Vol.2011-CG-142 No.16
2011/2/9

曲率を考慮に入れた3次元メッシュの展開図生成

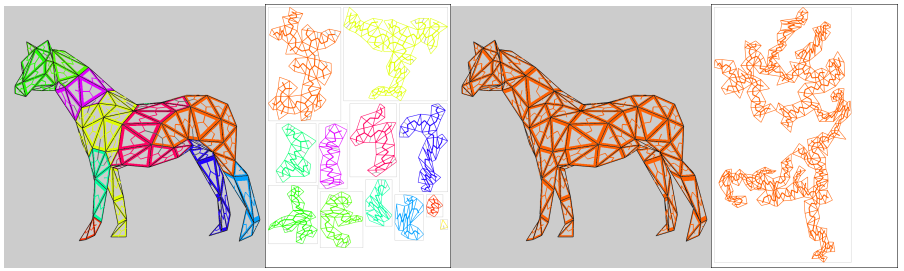
高 橋 成 雄[†1] Seow Hui Saw [†1] Hsiang-Yun Wu [†1]
Chun-Cheng Lin [†2] Hsu-Chun Yen[†3]

3 次元三角形メッシュの 2 次元展開図生成は，現実世界に 3 次元モデルを仮想的に再構成する手段を提供する．しかし，3 次元メッシュを 1 つの連結成分に展開する問題は一種の未解決問題であり，結局のところ切断する稜線集合のすべての組み合わせを調べるしか解法がない．本報告では，展開図パッチの個数とともに展開図に必要な紙面の大きさも効果的に最適化する，新しい展開図生成のための発見的手法を提案する．本手法の基本アイデアは，メッシュの頂点における曲率を評価することでそれらを鞍点と非鞍点に分類し，さらにそれぞれの頂点における切断稜線の個数を保持することで，展開図生成において問題となる局所的自己交差を即座に除外していく点にある．いくつかの事例を用いることで，提案手法が広い範囲の 3 次元メッシュを 1 つの連結成分に変換できることを示していく．

Curvature-Aware Optimized Unfolding of 3D Meshes

Shigeo Takahashi,[†1] Seow Hui Saw ,[†1]
Hsiang-Yun Wu ,[†1] Chun-Cheng Lin [†2]
and Hsu-Chun Yen[†3]

Unfolding 3D triangular meshes into 2D patterns allows us to recreate virtual 3D models in the real world. However, unfolding such a 3D mesh into a single connected component is still an open problem, and the only known algorithm for obtaining an optimal unfolding is to try all the combinations of edges to be cut over the given mesh. This report presents a new heuristic approach for effectively minimizing the number of connected components in the mesh unfolding together with the size of a paper sheet. The key idea behind our approach is to classify the mesh vertices into saddles and non-saddles by evaluating their surface curvatures, and retain the number of cut edges at each vertex so that we can instantly reject unwanted local overlaps between pairs of neighboring faces in the same connected component. Several examples are included to demonstrate that the proposed approach can transform a broad range of 3D meshes into single-connected patterns.



(a) (b)

図 1 horse(312 faces) モデルの展開結果．(a) 従来手法[1] による結果．展開図には小さい小片が含まれる．(b) 提案手法の結果．展開図は 1 つの連結成分になる．ここでは，展開図の連結成分ごとに異なる色を用いており，さらに暗転は青い球で表されている．

1. はじめに

紙工作モデル用の 3 次元メッシュの 2 次元への展開は，ディスプレイに表示される仮想モデルを，実在するモデルとして構築することを可能とする．このような展開図生成手法は，3 次元シーンや建築物の模型を作成したり，紙工作で都市計画を表現する上で，非常に重要な技術となる．また展開図からの 3 次元モデル構築それ自体も，子供や家族が楽しい時間を共有するための興味深い娯楽と位置づけることもできる．

展開図生成には従来いくつか手法が提案されているが，一般的に図 1(a) にあるように，入力 3 次元モデルは数多くの展開図パッチに分解され，それらのいくつかは非常に数個の面だけを含むような小片になることが多い．これでは，異なる展開パッチ間の境界稜線の同士の対応関係を見つけるのに時間がかかる上に，先に述べた小片を紙工作にぴったりと貼り合わせる手間が生じてしまう．この問題は，展開図パッチの個数をできるだけ少なく，望むらくは図 1(b) にあるように 1 つにしてしまうことで解決を図ることができる．しかしなが

