Origami Pop-up Card Generation from 3D Models Using a Directed Acyclic Graph

DER-LOR WAY[1], YI-SHAN TSAI[2] AND ZEN-CHUNG SHIH[2]
[1]*Department of New Media Art*
Taipei National University of Arts
Taipei, 112 Taiwan
[2]*Department of Computer Science*
National Chiao Tung University
Hsinchu, 300 Taiwan

Origami is a paper art used to create three-dimensional (3D) objects by cutting and folding a single sheet of paper. Origami is labor-intensive and requires a high skill level to generate two-dimensional (2D) objects that pop-up into realistic 3D objects. However, this special feature makes designing an origami architecture procedure challenging. This paper provides a novel algorithm to create an origami paper card from a 3D model with a user-specified folding line. The algorithm segments the 2D shape from a 3D model and creates layers of the pop-up paper card using a directed acyclic graph. After applying layers to the layout, the algorithm creates connections between layers, and then outputs the origami layout. Based on a directed acyclic graph, the algorithm computes a class of paper architectures containing two sets of layers and connections that approximate the input geometry while guaranteeing that a pop up card is physically realizable. The proposed method is demonstrated with a number of paper pop-ups, and physical experimental results are presented.

Keywords: origami architecture, pop-up card, planar layout, directed acyclic graph (DAG), computer art

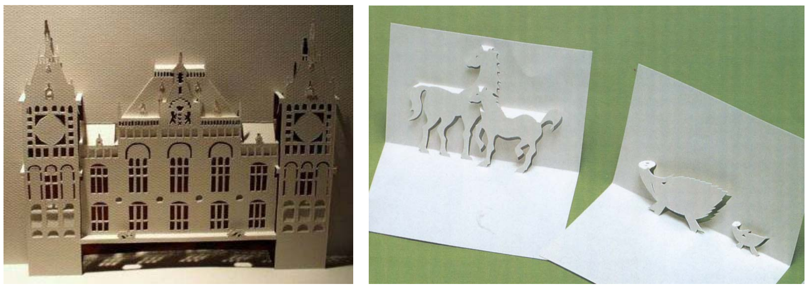
1. INTRODUCTION

The technique for making paper was developed by Cai Lun during the Eastern Han Dynasty, 114 AD. After papermaking spread to Europe in the 13th Century, European artists began developing paper sculpture in the mid-18th Century. A planar paper can be transformed into a three-dimensional object via cutting, crimping, folding, and gluing. Origami is a traditional Japanese paper art started in the 17th Century, which was popularized outside Japan in the mid-1900s. An origami architectural is a 3D structure "pops up" from a folded sheet of paper when opened at 90° [1, 2]. The structure is made by cutting and folding paper without gluing or splicing, which is stored by folding the two halves of the paper closed. An origami architectural structure is a type of pop-up card that appears in many forms, such as extremely realistic greeting cards [14]. Fig. 1 shows two examples created by Chatani [4]. In China, people send greeting cards during Chinese New Year. Chinese zodiac animals are the most common cards.

An origami architectural object is similar to pop-up books, but applies additional constraints to cut and fold patterns on the paper; this is called a planar layout. Some

Received October 17, 2011; revised December 30, 2011; accepted February 1, 2012.
Communicated by Yung-Yu Chuang.
[*] The authors would like to thank the National Science Council, Taiwan, for financially supporting this research under Contract No. NSC 101-2221-E-119-002.

books introduce on the mechanism of designing pop-up crafts [1-5]. Creating two-dimensional layouts that pop up into a desired 3D object is very difficult, labor-intensive, and demands a high skill level. The two major topics associated with origami architecture are "building" and "creature" (Fig. 1). Pop-up cards with building object are regular and easy computing the vertical and horizontal features by software tool. Conversely, pop-up cards with animals or flowers are difficult to create because creatures and flowers are having smooth curves. Additionally, origami animals can be more attractive than buildings.


(a) Building pop-up card. (b)Animal pop-up card.
Fig. 1. Paper Architectures by Masahiro Chatani.

A number of computer-aided tools have been developed [11, 13, 20, 25]. A user is responsible for deciding where and how cuts and folds should be placed on a piece of paper. Li *et al*. generated automatic paper architecture from 3D models [22]. Their algorithm performed best when an input model contained mostly horizontal and vertical planes (Fig. 2). However, the resulting bunny model contained large distortions and omitted some important features (Fig. 3). An organic model must express the shape of a creature with a limited number of curves and layers. Buildings generally have a regular structure with clear layering. Unlike buildings structures, creatures are comprised of smooth curves, such that designing a pop-up animal card is more challenging than creating pop-up building cards. This paper provides a novel algorithm that transforms 3D organic models into 2D layers, and applies the 2D layers onto an origami architecture, finally generating the pop-up card layout.

The main contributions of this investigation are as follows:

(1) This paper presents a directed acyclic graphic (DAG) formulation for planar layouts that can stably pops-up to a paper architecture
(2) A novel algorithm is applied for constructing origami architecture following the creation process by an artist.
(3) The proposed algorithm generates a foldable layout of origami architecture according to features of an input 3D model.

