指定した領域を局所的に伸長させ，緩和計算を行うことで様々な自然形状をモデリングした．しかし，これらの研究は，伸長のみを扱っており，伸縮が繰り返すような動きは扱っていない．また，これらの研究は主に薄い膜状のモデルのみを対象とし，厚みのある volume モデルを扱ってはいない．医療分野において，例えば心臓などの volume モデルの変形を扱う研究もなされているが[1][16]，これらは主に off-line での利用や特殊なハードウェアの利用を想定したものである．

仮想キャラクタのモデリング:パーツを組み立てることで仮想的なキャラクタをデザインできる環境が開発されている．*Juice*[15]や *Modulobe*[4]では，ユーザは，直方体や梁などを組み立て，キャラクタをデザインする．また，各オブジェクト間の接合角を周期的な関数で定義することで，キャラクタの動きもデザインできる．しかし、これらの手法は関節モデルのみに応用が限定される．*Sodaplay* [2]や *Springs World 3D* [5] では，ユーザは棒状のバネを繋ぐことで，キャラクタをモデリングする．さらに，各バネの自然長を周期的な関数により変化させることで，キャラクタの動きを制御できる． これらのシステムでは，潜在的には柔軟物体の表現が可能であるが，バネモデルは不安定なことや，volume を考慮していないため flipping の検出ができないといった問題がある．また、これらのシステムでは，ユーザは、各バネの挙動を一つ一つ定義する必要があり，多数のバネからなるモデルの動きをデザインするのは大変手間のかかる作業である．

柔軟物体の表現: Computer Graphics 分野では，柔軟物体をシミュレートするため様々な物理モデルが提案されている．Nealen らが柔軟物体の為の物理モデルに関するサーベイを行っている[9]．

近年，Shape Matching Dynamics という新しい柔軟物体の表現手法が研究されている．この手法は，エネルギー最小化ではなく幾何制約に基づいており，物理的な正しさには乏しいものの，高速にかつ無条件に安定に変形の計算を行うことが可能である。Müller らは[8]初めて shape matching の考えを柔軟物体の計算に利用した．彼らの手法では，変形前の形状(rest shape)が，変形後の形状(current shape)になるべく合うような平行移動と回転を計算し，rest shape を current shape にぴったり合わせる．そして，ぴったり合うように移動された rest shape の各頂点を，目標の場所(goal position)として，current shape の各頂点を対応する goal position へ向かって引っ張る．Rivers らは[10]，この手法を拡張した lattice shape matching (LSM)を提案した．LSM では，まず，オブジェクトを包む荒い格子状のメッシュを用意し，格子のすべての頂点に小さな局所領域を定義する．隣接する局所領域は互いに重なり合う．そして，各局所領域に shape matching を適用し，結果をブレンドすることで，格子の各頂点の goal position を求める．

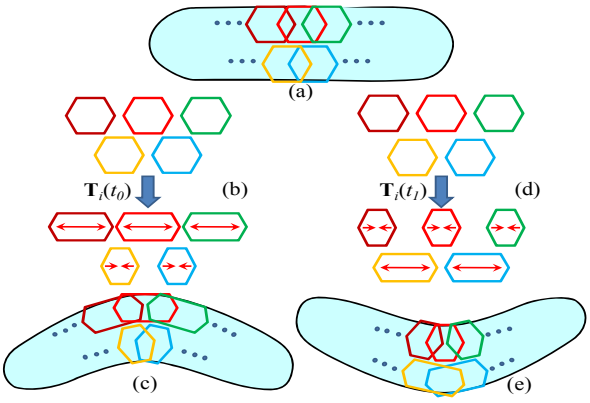

図 2．*変形フレームワークの概略*

3. 変形フレームワーク

図 2 に提案手法の概念図を示す．まず，四面体モデルを用意し，各頂点 $\mathbf{x}_i$ において，その最近傍頂点(1-ring neighborhood)を繋ぐことで，局所領域 $\boldsymbol{N}_i$ を定義する．Rivers らの手法[10]と同様に，隣接する局所領域どう