†1 東京大学
The University of Tokyo
†2 台北市立教育大学
Taipei Municipal University of Education
†3 国立台湾大学
National Taiwan University

らこの問題は，文献 2)，3) にある通りよく知られた未解決問題であり，いままで理論的な観点から多くの研究がなされてきているが，実際のところ唯一の解法は切り開く稜線のすべての組み合わせを試すことになる．そのため 3 次元メッシュの面数が 100 程度でも，組み合わせ爆発のため現実的な時間で問題を解くことが難しいのが現状である．

本報告では，展開図の紙片の個数とともに展開図に必要な紙面の大きさも効果的に最適化する，新しい発見的手法を提案する．面数が 500 程度の 3 次元メッシュが手で組み立てられる紙工作モデルの限界と考えられるが，図 1(b) に示されている通り，我々の手法はそのサイズのメッシュのほぼすべてを，1 つの展開図パッチに変換することができる．我々の手法は，多くの従来手法と異なり，入力された 3 次元メッシュを伸縮させることなく元の幾何・位相情報を保持することができる．

本手法のアイデアは，3 次元メッシュの頂点を，その点における離散ガウシアン曲率を用いて**鞍点**と**非鞍点**に分類し，実際に 2 次元に 3 次元メッシュを展開する前に，局所的自己交差を即座に取り除く点にある．これは，それぞれの頂点における切断稜線の個数を保持することにより実現される．なぜなら，鞍点において展開図パッチの局所的な自己交差は避けるためには，少なくとも 2 つの稜線を切断する必要があるからである．これにより提案手法は，不適切な切断稜線の組み合わせの多くのチェックを簡単に省略することができ，結果として展開図の探索空間を効果的に縮小することで必要な計算時間を大きく短縮することができる．

本手法の流れは以下の通りである．まず，入力 3 次元メッシュを，その双対の最小全域木 (minimum spanning tree) を用いて三角形ストリップの集合に分解する．そして，それらの三角形ストリップを，**遺伝的アルゴリズム**を用いて境界稜線に沿ってひとつずつ接合する．ここで先に述べた通り，展開図に局所的自己交差が生じる境界稜線における接合は即座に除外して考えないことにする．以下本報告では，2 節で関連研究に触れたのち，3 節において本研究の基本的なアイデアについて説明する．3 次元メッシュの展開処理については，上記のように三角形ストリップに分解する方法 (4 節) と，遺伝的アルゴリズムを用いて三角形ストリップを併合する方法 (5 節) について述べる．6 節でいくつかの展開図生成例を示したのち，最後に 7 節で本報告をまとめる．

2. 関 連 研 究

3 次元メッシュの展開図生成は，まず理論的な側面から計算幾何学の分野で多くの研究がなされてきている[2]．典型的なよくある問題として，多面体の稜線を切り開いて自己交差なく 2 次元平面に展開できるかいうものがあるが，唯一知られている解は稜線のすべての組み合わせを実際に切断して展開できるかどうかを試すしかない．詳細は，最近のサーベイ論文[3] を参照されたい．

一方実用の観点からは，3 次元メッシュ形状を都合よく変換してから展開する手法が考えられる．得に，展開図生成に都合よく 3 次元形状を近似形状に変換したのち 2 次元展開図に変換する手法は，1990 年代から CAD への応用としてよく研究がされており，可展面集合へと変換されることがしばしばである[4]．CG では，Mitani ら[5] が，入力 3 次元メッシュの形状を近似的に自己交差のない三角形ストリップ集合に展開する先駆的な研究を行った．この手法の提案が契機になり，いくつかの特定の型の可展面集合に 3 次元メッシュを近似展開する手法が提案されている[6],[7]．3 次元メッシュのセグメンテーションも，形状の伸縮をある程度許容すれば，展開図生成手法の一種と考えられる．その中で特に，曲率に基づきメッシュにセグメンテーションを施す手法[8],[9] は，形状の変形をある程度最小化することが可能である．

