

ユーザによる対応付けを考慮した3角形メッシュモーフィング*

4 E-6-6

金井 崇 鈴木 宏正[†] 木村 文彦[†]

理化学研究所 素形材工学研究室

〒351-0198 埼玉県和光市広沢 2-1

[†] 東京大学大学院工学系研究科 精密機械工学専攻

E-mail: kanai@postman.riken.go.jp

本論文は、同位相の二つの3角形メッシュからなるモデルに対し、ユーザによる頂点間の対応付けを考慮したモーフィングを実現するための効率的な枠組みを提供するものである。基本的なアイデアは、指定したいくつかの頂点を通してそれぞれのメッシュを分割し、分割された領域ごとにメッシュの面間の対応付けを自動的に行なうところにある。分割されたそれぞれのメッシュは、調和写像により平面上の凸多角形に展開され合成することで、双方の接続性を併せ持つ合成メッシュを生成するために用いられる。この枠組みは、モーフィングのインタラクティブな処理を目指したものである。いくつかの例題を通じてその有効性を検証するとともに、より洗練されたモーフィングのアニメーション結果が得られることを実証する。

1 はじめに

二つの形状モデル間の滑らかな移行を実現する3次元形状モーフィング (*metamorphosis or morphing*) は、今やコンピュータグラフィックスの分野における一つの研究トピックとなっている。2次元の画像に対する融合技術である画像のモーフィングに比べ、平行、回転している形状に対してや、視点が変化するようにするとき (画像の場合、アニメーションのフレームごとに形状が移動するような状況) の補間が可能であるという有用性を持つ。我々は、二つの3角形メッシュで構成されるモデル (以下メッシュと呼ぶ) に対するモーフィングを扱う。二つのメッシュ間のモーフィングに対する主な問題として、メッシュ上の点間の1対1の対応関係の構築、すなわち一方のメッシュ上のすべての点が、もう一方のメッシュ上のどこかの点へ写像する必要がある。一旦対応を構築すれば、それらの対応点の補間によって、連続的な中間形状を生成することが可能になる。これらのステップはそれぞれ対応問題 (*correspondence problem*)、補間問題 (*interpolation problem*) と呼ばれる [14]。ここで提案するアルゴリズムは、主に前者の問題について言及するものである。

ここでは対応問題について、モーフィングの操作性および質を向上するために、特に次の二つの問題について取り上げる。

1. 一般に、二つのメッシュは異なる頂点数や面数、または異なる接続性を持つ。このような異なる二つの形状モデル上のすべての点に対し、1対1の対応関係を構築するための方法論が必要になる。また、形状モデルの中には複雑な形状 (例えば凹凸の混在する形状など) があり、このような形状に対しても対応付

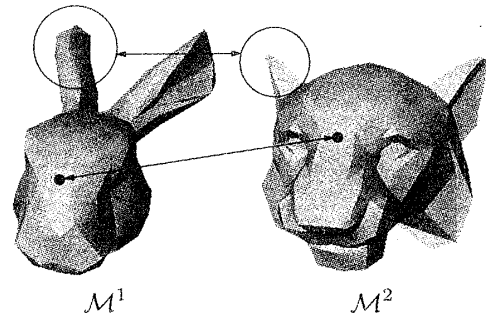


図 1: ユーザの意図する対応付け

けが行なえるような方法論が必要になる。

2. ユーザが適切な対応付けを行えること。例えば、図1のように兎と虎の間のモーフィングを考えると、兎の左耳と虎の左耳を補間したい、という要求があるはずである。このようなユーザの意図を計算によって自動的に抽出することは至極ナンセンスである。かといって、ユーザがメッシュ上のすべての点において対応付けを補助することは、大変骨の折れる作業となる。したがって、要求を満たすためにはユーザがある程度の対応付けの誘導を行ない、このユーザによる対応付けの介入をも考慮して、メッシュ間の対応関係を構築する必要がある。

これらの問題のうち 1. の方は、以前我々が提案した3次元形状モーフィングのための調和写像にもとづく手法 [13] で取り上げている。[13] では、メッシュから2次元平面内への埋め込みを生成し、埋め込みの合成を行なうことで、(形状が凹凸に限らない) 二つの任意メッシュ間の対応関係を構築している。しかしこの方法は、ユーザによる対応付けの制御を考慮した方法ではない。

本論文の目的は、ユーザの指定する (最低限の) 対応付

Triangular Mesh Morphing with User-Specified Correspondence, by Takashi KANAI, Hiromasa SUZUKI**, Fumihiko KIMURA** (*Materials Fabrication Lab., RIKEN, **Dept. Precision Engineering, University of Tokyo)