しは互いに重なり合う．各アニメーションステップにおいて，まずシステムは，各局所領域 N_i を，ユーザに指定された変形関数 $\mathbf{T}_i(t)$に従い変形する．次にシステムは，各局所領域の変形後の形状がなるべく満たされるように，オブジェクト全体を変形する．例えば，もし，物体の上に位置する領域を横方向に引き伸ばし，下の領域を縮めると(図 2b)，物体は下方向に曲がる(図 2c)．逆に，上の領域を縮めて下の領域を伸ばすと(図 2d)，物体は上方向に曲がる(図 2e)．

変形関数 $\mathbf{T}_i(t)$は，局所領域 i と時間 t に依存す関数である．問題を単純にするため，我々は $\mathbf{T}_i(t)$を affine 変換($\mathbf{R}^{3\times3}$)で定義した．ただし，局所的には線形な変形であっても，それが組み合わさることで非線形な，湾曲や捻りなどの変形が表現できる．次章にて，$\mathbf{T}_i(t)$ を効率的に指定するインタフェースについて述べる．

4. 局所変形の指定

局所領域の変形を表す $\mathbf{T}_i(t)\in\mathbf{R}^{3\times3}$ は，場所(局所領域 i)と時間 t に依存する．複雑な局所変形パターンを効率的に制御するため，我々は，tensor field，変形グラフ，amplitude field，phase-shift field を指定するインタフェースを提案する．各 field のデザインのため，スケッチインタフェースを提供する[13]．

Tensor field 指定：一般的に，生物はその内部に筋線維を持ち，筋線維の伸縮が生物の動きを引き起こす．この筋線維の走向を指定するのが tensor field である．我々は tensor field を直行する 3 つの vector field で定義する．四面体モデル内部に tensor field をデザインする際，まずユーザは，モデルに depth 値を塗ることで，depth field をデザインする(図 3a)．ここで，青は値 1.0(最も外側)，赤は値 0.0(最も内側)を表す．塗った色はスムースに補完され(図 3b)，この depth field の傾斜が tensor field の 2 次方向となる(図 3d 中)．我々は，depth 値を効率的に塗ることが出来るように，3 種のブラシモード(single polygon fill mode, flat region fill mode と stroke mode)を実装した[13]．

次にユーザは，depth field の等値面に，tensor field の 1 次方向を表すストロークを描く(図 3c)．システムは，ストロークの接線方向を RBF[14]によりスムースに補完し，tensor field の 1 次方向が得られる(図 3d 左)．最後に，1 次方向と 2 次方向の外積を求めることで tensor field の 3 次方向が得られる(図 3d 右)．

変形グラフ：tensor field を指定したら，次に，tensor field の各方向に伸縮する割合を“変形グラフ”により指定する．図 4 に変形グラフのインタフェースを示す．3 個のグラフは，左から 1 次，2 次，3 次方向の伸縮割合を指定する．横軸は時間[0, T]を表し，縦軸は伸縮の割合[1/A, A]を表す．$T>0$ と $A>1$ は，それぞれ周期と振幅(最大の伸長割合)である．また，グラフの左と右はつながっており，タイムラインが右端まで移動すると，また左端に戻り，伸縮を繰り返す．

変形関数 $\mathbf{T}_i(t)$の計算：時間 t，局所領域 i において，tensor field の 3 方向($\mathbf{d}_1$, $\mathbf{d}_2$, $\mathbf{d}_3$) と変形グラフで指定された伸縮割合 ($c_1(t)$, $c_2(t)$, $c_3(t)$) が得られると，局所領域の変形 $\mathbf{T}_i(t)\in\mathbf{R}^{3\times3}$ は以下のように計算できる．

$$\mathbf{T}_i(t)=\mathbf{D}\,\mathrm{diag}(c_1(t),c_2(t),c_3(t))\,\mathbf{D}^{\mathrm{T}}, \qquad (1)$$