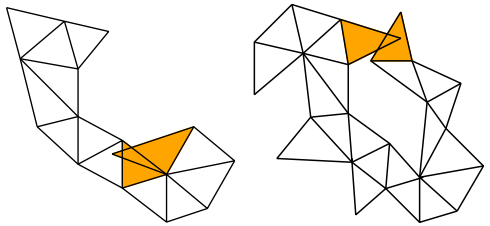
しかし我々の理解する限り，入力 3 次元メッシュ形状を伸縮なしに 2 次元平面に展開する問題は，まだ難しい側面がある．なぜなら，この条件下では展開図の自己交差が生じるため一般的に展開図パッチの個数を制御することが難しい上に，少数の三角形で構成される小さいパッチが生じることが避けられないからである．Straub ら[1] は，3 次元メッシュの双対上で定義される最小全域木を用いた，自己交差のない展開図生成のための興味深い手法を提案している．しかし，最小全域木を制御するための稜線の重み値を適切に選んでも，図 1(a) にあるように，結果として生じる展開図パッチの個数を減らすのはまだ難しいのが現状である．

3. 基本アイデア

3 次元メッシュを平面に展開するとき，一般的に展開図が自己交差をもつため，展開図をさらに多くの連結成分に分割していく必要がある．ここで，自己交差には 2 つの種類がある．ひとつは，隣接している面が交差をもつ**局所的自己交差** (図 2(a))，もうひとつは隣接はしていないが同じ連結成分に属する面の対が交差する**大局的自己交差** (図 2(b)) である．展開図生成ではできればこの自己交差の存在を事前に検出して，実際に 3 次元メッシュを平面に展開するコストの高い幾何計算を避けたい．実は，これは特に局所的自己交差に関しては，各頂点における曲率と切断稜線の個数を考慮に入れることで実現可能である．

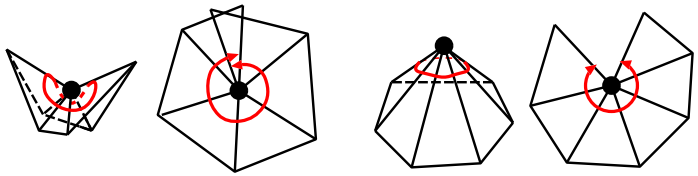
実際には，文献 10) にある通り，メッシュの頂点を負の離散ガウス曲率をもつ鞍点 (図 3(a))

Vol.2011-CG-142 No.16
2011/2/9



(a) (b)

図 2 展開図における自己交差: (a) 局所的自己交差, (b) 大局的自己交差.



(a) (b)

図 3 (a) 鞍点と (b) 非鞍点.

と正の曲率をもつ非鞍点 (図 3(b)) に分類する．ここで離散ガウス曲率は，2π(360 度) から頂点に隣接する隣り同士の稜線がなす角の合計を差し引いたものと定義する．図 3 から明らかな通り，鞍点では隣接する面が合計で 2π 以上の角度をなすため，少なくとも 2 つの切断稜線を導入しないと局所的自己交差を避けることができない．逆に，非鞍点では 1 つの切断稜線だけ導入できればよいことになる．

上記のことを考慮に入れると，3 次元メッシュの面を切断したり接合したりする際に，各頂点における切断稜線の現在の個数を適切に更新していくことで，局所的自己交差を適切に避けることができる．本手法では，この各頂点における切断稜線の個数を明示的に管理するためのデータ構造を導入し，局所的自己交差の存在を平面に実際に 3 次元メッシュを展開することなく，即座に見分けられるようにしている．特に，この機構は最初 3 次元メッシュを分割して得られた三角形ストリップを接合していく際に，局所的自己交際を回避する重要な手立てとして利用していく．一方大局的自己交差に関しては，特に効果的に識別する手段がないので，実際に平面に展開してみて自己交差が生じないかを検出する．これについても，各展開パッチのバウンディングボックスなどを用いることで前処理としての粗い交差チェックが可能であり，実際の展開に必要な幾何計算時間をかなり短縮することができる．自己交差の検出には，文献 1) に提案されている手法を利用している (4 節参照)．

以降，仮定として入力される 3 次元メッシュは，手で作る紙工作の便を考えて，11) の手法を用いて面数が 500 個程度のメッシュにあらかじめ簡単化することとする．しかしながら，実際の結果ではそれ以上の面数を持つ 3 次元メッシュでも，本手法が問題無く動作することを示していく．