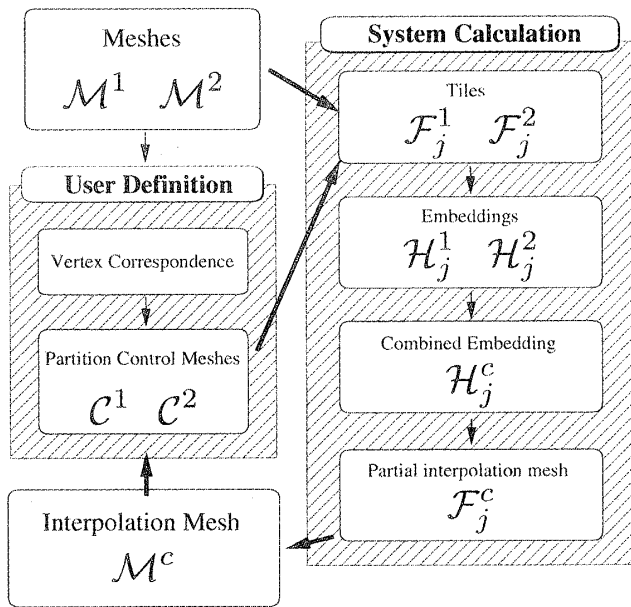


図 2: 提案するモーフィングの枠組み

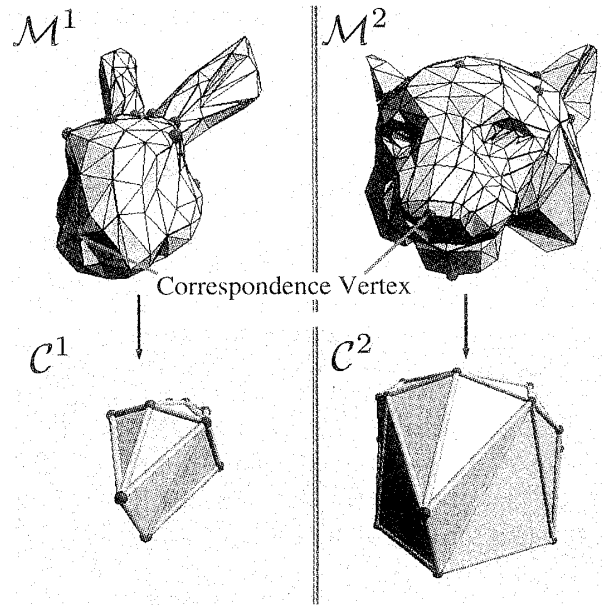


図 3: 分割制御メッシュの生成

けを考慮した二つの位相同形な任意メッシュ間の 3 次元形状モーフィングを実現するための枠組みを構築することにある。このため、ユーザの指定するメッシュの頂点間の対応を考慮できるよう [13] の手法を拡張する。

2 関連研究

モーフィング、特に 3 次元の形状を扱ったモーフィングに関する研究のサーベイは [9, 16] に詳しい。本研究ではその中でもメッシュ (ポリゴン) に関するモーフィングの部類に属する。過去のメッシュに関するモーフィングの研究では、星型形状 [14] やスワイプ形状 [14] など、扱える形状の幾何的性質に制限を持つものが多かったが、ごく最近の研究 [6, 13, 10, 3, 2, 17] では同位相の任意メッシュを扱えるものが多い。その中でも本研究では、ユーザがモーフィングの結果に関して思考錯誤できるだけ、効率のよい枠組みの確立により重点を置いている。

3 枠組みの概要

ユーザによるモーフィングの制御には様々な方法が考えられるが、ここでは頂点对応による制御に焦点をあてる。頂点对応は、複数の二つの頂点对群のことを指し、一つの頂点对を構成する頂点は、二つのメッシュからそれぞれ一つずつ選ばれる。この対をここでは対応頂点对 (corresponding vertex pair, CVP) と呼ぶことにする。CVP は、一方のメッシュにおけるその頂点が、もう一方のメッシュにおける対応する頂点に移る、という制約となる。これは最も基本的な制御方法であるとみなすことができる。このような制約を加味するために、本研究では、ユーザの

指定するこの頂点对応を通るようにメッシュを分割する、という方法をとることにする。

図 2 に我々の提案するモーフィングの枠組みを示す。この枠組みは、ユーザによる頂点对応の指定と、システムによる対応付けの計算から構成される。

ユーザによる指定では、ユーザが望むモーフィングの制御を行なうために、二つのメッシュ M^1, M^2 間に対応頂点 (correspondence vertex, CV) を指定する。この CV を用いて、次に分割制御メッシュ (partition control mesh, PCM) C^1, C^2 を設計する。

システムは、二つのメッシュのそれぞれを PCM に従って分割し、分割したメッシュをグループ化して、 $\hat{M}^1, \hat{M}^2$:

$$\hat{M}^1 = \bigcup_j^{n_j^c} \mathcal{F}_j^1, \quad \hat{M}^2 = \bigcup_j^{n_j^c} \mathcal{F}_j^2, \quad (1)$$

を得る。このグループ化されたメッシュのそれぞれを、ここではタイル (tile) と呼ぶことにする。そしてこのタイルごとに合成し、組み合わせることで、合成メッシュ M^c を得る。この合成メッシュを用いてモーフィングのアニメーション結果を評価し、CVP の指定や PCM の設計の改良にフィードバックする、というのが全体の枠組みである。

4 頂点对応指定にもとづくメッシュの分割とグループ化

4.1 分割制御メッシュ

図 3 のように、ユーザは最初に二つのメッシュから一つずつ頂点 (CV) を選択することで、CVP を指定する。

次に、これらの頂点を通るように PCM C^1, C^2 を設計する。PCM の一つの面は、CV を連続的に選択し、その頂点ループにより構成される。このときの面は必ずしも3角形である必要はない。ただ、頂点ループの向きは、メッシュの面のループの向きと同方向（ここでは反時計回りとした）にする必要がある。PCM C^1, C^2 は、メッシュ M^1, M^2 と同位相になるように設計される。 C^1, C^2 は、 C^1, C^2 のどちらかの CV のみから設計することで、互いに同位相、かつ、同じグラフ構造を持ち、頂点座標のみが異なるメッシュとなる。

メッシュの面間の対応の制御が必要かどうかは、対応付けした CVP の数に依存する。最低必要な CVP の数は、面間の対応付けに必要なメッシュのパラメトリゼーションによって決まる。我々のとった後述のパラメトリゼーションでは、球に位相同型のメッシュの場合、二つの円盤に分割するために、PCM の面は最低二つ必要である。一つの円盤の最小構成要素は3角形なので、少なくとも三つの CVP が必要である。CVP が三つのときは、ユーザは同じ三つの CVP を持つ（向きの異なる）二つの面を作成すればよい。ただ、我々の現在の実装では、PCM をユーザが視覚的に理解できるように、最低4面体の PCM を持つために四つ以上の CVP を定義する必要がある、という付加的な制限を設けている。