ここで，$\mathbf{D}=(\mathbf{d}_1\ \mathbf{d}_2\ \mathbf{d}_3)\in\mathbf{R}^{3\times3}$．また，$\mathrm{diag}(c_1, c_2, c_3)$ は，c_1, c_2, c_3 を対角要素に持つ 3 × 3 の対角行列である．

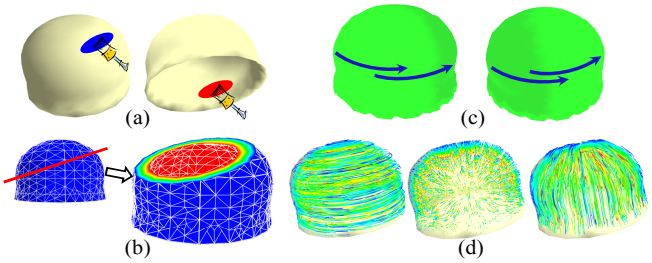

図 3: *Tensor field のデザイン．*

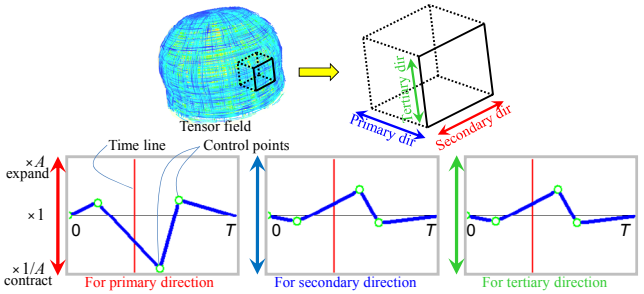

図 4: *変形グラフのインタフェース．*

Tensor field と変形グラフの二つの制御だけでは，すべての局所領域が同じタイミングかつ同じ割合で伸縮する様な動きしかデザインできない．より複雑な局所変形パターンのデザインを可能にするため，我々は，amplitude field と phase-shift field を利用する[7]．

Amplitude field を指定すると，場所ごとに伸縮の大きさを変化させる事ができる．Amplitude field は，各頂点が[-1, 1]の値を持つスムースな field として定義される．この field をデザインするとき，まずユーザは値を選択し，ペンツールでその値をモデル表面に塗る．すると，システムは RBF を利用してモデル内部を補完する．本システムでは，値 1.0 は赤，値 0.0 は黒，値-1.0 は青で可視化されている．変形の計算時に，ある頂点に amplitude field の値 A_s と変形グラフの伸縮割合 c が指定されているとき，システムは，c を次のように変化させる：$c' = c^{As}$．つまり，値 1.0 は伸縮の割合を変化させず，値 0.0 は伸縮を起こさせず，値 - 1.0 は伸長と収縮を逆転させる．この amplitude field は，曲げる

動きをデザインするのに便利である(図 5).

Phase-shift field を指定すると，局所領域の伸縮のタイミングをずらすことができる. Phase-shift field は，各頂点が[0, 1]の値を持つスムースな field として定義される．この field は，amplitude field と同様の塗るインタフェースでデザインできる．本システムでは，値 1.0 は緑, 値 0.0 は黒で可視化される. 変形の計算時に，システムは Phase-shift field の値 p_s に応じて, 変形グラフの時間をずらす：$c'(t) = c(t - p_s/v_p)$．ここで，t は時間，$c(t)$は時間 t における変形グラフの値，v_p は変形の伝わる速さである．この Phase-shift field は，変形が伝播する動きデザインするのに便利ある(図 6).