4. 3 次元メッシュ分割

2 節で述べた通り，Straub ら[1] は，入力 3 次元メッシュの双対の最小全域木を計算することで，展開図を生成する手法を提案した．しかしながら彼らの手法は，結果として生じる展開図を自己交差なしには平面に展開できない．本手法は，最小全域木を 3 次元メッシュをいくつかの三角形ストリップに分解する手段として用いる．これにより，分解された展開図パッチを接合する際の自由度を確保し，平面に 1 つの連結成分として展開図を構成することを可能にしていく．ここで，メッシュを単一の面要素にまで分割することも考えられるが，のちの展開図パッチを接合する際の計算量を低く抑えたいこと，実際三角形ストリップ分割から 1 つの連結成分を構成できることから，最初に三角形ストリップに分解する選択肢を採用している．

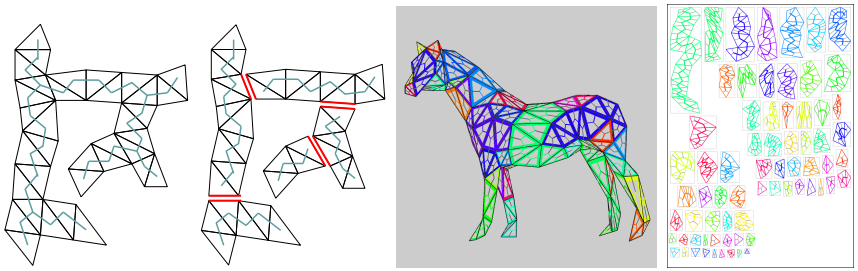
4.1 最小全域木の構築

3 次元メッシュの三角形ストリップへの分割処理は，まずメッシュの双対グラフを覆う最小全域木の計算から始まる．その際，メッシュの双対グラフの稜線に適切な重み値を割り振る必要がある．ここでは，メッシュの切断稜線の長さの和が最小となるような，**最小周長**[1] の稜線重み付けを採用する．今，入力 3 次元メッシュの稜線の長さの最小と最大を $l_{\min}, l_{\max}$ とそれぞれ定義する．この時，最小周長を実現する双対稜線の重み付けは，対応する入力メッシュの稜線長 l を用いて，$w(e) = (l_{\max} - l)/(l_{\max} - l_{\min})$ と書くことができる．

4.2 三角形ストリップへの分解

最小全域木を計算したのち，三角形ストリップ集合を初期の展開図パッチ集合として計算する．これは，最小全域義により得られる面の接続関係を，とある葉 (木の先端部分) から走査し分岐部分に出くわしたら，1 つの分岐部分で切断を施すことで実現していく．図 4(a) はその様子を示しており，分岐の部分では 4.1 節の最小周長の考え方にならって，2 つの分岐部分のうち稜線の短い方を切断していく．図 4(b) は，このような分解操作で得られる，horse モデルの最初の三角形ストリップ集合を示している．

Vol.2011-CG-142 No.16
2011/2/9



(a) (b)

図 4　メッシュの三角形ストリップへの分解: (a) 最小全域木により展開された最初の面の接続関係を分岐部分を分割することにより三角形ストリップに分解. (b) 実際に horse モデルを三角形ストリップに分割した結果.

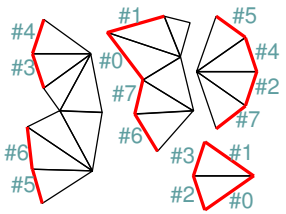


図 5　接合可能稜線

5. 展開図パッチの接合

本手法のおける次の作業は，展開図パッチの個数を減らしていくために，対応する境界稜線にそってパッチ対を順々に接合していくこととなる．ここで，展開図パッチを接合する境界稜線を適切に選択する必要がある．

5.1 接合可能な境界稜線の選択

まず，異なる展開図パッチの接合を試みる境界稜線の選択方法について記す．3 節で説明した通り，我々の定式化では，局所的自己交差を避けられない境界稜線を瞬時に識別することができる．このことは，それぞれの境界稜線の端点における曲面曲率の種類 (鞍点か非鞍点か) とともに現在の切断稜線数を調べ，局所的自己交差が生じないものを選ぶ必要がある．この様な境界稜線を，ここでは**局所的接合可能稜線**と呼ぶことにする．図 5 は，展開図パッチの例とそれに付随する局所的接合可能稜線を識別子とともに示したものである．もちろん，局所的接合可能稜線も大局的自己交差の原因にはなるが，それでもなお接合を検討すべき境界稜線の個数はかなり減らすことができ，実質的にも展開図の探索空間を大きく縮小することができる．