4.2 メッシュの分割とグループ化

PCM C^1, C^2 の位相に従い、メッシュ M^1, M^2 を複数のタイル:

$$\begin{aligned} \mathcal{F}_j^1 \subseteq \mathcal{M}^1 \quad \mathcal{F}_j^2 \subseteq \mathcal{M}^2 \\ j = 1 \dots n^{\mathcal{F}} \quad (n_{\mathcal{F}}^1 = n_{\mathcal{F}}^2) \end{aligned}$$

に自動的に分割する方法について述べる。分割方法は、次の三つのステップからなる：まず、 C のエッジを、端点が CV となり、 M 上の経路になるよう M に“埋め込む”。次に、経路が通る M の面を、経路に従って“切断”する。最後に、複数のタイルに分割するために、 M の面を集める。以下に各ステップについて述べる。

分割制御メッシュのエッジのメッシュへの埋め込み

C のエッジを M に“埋め込む”ために、我々は、これを頂点間の M 上を通過する経路算出問題に置き換える。

一つの方法としては、 M の頂点/エッジのグラフ構造を用いて、グラフ上の最短経路の計算に置き換える方法が挙げられる [10]。しかし、この方法だと、我々の実験では、二つの近い距離にある経路が重なってしまうケースが起こる。特に、PCM を細かくすればするほど、このようなケースが頻繁に起こり得ることがわかった。なお、このようにしてできた経路はメッシュのエッジ上にあるため、しばしばギザギザな形状となり、これが最終的なモーフィングの結果に影響を及ぼすこともわかった。かといって、正確な最短経路計算を用いると [20, 5]、最低でも M の

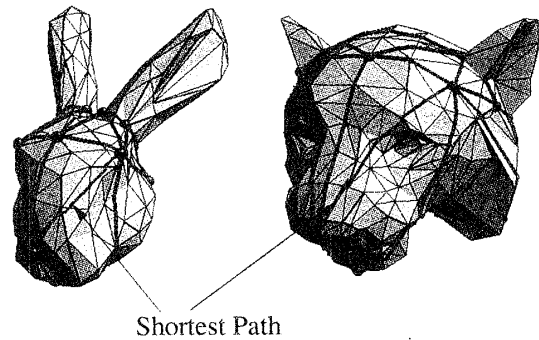


図 4: 近似最短経路による C のエッジの M への埋め込み

頂点の数 n に対し $O(n^2)$ の計算量を必要とする。我々がここで欲しいのは二つの頂点間の滑らかな経路であるが、必ずしも最短経路である必要はない。そこで我々は、メッシュ上の二点間の近似最短経路算出法 [15] を用いた。以下に経路算出手法について述べる。

経路計算の前に、メッシュの面上に中間エッジを次のように作成しておく：まず、 M のエッジ上に、エッジの長さに応じたスタイナ点 (Steiner Point) と呼ばれる中間頂点を発生させる。長いエッジにはより多くの点を発生させるようにする。次に、生成された付加的な頂点および M の頂点の中から二つの頂点对 $\langle v_a, v_b \rangle$ を選び、その対が次の条件 1 を満たすとき、 $e = \{v_a, v_b\}$ となるエッジを加える：

条件 1

1. $\langle v_a, v_b \rangle$ はある面の異なるエッジ上にある (分割) 頂点である
2. $\langle v_a, v_b \rangle$ は同じエッジ上にある隣接した (分割) 頂点である

これより、経路探索用のグラフ G が生成される。なお、このグラフは一度求めるだけで良い。近似最短経路は、この経路探索用のグラフに対してダイクストラのアルゴリズム [1] を適用してできた初期経路に、[12] の多角形内の経路算出法を拡張した方法を適用して算出される。計算量は、グラフのエッジに対し $O(n \log n)$ だけかかる。図 4 に C のエッジを M に“埋め込んだ”結果を示す。このように視覚的には非常に滑らかな経路を生成できていることがわかる。

ただし、(我々の方法に限らず) この最短経路に基づくエッジの埋め込みのすべての方法にいえることだが、どのような状況でも経路が望み通りに求まるとは限らない。極端なことを言えば、球の南極と北極の間の経路は無限にあるわけで、しばしば (特に経路が長いときには) ユーザの意図しない経路が求まることも起こり得る。我々は、このような場合には、生成された経路を確認し、もし意図しないものであれば、経路が長くないように CVP を増やすことで対応している。

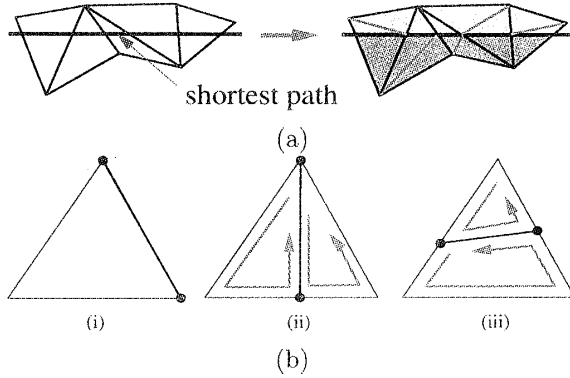


図 5: (a) 最短経路上でのメッシュの面の分割 (b) 面の分割のための三つのパターン

メッシュの面の分割

面ベースのグループ化を行なうためには、まずグループ化に先立ち、最短経路が通る M の面の分割を行なう (図 5(a)). ここで、一つの面に頂点以外の交点が何点存在するかで、図 5(b) のような三つの分割パターンに分けることができる。分割した面は必ずしも 3 角形にはならないので、分割を行なった後に 3 角形化する。これらのすべての頂点は平面上に乗っており、しかもせいぜい 4 角形であるため、3 角形化は容易に行なうことができる。