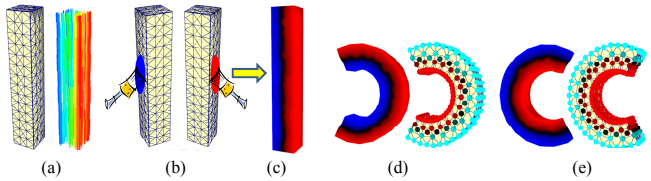

図 5: ***Amplitude field による湾曲.*** *モデルの右側に値 1.0 を，左側に値-1.0 を塗ると，スムースな field が生成でき(a-c)，この field によりモデルが湾曲する．(d) では 1 次方向の伸長が 1.5 倍，(e) では 1 / 1.5 倍である.*

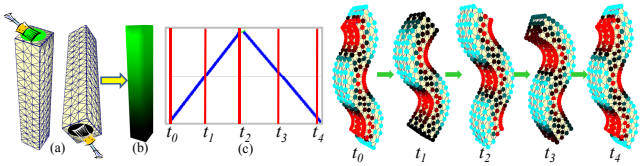

図 6: ***phase-shift field による変形の伝搬.*** *ユーザはモデルの下に値 0.0，上に値 1.0 を塗ることで，スムースな field をデザインする(a, b)．(c)のように 1 次方向の変形グラフを定義すると，変形が伝播する動きが生成される(右).*

5. 変形アルゴリズム

この章では，局所領域の変形から大域的な変形を計算するアルゴリズムを紹介する．提案システムでは，モデルは四面体メッシュで表現され，各頂点 $\mathbf{x}_i$ に，最近傍頂点をつないだ局所領域 N_i が定義されている．隣接する近傍領域は重なり合う[10]．ある時間 t における領域 N_i の変形は，前章で述べた $\mathbf{T}_i(t)$ により決定される．我々は，変形した各局所領域がなるべく維持されるように，モデル全体を変形する.

提案アルゴリズムは，Rivers らの Lattice Shape Matching (LSM) [10]に基づいている．LSM では，元の形状における局所領域と変形された現在の形状における局所領域との shape matching を行うことで，各頂点 $\mathbf{x}_i$ のあるべき場所(goal position) $\mathbf{g}_i$ を計算する. さらに，各頂点を goal position に向かって引っ張ることで，現在形状をなるべく元の形状に近づける. LSM と我々の手法の主な違いは，以下の 2 点である．LSM ではグリッド状の lattice を用いたのに対し，我々は四面体モデルを利用した. また, LSM ではモデルの元の形状を rest shape としたのに対し，我々は，$\mathbf{T}_i(t)$により変形された形状を rest shape として利用した.

Goal position の計算: 四面体メッシュの各頂点 $\mathbf{x}_i$ の質量を m_i とする．頂点 $\mathbf{x}_i$ は，複数の局所領域に含まれるため，shape matching の計算には，修正した質量 $\tilde{m}_i = m_i / |N_i|$を利用する．$|N_i|$は領域 N_i に含まれる頂点数である．各局所領域 N_r について，LSM では，以下の最小化問題を解くことで, 最適な回転行列 $\mathbf{R}_r$ を計算した,

$$\underset{\mathbf{R}_r}{\operatorname{argmin}} \sum_{i \in N_r} \tilde{m}_i \left(\mathbf{R}_r(\mathbf{x}_i^0 - \mathbf{c}_r^0) - (\mathbf{x}_i - \mathbf{c}_r)\right)^2, \tag{2}$$

ここで, $\mathbf{x}_i^0$ と $\mathbf{x}_i$ は変形前と変形後の頂点の位置であり, $\mathbf{c}_r^0$ と $\mathbf{c}_r$ は変形前と変形後の領域 N_r の重心である.

$$\mathbf{c}_r^0 = \frac{1}{M_r}\sum_{i \in N_r} \tilde{m}_i \mathbf{x}_i^0,\ \mathbf{c}_r = \frac{1}{M_r}\sum_{i \in N_r} \tilde{m}_i \mathbf{x}_i,\ M_r = \sum_{i \in N_r} \tilde{m}_i . \tag{3}$$

一方我々は, 元の形状の局所領域 N_r を affine 変換 $\mathbf{T}_r(t)$ で変形し，これを rest shape として利用する.