5.2 接合可能稜線の順番の符号化

次にどのように複数に分解されている展開図パッチを接合し，最適な展開図を構成するかについて検討する．十分に展開図のレイアウトを最適化するために，ここでは展開図パッチを接合する境界稜線の順番を符号化することをおこなう．これは，どの展開図パッチ対を順番に接合していくかを表現することに他ならない．本提案手法では，**遺伝的アルゴリズム**を用いて，この境界稜線の最適な順番を探索していく．具体的には，図 6(a) にあるように，境界稜線の識別子を染色体と考え，その順番を遺伝子配列として遺伝的アルゴリズムを用いた計算に利用していく．

実際の計算では，最初ランダムに稜線の識別子を並べた遺伝子配列の集合を最初の世代として準備する．しかし，遺伝子を構成する各境界稜線が，大局的自己交差も含め展開図パッチの接合において自己交差を生じないかどうかを見て，その順番の再度の並べ替えをおこなう．仮に，図 6(a) の上にあるような遺伝子配列があるとする．白色の箱に書かれている稜線識別子は自己交差なく接合可能なものを表すのに対し，灰色の箱にあるのは自己交差が生じる稜線識別子に対応する．ここでは，図 6(a) の下にあるように，実際に大局的自己交差も含めて自己交差なく接合できる稜線識別子は遺伝子配列の先頭の方に移動し，そうでないものは配列の末尾の方に移動することにする．これは，展開図パッチが自己交差なく接合しやすい稜線を先に処理することで，自己交差チェックのための幾何計算の量を減らすことができるとともに，次の 5.3 節で説明されるような交叉や突然変異を伴う遺伝子の進化的計算を，処理しやすくしてくれる．我々の実装においては，この遺伝子配列の並べ替えは，5.4 節で述べられる各遺伝子配列の適応度の計算が行われる際に行われる．注意したいのは，最初局所的接合可能として判定された境界稜線が，展開図パッチの接合の過程で局所的自己交差で (局所的にも) 接合不可能になる場合がある．しかしこれは，その稜線の端点における切断稜線の個数を適宜更新していけば容易に検出可能であり，接合で生じる局所的自己交差は依然として実際の幾何計算なしに事前に見分けることが可能である．

5.3 交叉と突然変異

最初の遺伝子配列集合を用意したあとは，交叉と突然変異の操作を施して次の世代の遺伝子配列集合を生成する必要がある．本手法では，これらの 2 つの操作を，提案するアルゴリズムに適合させるために新たに定式化を行った．

Vol.2011-CG-142 No.16
2011/2/9

まず交叉を適用する 2 つの遺伝子配列 p_1 と q_1 があり，図 6(b) にある通り，局所的かつ大局的自己交差なく展開図パッチが接合可能な稜線と不可能な稜線が既に分類済みであるとする．次の 2 つの遺伝子配列の接合可能であった稜線識別子集合の共通部分を抽出し，相対的な順番は保ちながら共通部分の遺伝子群 (この場合 $\{\#4\}$) が先頭に来るように，それぞれの遺伝子配列を並び替える．さらに，接合不可能であった稜線の共通部分 (この場合 $\{\#2, \#3, \#5, \#7\}$) が遺伝子配列の最後に来るように同じ操作を施し，p_2, q_2 とする．ここで，どちらにも含まれない残りの稜線識別子は，図の緑色の破線で囲まれるように，遺伝子配列 p_2, q_2 の中心部分に来ることになる．実際に，我々の交叉処理はこの部分の遺伝子群に限定して適用され，対応する始点と終点がこの範囲から選ばれる．図 6(b) の例では，交叉は青色の破線で囲まれた配列に対して適用され，結果として p_2 の#6 が q_2 の#0 と置き換わり，さらに稜線識別子が一度だけ遺伝子配列に現れるように保証するため，p_2 の#0 は #6 に変更される．同様の操作が q_2 にも施され，結果として交叉の操作を施したあとの遺伝子配列は，p_3 と q_3 となる．このあと，交叉の最初に行った並べ替えと逆の並べ替えを遺伝子配列 p_3, q_3 に施し，p_4, q_4 にあるように元の遺伝子の順番を回復する．突然変異については，図 6(c) にあるように，単に接合可能な遺伝子群と不可能な遺伝子群から 1 つずつの遺伝子を選び，それを交換することで実現する．