メッシュのグループ化

上記の操作後に、いくつかの経路で区切られたタイル群を定義する。一つのタイルは連続的な M の面群から成り、これらは面のグループ化を施すことで得られる。

面のグループ化は、まずグループ内にある一つの面からはじめ、隣接する面を順次探索する、という再帰的な方法をとる。このとき、隣接する面間にあるエッジが経路のエッジであれば、その面に関する探索はそこで終了する。このようなアルゴリズムにより、面の数を n とすると $O(n)$ で求めることが可能である。

なお我々の実装では、 M^1 と M^2 のグループ化をそれぞれ独立に処理するようになっているが、その中の対応するタイルの頂点と面は同じ ID を持たせるようにしている。このとき、CVP や経路上の M の頂点は、二つ以上の異なるタイルに属することになる。

4.3 分割補間メッシュの生成

グループ化されたメッシュ $\hat{M}^1, \hat{M}^2$ から、[13] の手法により合成メッシュ M^c を生成する。

まず、各タイル $\hat{F}_j^1, \hat{F}_j^2$ それぞれを、調和写像 [7] により 2 次元平面上の原点を中心とする単位円に展開し、埋め込みメッシュ $\hat{H}_j^1, \hat{H}_j^2$ を生成する。 $\hat{F}_j$ と $\hat{H}_j$ はそれぞれ同じ頂点数、面数および接続性を持つが、頂点の座標値が異なる。 $\hat{F}_j$ の頂点の座標値はもとのメッシュと同じ 3 次元空間上の座標値であるのに対し、 $\hat{H}_j$ のそれはすべて 2

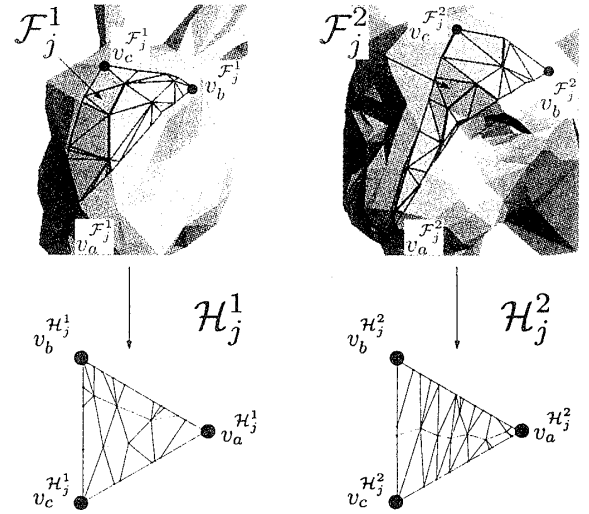


図 6: 分割メッシュごとの埋め込みの生成

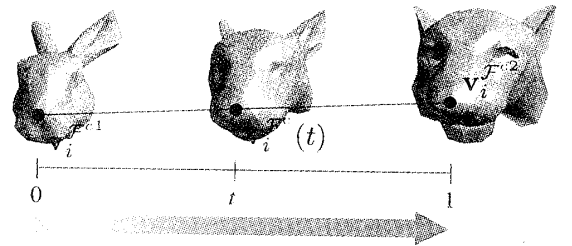


図 7: 分割補間メッシュにおける頂点の座標値の算出

次元平面の単位円内に置かれている。 $\hat{F}_j^1, \hat{F}_j^2$ の境界は、それぞれ境界写像 g により埋め込みメッシュの境界に写される。このとき、境界を構成する頂点のうちの CV に相当する頂点が、 n 角形の頂点を構成する。 $\hat{F}_j^1, \hat{F}_j^2$ の CV は、 g によって 2 次元平面上の同じ座標値に写される。例えば図 6 では、ユーザが指定した CV のうち、二つのモデルの鼻の部分は、埋め込みメッシュの●印の点に写され、これらは同じ座標値を持つ。

次に、 j 個のペアとなる二つの埋め込み $\hat{H}_j^1, \hat{H}_j^2$ を合成し [23]、埋め込み合成メッシュ $\hat{H}_j^c$ を得る。 $\hat{H}_j^c$ は $\hat{H}_j^1, \hat{H}_j^2$ を合わせた接続性を持つ。 $\hat{H}_j^c$ の各頂点は、 $\hat{H}_j^1, \hat{H}_j^2$ の頂点、あるいは面上の 1 点と対応付けされている。

最後に、各タイルごとに生成された埋め込み合成メッシュ $\hat{H}_j^c$ の和を取ることで合成メッシュ M^c を得る。 M^c の各頂点は二つの座標値を持ち、一つは M^1 の、もう一つは M^2 の各頂点、あるいは面上の 1 点の座標値である。そして、二つの座標間を線形補間することにより (図 7)、 M^1, M^2 間の中間形状を生成する。

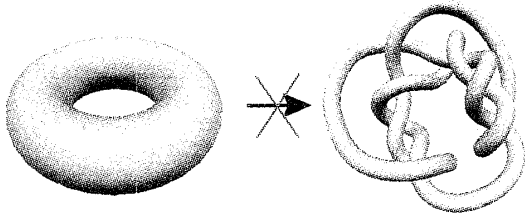


図 8: トーラス型メッシュで補間制御メッシュを設計できない例

4.4 境界付きメッシュ・トーラス型メッシュ他における補間制御メッシュの生成

本手法は、適切な CVP や PCM を設計することで、境界 ∂M を持つメッシュや、トーラス型メッシュ (種数 g が 1 のメッシュ) などにも適用することが可能である。