$$\underset{\mathbf{R}_r}{\operatorname{argmin}} \sum_{i \in N_r} \tilde{m}_i \left(\mathbf{R}_r\mathbf{T}_r(\mathbf{x}_i^0 - \mathbf{c}_r^0) - (\mathbf{x}_i - \mathbf{c}_r)\right)^2 . \tag{4}$$

Rest shape と現在の局所領域を最適にマッチするような回転行列 $\mathbf{R}_r$ は, 以下の行列から拡大縮小成分を取り除くことで得られる.

$$\mathbf{A}_r \equiv \sum_{i \in N_r} \tilde{m}_i (\mathbf{x}_i - \mathbf{c}_r)\,(\mathbf{T}_r(\mathbf{x}_i^0 - \mathbf{c}_r^0))^T \quad \in \mathbf{R}^{3\times 3} . \tag{5}$$

我々は，$\mathbf{A}_r$ を次のように分解することで，回転行列 $\mathbf{R}_r$ を計算する: $\mathbf{A}_r = \mathbf{R}_r\mathbf{S}_r$. ここで，$\mathbf{S}_r = \sqrt{\mathbf{A}_r^T\mathbf{A}_r}$ は，拡大縮小成分を含む対称行列である. 以上により得られた，$\mathbf{R}_r$, $\mathbf{c}_r$,と $\mathbf{c}_r^0$ により，局所領域 N_r に含まれる各頂点 $\mathbf{x}_i$ の goal position が計算できる.

$$\mathbf{x}_i = \mathbf{R}_r\mathbf{T}_r(\mathbf{x}_i^0 - \mathbf{c}_r^0) + \mathbf{c}_r \quad i \in N_r . \tag{6}$$

最後に，全ての局所領域の shape matching の結果をブレンドすることで，すべての頂点の goal position $\mathbf{g}_i$ が求められる.

$$\mathbf{g}_k = \frac{1}{|N_k|}\sum_{i \in N_k} (\mathbf{R}_i\mathbf{T}_i(\mathbf{x}_k^0 - \mathbf{c}_i^0) + \mathbf{c}_i) . \tag{7}$$

硬さの調節: LSM では，局所領域のサイズを変化させることで，オブジェクトの硬さを調節できる．大きな局所領域を利用すると，ある点に加えられた力がより遠くまで届くため，オブジェクトは固くなる．逆に小さな局所領域では，オブジェクトは柔らかくなる．しかし，この硬さの制御手法は我々の場合には利用できない．なぜなら，我々は，局所領域が線形変換 $\mathbf{T}i(t)$ により変形するモデルを利用しているため，局所領域のサイズを大きくすると，大きな領域を線形変換で変

形することになり，サンプリングエラーが大きくなってしまう．大きな領域を複数の線形変換で非線形に変形することも可能ではあるが，これは計算コストが爆発的に高くなるという問題がある．そこで我々は，イテレーションによる硬さの調節手法を提案する．

Goal position の計算時に，我々は，まず現在の形状 $\mathbf{x}_i$ をターゲットとして shape matching を行い，goal position $\mathbf{g}^0{}_i$ を得る．次に，今得られた $\mathbf{g}^0{}_i$ をターゲットとして shape matching を計算することで，新たな goal position $\mathbf{g}^1{}_i$ が得られる．以下，反復的に，$\mathbf{g}^{N-1}{}_i$ から $\mathbf{g}^N{}_i$ が計算でき，得られた $\mathbf{g}^N{}_i$ を goal position として利用する．N はユーザが指定する反復回数である．より大きな N を指定すると，goal shape の形状がより元の形状に近くなるため，結果としてオブジェクトは硬くなる(図 7)．この手法では，計算コストは，N に線形に大きくなる．