我々の実装では，この世代交代の操作を，世代の一番最適なスコアが収束するまで行う．この 2 つの操作において，接合可能な遺伝子と不可能な遺伝子を意図的に入れ替えるようにしているのは，新しい遺伝子配列が以前と同じ 3 次元メッシュの展開図に対応するのを避けたいためである．また，新しく生成された遺伝子配列も適応度を評価される際には，5.2 節で記述されているような遺伝子配列の並べ替えを行うことに注意する．このようにして新しく定式化した交叉と突然変異の操作は，有効に既存の遺伝子配列からよいものを作り出すことができ，結果として遺伝的アルゴリズムを用いた最適化計算を高速にしてくれる．

5.4 展開図パッチの適応度の評価

もう一つの重要な要素として，展開図パッチ集合を表す遺伝子配列の適応度を評価する目的関数の定義である．本手法においては，展開図パッチの連結成分が最小化できるように目的関数を定義している．実際には，目的関数 f は，

$$f = N_{\mathrm{p}} - 1/A_{\mathrm{s}} - T_{\mathrm{l}}/T_{\mathrm{t}},$$

のように定義している．ここで，N_{p} は，展開図における連結成分数，A_{s} は，メッシュの展開に必要な紙の大きさ，T_{l} は，一番大きい展開図パッチに含まれる面数，T_{t} は，入力 3 次元メッシュの全体の面数を表している．3 番目の項は，なるべく展開図パッチに含まれる面数を偏らせ，2 つのパッチに均等に面が分類されて平衡状態にならないようにするため導入されている．これは，小さい展開図パッチが大きいパッチに接合しやすいという，我々の観察結果にも符合する．展開図パッチの紙面へのレイアウトは，Igarashi ら[12] により提案されているアルゴリズムを用いている．この適応度の定義により，我々が定式化した遺伝的アルゴリズムは展開図の進化的計算による最適化を行うことができる．図 1(b) は，このようにして得られた horse モデルの展開図を示している．

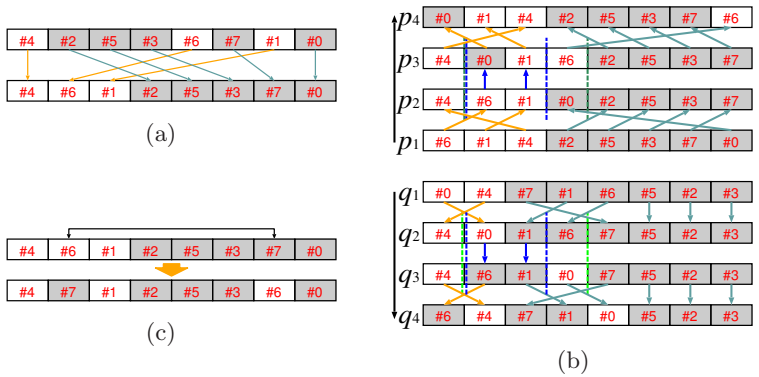


図 6 遺伝子配列の操作: (a) 並べ替え．(b) 交叉．(c) 突然変異．白色の箱は展開図パッチの接合に成功する境界稜線の識別子，灰色の箱は失敗する識別子に対応する．

6. 結　　果

図 7 は，面数が 300 から 1000 程度まで 3 次元メッシュの展開図生成の例を示している，計算時間は，3 次元メッシュモデルの面数に応じて長くなり，CPU が Intel Core i7 CPU (2.80GHz, 8MB cache)，メモリが 8GB RAM の PC を用いて，面数 500 程度までで 1〜2 分，面数が 1000 近くになると 10 分程度かかる．我々の遺伝的アルゴリズムによる計算では，各世代を 32 の遺伝子配列で構成し，その半分を交叉や突然変異などで生成した遺伝子配列群で置き換えている．ここで，交叉と突然変異を起こす割合はそれぞれ 0.8 と 0.1 に設定している．図 8 は，実際にこれらのモデルを紙工作で構築した例を示している．