境界付きメッシュに対しては、ユーザは M の境界に最低二つ以上の CVP を作る必要があり (ただ我々は球型の場合と同様、ユーザによる視覚的理解の点から三つ以上という制約をおいている)、なおかつそれぞれのメッシュにおける境界上の CV 間の経路は、境界上のエッジのみで構成される必要がある。ただこの経路の算出には、4.2 項の近似最短経路算出法を用いる必要はなく、単に境界上のエッジを探索することで得られる。

種数 g が 1 のトーラス型メッシュや、 g が 1 よりも大きいメッシュに関しては、我々の手法では PCM を設計できない場合がある。図 8 にそのような場合の例を示す。トーラス型メッシュの場合は、少なくとも同位相の条件に加えて “tamely homeomorphic” [4] の条件が必要となる。この条件はメッシュの結び目 (knots) に関するものである。さらに我々は、適切な PCM を設計する必要がある。現時点でそのための正しい条件を記述することはできないが、少なくとも言えることは、PCM はメッシュの穴 (hole) をカバーするように設計されていることが必要である。

5 適用結果

我々は、SGI 社のグラフィックスワークステーション Indigo² High Impact (MIPS R4400 250MHz CPU, 128MB memory) 上でプロトタイプシステムを実装した。そして、システム上でいくつかの例題に本手法を適用し、検証した。図 9 に作成したシステムの表示例を示す。

図 10 に、兎の頭と虎の頭の二つのメッシュに対し、異なる PCM によるモーフィング結果を示す。これら二つのメッシュは球に位相同形である。(a) には 5 つの CVP による PCM を、(b) には耳の付け根により詳細な頂点对応を施した 16 個の CVP による PCM を示す。(c),(d) を比較すると、(d) の方がより洗練されたモーフィングを行えることがわかる。図 11 には、他のメッシュに関するモーフィングの結果を示す。図の上からそれぞれ星型形状 →

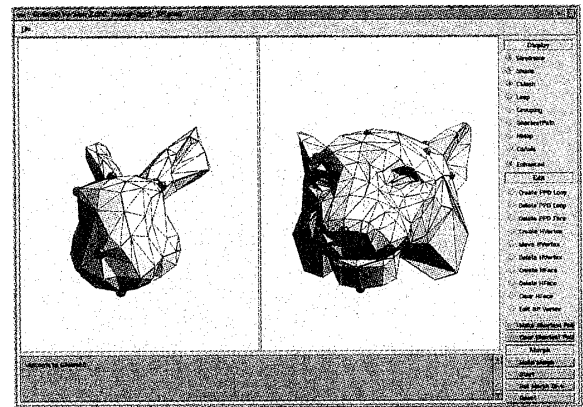


図 9: モーフィングシステムの表示例

フィギュア (面の数の多いメッシュへの適用例), デロリアン → ボルシェ (境界付きメッシュへの適用例), トーラス → ボトル (トーラス型メッシュへの適用例) を表している。

表 1 に、各例題におけるメッシュ、制御補間メッシュ、それに合成メッシュの頂点数、面数、と各処理における計算時間を示す。ここで、 T_s, T_h, T_c は近似最短経路の算出、調和写像の計算、メッシュの合成の計算のそれぞれの合計時間を示す。表の中で調和写像の計算や、メッシュの合成の計算は高速に処理できることが確認できる。最も時間のかかる処理は近似最短経路の計算であるが、これは経路探索用のグラフ G のエッジの数が多くなり、グラフ上の最短経路探索に時間がかかっていることが原因である。我々の実験では、滑らかな経路を生成するためには、平均して一つの M のエッジにつき 6 点程度のスタイナ点を追加する必要がある¹。

6 考察

6.1 モーフィングの質に対する考察

対応付けの観点から見れば、モーフィングの質に影響を与えるのは、ユーザが指定する頂点对応と、パラメトリゼーションに依存する面間対応の二つに分けられる。

頂点对応の点から考えると、モーフィングの質は CVP を増やすか、各メッシュにおける CV を適切に再配置することで、改良することが可能である、と言える。我々の経験によれば、メッシュの特徴的な部位 (例えば耳とか目など) の対応付けには、その境界付近に数点の CV を配置することで、ある程度の制御は可能になる。より洗練された制御を求めるのであれば、内部にさらに数点の CV を加えていけばよい。特徴的な形状をより多く含むメッシュの場合は、より多くの CVP が必要になる。ただ、メッシュ上のある特定の領域に CV を密集させることは、計算時間がかかるだけで、モーフィングの質の改良にはつながら

¹最近我々が提案した手法 [22] では、 G における付加すべきスタイナ点を軽減できることが確認されている。

Examples		兎一虎 (a)	兎一虎 (b)	デロリアン-ボールシェ	星型形状-フィギュア	トーラス-ボトル
$\mathcal{M}^1$	$n_v^{\mathcal{M}^1}$	188	188	400	5,534	480
	$n_f^{\mathcal{M}^1}$	372	372	738	11,064	960
$\mathcal{M}^2$	$n_v^{\mathcal{M}^2}$	254	254	1,726	12,344	474
	$n_f^{\mathcal{M}^2}$	504	504	3,311	24,684	948
$\mathcal{C}$	$n_v^{\mathcal{C}}$	5	16	40	28	17
	$n_e^{\mathcal{C}}$	9	38	68	55	35
	$n_f^{\mathcal{C}}$	6	24	29	29	18
$\mathcal{M}^c$	$n_v^{\mathcal{M}^c}$	1,865	2,320	9,724	77,269	4,965
	$n_f^{\mathcal{M}^c}$	3,726	4,636	19,265	154,534	9,930
Time (Sec.)	T_s	24.6	104.4	99.7	169.23	109.1
	T_h	0.1	0.1	0.7	11.95	0.2
	T_c	0.4	0.5	2.1	18.77	1.3

表 1: 3 角形メッシュモーフィングの各例題に対する各要素の数と計算時間

い.