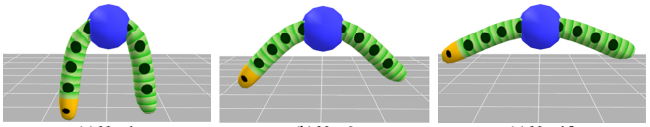

図 7:イテレーションによる硬さ表現. *図は芋虫モデルの中央を空中に固定し，N の値を様々に変えた例.*

運動計算: 上で述べた shape matching により，goal position が得られたら，我々は各頂点の場所 $\mathbf{x}_i$ と速度 $\mathbf{v}_i$ を以下のように更新する[8].

$$\mathbf{v}_i(t+h) = \mathbf{v}_i(t) + \frac{\mathbf{g}_i(t) - \mathbf{x}_i(t)}{h} + h\frac{\mathbf{f}_i^{ext}(t)}{m_i} \tag{8}$$

$$\mathbf{x}_i(t+h) = \mathbf{x}_i(t) + h\mathbf{v}_i(t+h), \tag{9}$$

ここで h はタイムステップであり，$\mathbf{f}_i^{ext}$ は頂点 $\mathbf{x}_i$ に掛る外力である．また、我々は各アニメーションステップにおいて，Muller ら[8]や Revers ら[10]のモデルを利用し速度の減衰を計算した．

6. 結果と考察

図 1 と図 8 はそれぞれ，提案システムを利用してデザインされたクラゲと芋虫のアニメーションの例である．本システムでは泳ぐクラゲや歩く芋虫などの複雑な動きでも，いくつかの field を指定するだけで簡単にデザインすることができる．図 1 の例では，Phase-shift field を指定することにより，変形する領域が徐々に移動していく効果がデザインされている．また，図 8(d)(e)では，amplitude field を指定することにより，蛇のような曲がる動きが生成できる．図 9 に複数のオブジェクトを含むシーンを示す．これらのシーンでは，変形と交差判定が実時間で計算されている．

提案システムは，臓器のアニメーションをデザイン

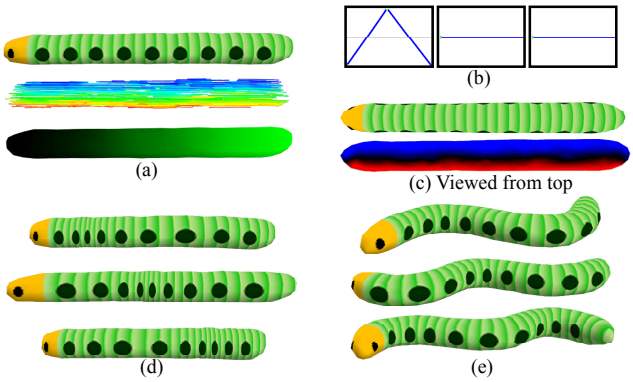

図 8: 芋虫のアニメーション. *モデル上に tensor field と Phase- shift field をデザインし(a)，変形グラフを指定すると，芋虫の歩く動きが生成される．Amplitude field を指定すると(c)蛇のように曲がる動きが生成される.*

モデル	図	頂点数 (領域数)	モデル数	Iteration N	時間 (msec)
クラゲ	図 1	854	1	2	4.37
芋虫	図 8	470	1	6	5.04
クラゲ	図 10 左	315	31	1	31.64
芋虫	図 10 右	73	101	1	34.01
心臓	図 11a-c	1302	1	6	14.79
腸	図 11d-g	1576	1	6	17.49