7. お わ り に

本報告では，3 次元メッシュを平面に展開する新しい手法を提案した．本手法の基本アイ

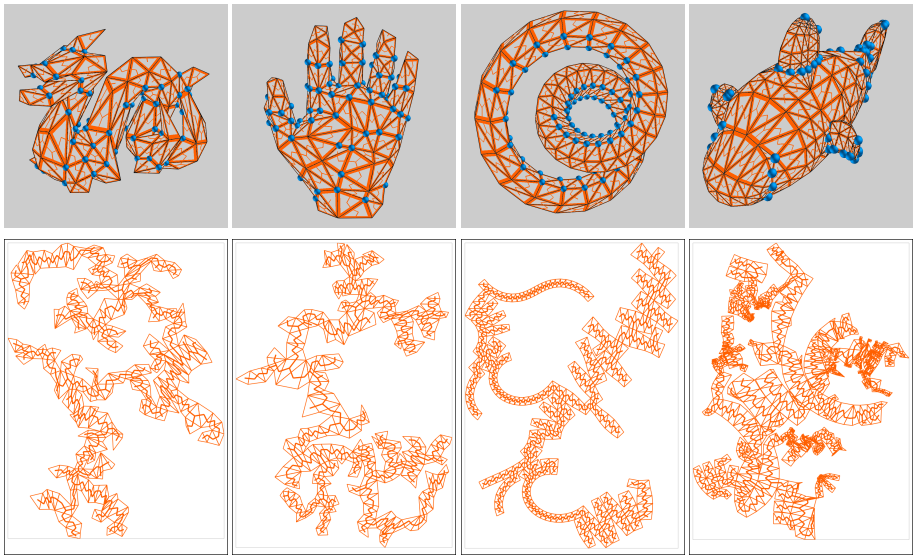


図 7 展開図の生成例．上は入力 3 次元メッシュ，下は対応する展開図パッチ．橙色の全域木は面の接続関係を表し，青の球は鞍点を示す．

図 8 実際の紙工作による 3 次元メッシュの構築．

デアは，メッシュの頂点をそこでの曲率に応じて鞍点と非鞍点に分類し，付随する切断稜線の個数を適切に管理することで，展開図における局所的自己交差を即座に識別し取り除くことにある．展開図は，3 次元メッシュをまず三角形ストリップに分解したのちに，それらの接合の仕方を遺伝的アルゴリズムを用いて最適化することで構成される．

参 考 文 献

1) Straub, R. and Prautzsch, H.: Creating Optimized Cut-Out Sheets for Paper Models from Meshes, *Proc. SIAM Conference on Geometric Design and Computing 2005* (2005).
2) Shephard, G.C.: Convex Polytopes with Convex Nets, *Mathematical Proc. Cambridge Philosophical Society*, Vol.78, pp.389–403 (1975).
3) Demaine, E.D. and O'Rourke, J.: A Survey of Folding and Unfolding in Computational Geometry, *Combinatorial and Computational Geometry*, Vol.52, Cambridge University Press, pp.167–211 (2005).
4) Pottman, H. and Farin, G.: Developable Rational Bézier and B-spline Surfaces, *ACM Trans. Graphics*, Vol.12, No.5, pp.513–531 (1995).
5) Mitani, J. and Suzuki, H.: Making Papercraft Toys from Meshes using Strip-based Approximate Unfolding, *ACM Trans. Graphics*, Vol.11, No.3, pp.259–263 (2004).
6) Shatz, I., Tal, A. and Leifman, G.: Paper Craft Models from Meshes, *The Visual Computer*, Vol.22, No.9, pp.825–834 (2006).
7) Massarwi, F., Gotsman, C. and Elber, G.: Papercraft Models using Generalized Cylinders, *Proc. Pacific Grahics 2007*, pp.148–157 (2007).
8) Julius, D., Kraevoy, V. and Sheffer, A.: D-Charts: Quasi-Developable Mesh Segmentation, *Computer Grahics Forum*, Vol.24, No.3, pp.581–590 (2005).
9) Yamauchi, H., Gumhold, S., Zayer, R. and Seidel, H.-P.: Mesh Segmentation Driven by Gaussian Curvature, *The Visual Computer*, Vol.21, No.8–10, pp.659–668 (2005).
10) Bern, M. W., Demaine, E. D., Eppstein, D., Kuo, E. H.-S., Mantler, A. and Snoeyink, J.: Ununfoldable Polyhedra with Convex Faces, *Computational Geometry Theory and Applications*, Vol.24, No.2, pp.51–62 (2003).
11) Garland, M. and Hackbert, P.S.: Surface Simplification using Quadric Error Metrics, *Proc. ACM SIGGRAPH 1997*, pp.209–216 (1997).
12) Igarashi, T. and Cosgrove, D.: Adaptive Unwrapping for Interactive Exture Painting, *Proc. Symp. Interactive 3D Graphics 2001*, pp.206–216 (2001).