一方、面間対応の観点から見れば、我々の利用したパラメトリゼーションである調和写像 [7] は、3 角形のアスペクト比は保存するものの、面積は保存しない。このため、多くの数の 3 角形群が、狭い領域に密集するということが起こる。この性質は、面間対応がある領域に密集し、アンバランスな対応になる、という悪影響を及ぼす。他のパラメトリゼーション手法 (例えば [8, 19, 18, 10] など) を用いることで、この不具合はある程度改善されると予想されるが、我々の手法でも、メッシュが密集した領域で CVP をより多くとることで改善できると考える。

6.2 パフォーマンスに関する考察

CVP の指定や PCM の設計におけるパフォーマンスは、GUI で作業の労力に大きく依存する。CVP の指定そのものは、頂点をピックする作業が主で、それほど負担のかかるものではない。さらに、PCM の設計は、その指定した CV を見ながら行なえばよいので、この作業も難しい作業ではない。我々のプロタイプシステムを使用した経験では、PCM の設計は、図 10 の例でせいぜい数分、最も面の数の多い図 11 の星型形状 → フィギュアの例でも 30 分以内で行なうことができた。

ユーザにとって本手法が有効であるかどうかについて考慮すべきこととして、二つの問題がある。

一つ目に、CVP と同時に PCM を設計しなければならないことが、退屈な作業ではないか、という点である。我々は、この作業が必要であるかどうかは、ケースバイケースであると考え。我々の考えでは、PCM を設計する、という作業は、二つのメッシュの特徴的な部位間の明示的な指定と、その視覚的な確認作業を併せ持ち、これはユーザが対応付けを直感的に理解するために、自然で重要なプロセスである。ただ、このような明示的な指定の必要のない場合に関しては、PCM を自動的に生成できた方が、ユーザにとって作業を軽減するという意味で有効であると思われる (現に [10] や [17] では自動生成を実現している)。

二つ目に、PCM が同じ位相になるように二つのメッ

シュから CV を選択するのは難しい作業ではないか、という点である。確かに、CVP が多くなればなるほど、明らかに PCM の設計が難しくなるということは言える。もしそうだとすると、以下のように PCM の設計方法に制約を設けることで、ある程度は改善されると思われる：まず、PCM を 4 面体などのラフなものから設計し始める。設計途中の段階で、一つのメッシュ上のある頂点を CV として指定するとき、もう一つのメッシュ上の頂点は、対応するタイルの中の頂点しか指定することができない。この制約は、CVP を選択するための一つのガイドラインになり得るが、PCM の設計方法のさらなる改良に関しては、今後の課題の一つである。

7 結論と展望

本論文では、ユーザによる対応付けを考慮した 3 角形メッシュモーフィングを実現するための効果的な枠組みを導入し、調和写像を用いた面間の対応付けを利用するために、メッシュを分割するアルゴリズムを提供した。そして、本手法をいくつかの例題に適用し、枠組みの有用性とアルゴリズムの効率性について実証した。本研究で得られた知見としては、モーフィングの質を高めるには、どのような手法を用いるかということ以外にも、途中の結果に対してユーザが思考錯誤できるだけの枠組みが重要である。その意味で、本手法のようにインタラクティブな処理を目指した枠組みは有効である、ということが言える。

我々は、以下に述べる展望がモーフィングのさらなる質の向上に役立つと考えている。

(我々の手法に限らず) 合成メッシュの面の数は、もとのメッシュの数に比べ、極めて多くなってしまふ。それ故、モーフィングのアニメーション表示には時間がかかる。モーフィングのリアルタイム表示は、ユーザによる結果の試行錯誤には重要な課題の一つであり、そのためにはメッシュの特徴を失わずに合成メッシュの数を減らせるような方法が必要になる。

また、我々は補間方法に関しては触れなかったが、我々の用いた線形補間では、中間形状が自己干渉を起こすな

ど, いくつかの問題があることがわかっている. 補間手法に関してはいくつかの研究が存在するが(例えば[21, 11]など), 我々の方法にマッチするような補間手法を確立することが課題となる. ただ我々は, この自己干渉の問題を完全に解決するのは極めて難しく, 例えばアニメーションの間にそれを目立たなくする等の方法も検討する必要がある, と考えている.

また, 本論文でのモーフィングにおける補間関数は, 各タイルにつき一様の関数(線形補間)を用いているが, 個々のタイルに別の関数を割り当てることも可能であり, このことがモーフィングの柔軟性を向上させる, と考えている.

なお, 以下のホームページに3次元形状モーフィングに関する情報を集めているので, 是非参照されたい:

<http://www.riken.go.jp/lab-www/mat-fab/personal/kanai/Gmorph.html>

謝辞

例題で用いた兎は, スタンフォード大のグラフィックス研究室で作成されたメッシュを用いている. また, 虎, デロリアン, ボルシェ, ボトルのメッシュは, Viewpoint Datalabs Inc. のメッシュデータである. 東京大学大学院生の山本徹也氏(現東芝), 三谷純氏(現修士課程)には, カラー出力作成に際しご協力頂いた. また, (株)ナブラの小高金次社長を始めとする開発メンバーには, 実用上の面から有用なご意見を頂戴した. また, この研究の一部は(財)大川情報通信基金の支援を受けた. 併せてここに感謝の意を表する.