表 1: ***提案システムのパフォーマンス.*** *時間の列は変形と交差判定に掛った時間(msec)であり，レンダリングに掛った時間は含まない.*

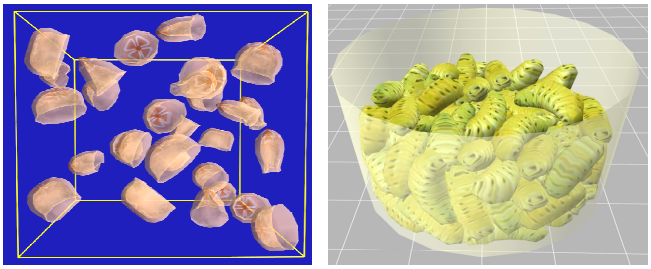

図 10: 複数オブジェクトを含むシーン. *左のシーンでは 31 匹の泳ぐクラゲの，右では 101 匹の芋虫のアニメーションがリアルタイムに計算されている.*

するのにも利用できる．我々は，心臓の四面体メッシュモデルを用意し，心筋線維方向を Takayama らの論文[12]に従いデザインした(図 10(a))．さらに，一次方向に収縮させ，二次方向(厚み方向)には伸長させるように変形グラフを指定すると(図 10(b))，実際のものとよく似た心臓の拍動アニメーションが生成される(図 10(c))．本システムでは，オブジェクトを切って断面のアニメーションを観察することも可能である(図 10(c)下)．図 10(d)-(g)は，腸のアニメーションの例である．我々は，腸のモデルに輪状筋に沿うように tensor field をデザインし，図 10(e)のように変形グラフをデザインした．図 10(f)が結果のアニメーションで，図 10(g)は，一次方向の収縮を赤で，伸長を青で可視化し

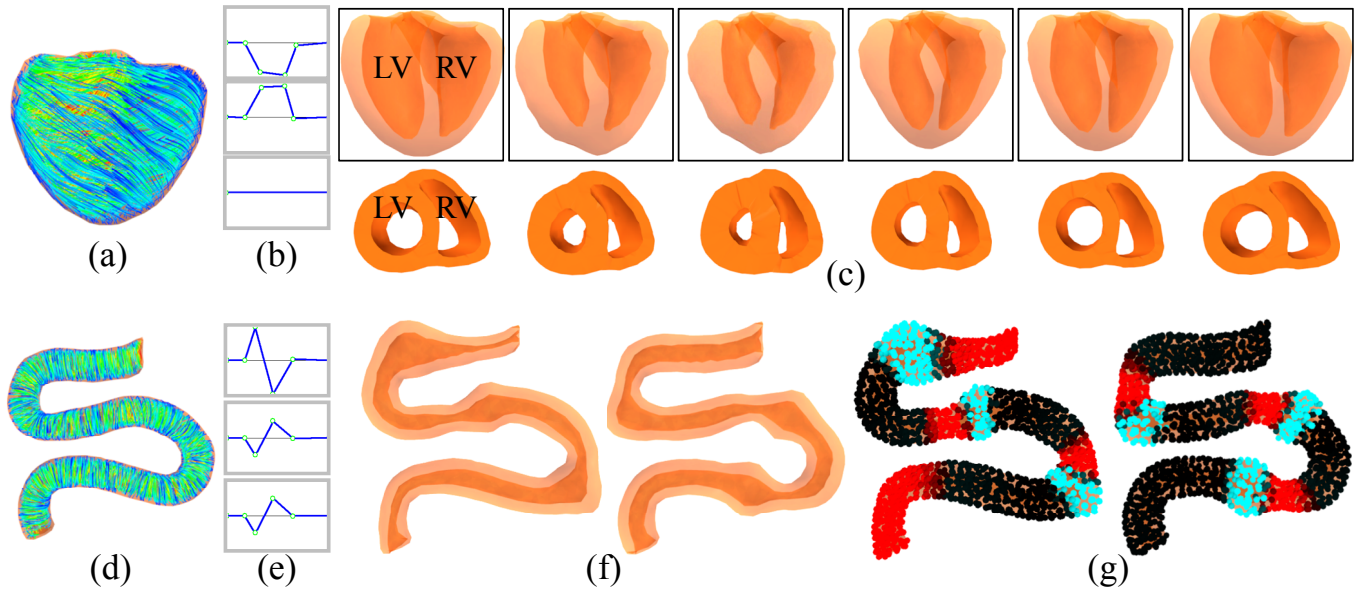

図 10: 心臓と腸のアニメーション.

たものである．これらのアニメーションは，通常の laptop PC 上で実時間で計算されており，提案アルゴリズムは非常に安定なため，ユーザは自由に切ったり引っ張ったりするようなインタラクションが可能である．この点から，提案システムは，手術シミュレータや電子カルテへの応用が期待できる．

表 1 に提案システムのパフォーマンスを示す．この結果は，2.4-GHz Intel Core 2 Duo CPU を利用して計算されたものである．

7. まとめ

本研究では，柔軟物体の能動的な動作をデザインするための新しいフレームワークを提案した．基本となるアイディアは，局所領域の変形の積み重ねにより大域的な変形をデザインする，というものである．我々は，局所領域の変形パターンを効率的にデザインできるインタフェースや，局所変形から大域的な形状を合成する高速かつ堅固なアルゴリズムも紹介した．

現在のシステムでは，硬さの表現に課題がある．我々は，イテレーションにより硬さを制御したが，この計算コストはイテレーションのサイズに線形に大きくなる．つまり，硬いものを表現しようとするとそれだけ計算が遅くなる．計算コストが低くかつサンプリングエラーの少ない，新しい硬さ表現の実装は我々の将来課題である．もう一つの将来課題として，アニメーションの逆問題がある．現在のシステムでは，局所変形パターンをデザインすると大域的な動作が生成されるが，大域的な動作から局所変形パターンを計算できれば，便利なツールとなると考えられる．

文　献

[1] Amano, A., Kanda, K., Shibayama, T., Kamei, Y., and Matsuda, T. Model Generation Interface for Simulation of Left Ventricular Motion. In *IEEE EMBC*. 2004, 3658-3661.

[2] Burton, E. Sodaplay. www.sodaplay.com/.

[3] Combaz, J., and Neyret F. Semi-interactive Morphogenesis. In *Shape Modeling International 2006*, 35.

[4] Eto, K., *et al.* Modulobe: A New 3D Model Creation Platform for Complex Motion Design. *In Proc. Content Creation Activity Support by Networked Sensing 2008.*

[5] Falco. M., Springs World 3D. www.sw3d.net.

[6] Ijiri, T., Yokoo, M., Kawabata, K., and Igarashi, T. Surface-based Growth Simulation for Opening Flowers. In *Proc. Graphics Interface 2008*, 227–234.

[7] Kass, M., and Anderson, J. Animating Oscillatory Motion with Overlap: Wiggly Splines. *ACM Trans. Graph. 2008*, 27, 3.

[8] Müller, M., Heidelberger, B., Teschner, M., and Gross, M. Meshless Deformations Based on Shape Matching. *ACM Trans. Graph. 2005*, 24, 3, 471–478.

[9] Nealen, *et al.* Physically Based Deformable Models in Computer Graphics. *Computer Graphics Forum 2005,* 25, 4, 809–836.

[10] Rivers, A., and James D. L. FastLSM: Fast Lattice Shape Matching for Robust Real-Time Deformation. *ACM Trans. Graph. 2007*, 26, 3, 82.

[11] Rolland, A.-G., Bangham, J. A., and Coen, E. Growth Dynamics Underlying Petal Shape and Asymmetry. *Nature*, 422, 161–163.

[12] Takayama, K. et al. A Sketch-based Interface for Modeling Myocardial Fiber Orientation that Considers the Layered Structure of the Ventricles. *J. Physiol. Sci.,* 58, 7, 487–492.

[13] Takayama, K., Okabe, M., Ijiri, T., and Igarashi, T. Lapped solid textures: filling a model with anisotropic textures. *ACM Trans. Graph,* 27, 3(2008).

[14] Turk, G., and O'Brien, J. F. Shape Transformation Using Variational Implicit Functions. In Proc. SIGGRAPH '99, 335–342.

[15] Waddoups, N. Juice. www.natew.com/juice/.

[16] Watanabe, H. *et al.* Finite Element Analysis of Ventricular Wall Motion and Intra-Ventricular Blood Flow in Heart with Myocardial Infarction. *JSME* 47, 4. 1019–1026.