参考文献

- [1] A. V. Aho, J. E. Hopcroft, and J. D. Ullman. データ構造とアルゴリズム(邦訳). 培風館, 1987.
- [2] M. Alexa. Merging polyhedral shapes with scattered features. In *Shape Modeling International '99*, pp. 202-210. IEEE CS Press, Los Alamitos, Calif., 1999.
- [3] H. Bao and Q. Peng. Interactive 3D metamorphing. *Computer Graphics Forum (Proc. Eurographics '98)*, 17(3):C-23-C-30, 1998.
- [4] M. Boyer and N. F. Stewart. Modeling spaces for tolerated objects. *Int. J. Robotics Research*, 10(5):570-582, 1991.
- [5] J. Chen and Y. Han. Shortest paths on a polyhedron. In *Proc. 6th ACM Sympo. on Computational Geometry*, pp. 360-369. ACM Press, New York, 1990.
- [6] D. DeCarlo and J. Gallier. Topological evolution of surfaces. In *Proc. Graphics Interface '96*, pp. 194-203. Morgan Kaufmann, San Francisco, Calif., May 1996.
- [7] M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery, and W. Stuetzle. Multiresolution analysis of arbitrary meshes. In *Computer Graphics (Proc. SIGGRAPH 95)*, pp. 173-182. ACM Press, New York, 1995.
- [8] M. S. Floater. Parametrization and smooth approximation of surface triangulations. *Computer Aided Geometric Design*, 14(3):231-250, 1997.
- [9] J. Gomes, L. Darsa, B. Costa, and L. Velho. *Warping and Morphing of Graphical Objects*. Morgan Kaufmann, San Francisco, Calif., 1998.
- [10] A. Gregory, A. State, M. Lin, D. Manocha, and M. Livingston. Feature-based surface decomposition for correspondence and morphing between polyhedra. In *Proc. Computer Animation 98*, pp. 64-71. IEEE CS Press, Los Alamitos, Calif., June 1998.
- [11] A. Gregory, A. State, M. Lin, D. Manocha, and M. Livingston. Feature-based surface decomposition for polyhedral morphing. Technical report, TR98-014, Dept. of Computer Science, Univ. of North Carolina at Chapel Hill, 1998.
- [12] L. J. Guibas and J. Hershberger. Optimal shortest path queries in a simple polygon. *Journal of Computer and System Sciences*, 39(2):126-152, 1989.
- [13] T. Kanai, H. Suzuki, and F. Kimura. Three-dimensional geometric metamorphosis based on harmonic maps. *The Visual Computer*, 14(4):166-176, 1998.
- [14] J. R. Kent, W. E. Carlson, and R. E. Parent. Shape transformation for polyhedral objects. In *Computer Graphics (Proc. SIGGRAPH 92)*, pp. 47-54. ACM Press, New York, 1992.
- [15] M. Lanthier, A. Maheshwari, and J.-R. Sack. Approximating weighted shortest paths on polyhedral surfaces. In *Proc. 13th ACM Sympo. on Computational Geometry*, pp. 274-283. ACM Press, New York, June 1997.
- [16] F. Lazarus and A. Verroust. Three-dimensional metamorphosis: a survey. *The Visual Computer*, 14(8/9):373-389, 1998.
- [17] A. W. F. Lee, D. Dobkin, W. Sweldens, and P. Schröder. Multiresolution mesh morphing. In *Computer Graphics (Proc. SIGGRAPH 99)*, 1999. to appear.
- [18] A. W. F. Lee, W. Sweldens, P. Schröder, L. Cowsar, and D. Dobkin. MAPS: Multiresolution adaptive parameterization of surfaces. In *Computer Graphics (Proc. SIGGRAPH 98)*, pp. 95-104. ACM Press, New York, 1998.
- [19] B. Lévy and J.-L. Mallet. Non-distorted texture mapping for sheared triangulated meshes. In *Computer Graphics (Proc. SIGGRAPH 98)*, pp. 343-352. ACM Press, New York, 1998.
- [20] J. S. B. Mitchell, D. M. Mount, and C. H. Papadimitriou. The discrete geodesic problem. *SIAM J. Computing*, 16(4):647-668, 1987.
- [21] Y. M. Sun, W. Wang, and F. Y. L. Chin. Interpolating polyhedral models using intrinsic shape parameters. In *Proc. Pacific Graphics '95*, pp. 133-147. World Scientific, Singapore, 1995.
- [22] 金井, 鈴木. グラフの選択的詳細化による多面体モデル上の近似最短経路算出法. 1999 年度精密工学会秋季大会学術講演会, 1999. (発表予定).
- [23] 金井, 鈴木, 木村. 円盤と位相同型の任意の三角形メッシュ間の高速な合成法. 情報処理学会 グラフィクスと CAD 研究会 夏の集中研究集会, 1999. (発表予定).

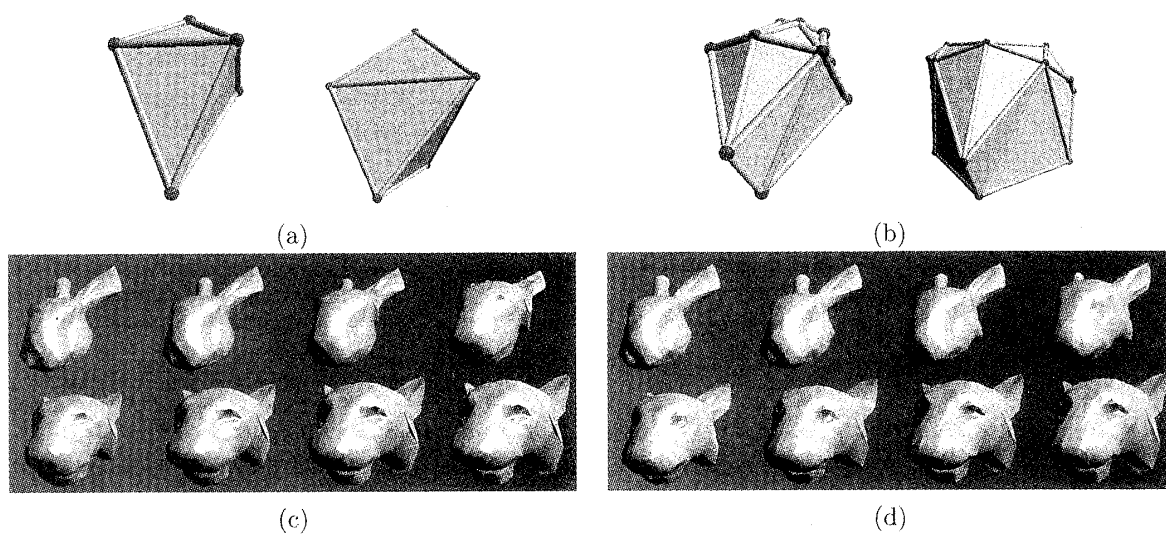


図 10: 頂点指定によるモーフィングの制御例

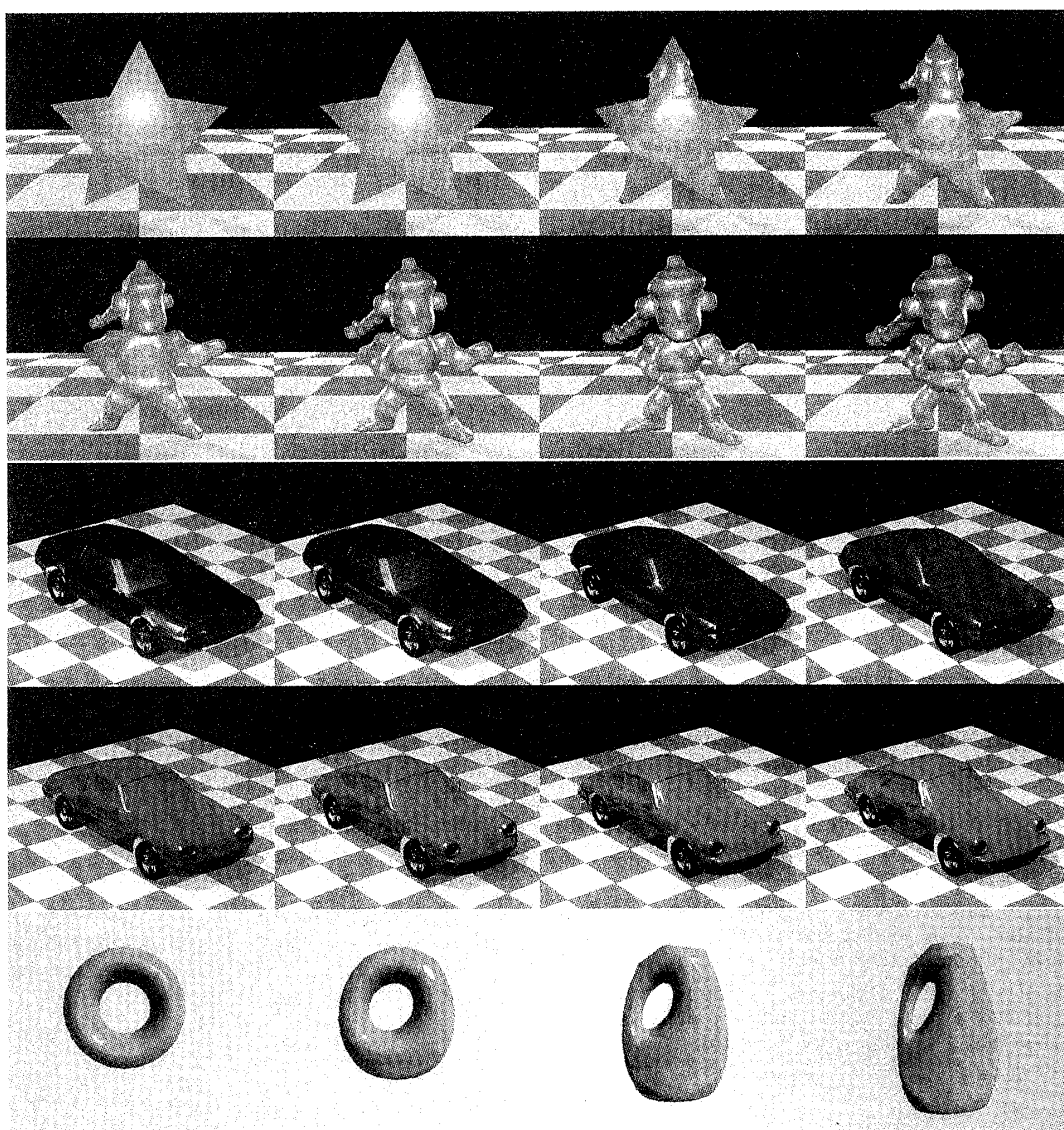


図 11: モーフィング・ギャラリー