The remainder of this paper is organized as follows. Section 2 reviews work related to origami architecture. Section 3 introduces the design process of the proposed method and the DAG formulation. Sections 4, 5, and 6 describe the process of shape segmentation, 2D pattern layer conversion and foldable layout generation, respectively. Section 7 demonstrates the effectiveness of the proposed method. Finally, Section 8 gives conclusions and suggests directions for further research.

2. RELATED WORKS

2.1 Paper Crafting

Paper cutting, origami, and strip modeling are paper crafts. Paper cutting, a traditional Chinese art, cuts stylistic patterns using the symmetry of patterns and paper overlapping. Xu developed a simple and efficient algorithm for automatically generating a paper-cutting pattern from a given image [31]. Li extended the 2D paper-cutting scheme into 3D paper cuts, and generated animation with paper cuts in an interactive design [21]. Origami, a traditional Japanese folk art, generates an object by folding paper without damaging the paper. Hull generated an origami method using a piece of paper without cutting or gluing [15]. Folding algorithms and the fold ability of paper have garnered considerable attention in the field of computer geometry [8]. Tachi applied an algorithm that automatically generates arbitrary polyhedral surfaces [29]. For curved folding, which is much more restrictive than conventional origami, Kilian proposed an algorithm that generates curved folds automatically based on an analysis of a developable surface [17]. Paper modeling produces 3D models using developable patches or strips. Mitani developed a method for modeling the surface of a model using strips of paper [26]. Shatz offered a paper crafting algorithm for segmenting a mesh into developable approximations that easy to cut and be glued together [28]. Other methods that use mesh simplification for paper modeling were developed by Garland, Cohen, and Wei [6, 10, 30].

2.2 Paper Architecture

Most work on pop-up crafting used a computer-aided environment to design pop-up cards. Matini introduced a computer-aided origami architecture system based on the concept of CAD [25, 26]. Although this system ensures that the output planar layout is foldable, it cannot ensure that the output layout is stable. A similar system is the Popup Workshop for children designed by Hendrix and Eisenberg [13]. Hara *et al.* considered a 2D version of the pop-up problem and generated an automated solution involving polygonal subdivision [12]. Glassner developed a system that allows users to design *V*-fold cards interactively [11]. Li *et al.* developed an interactive tool for designing *v*-style pop-ups and an automated construction algorithm from a given geometry [23].

Algorithmic solutions for automatically generating pup-up cards have been achieved in recent years. Li *et al.* designed an algorithm that automatically generates an origami architecture, and ensures that the layout can be erected stably [22]. Fig. 2 shows some examples. However, Li's method only approximates model shape using two directions of patches under origami architecture constraints, and does not work well for organic models that need to retain important features (Fig. 3). Notably, the bunny model contains large distortions and lacks some important features. This paper proposes a novel method for creating an origami architecture by considering the characteristics of an organic model.

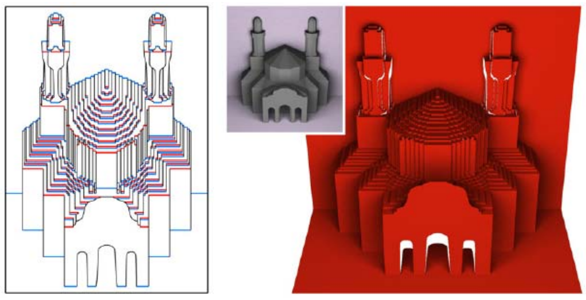


Fig. 2. Li's method for architecture.

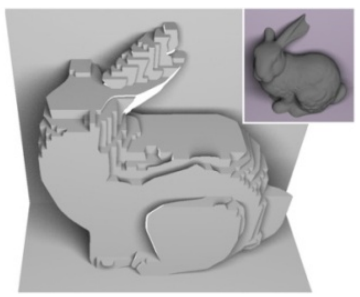


Fig. 3. Li's method for non-architecture.

2.3 Shape Abstraction

Many methods can approximate 3D models for shape abstraction. Segmenting meshes into natural regions is useful for model understanding and shape abstracting applications. Shamir presented a survey of segmentation and partitioning techniques of boundary meshes [27]. They formulated the segmentation problem as an optimization problem and identify two primarily distinct types of mesh segmentation, namely part-type segmentation and surface-type segmentation. In part-type segmentation, the goal is to segment the object represented by the mesh into meaningful, mostly volumetric and parts. Lai used feature sensitive remeshing to produce a hierarchy of meshes, allowing us to efficiently construct a hierarchical segmentation [18, 19]. Fig. 4 shows bunny's examples. Kalogerakis proposed a data-driven approach to simultaneous segmentation and labeling of parts in 3D meshes [16]. Mehra abstracted 3D models by characteristic curves, and reconstructed the abstracted model via these curves using a novel algorithm [24]. Eisemann developed a view-dependent method for converting 3D models into 2D layers [9]. In this paper, the proposed model layering approach considers a viewer's direction and preserves 2D shape features in the model, such that the origami architecture looks stereo.

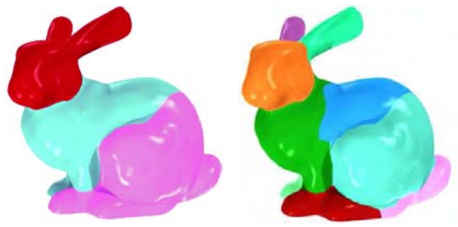


Fig. 4. Lai's feature sensitive mesh segmentation.

3. ALGORITHM FORMULATION

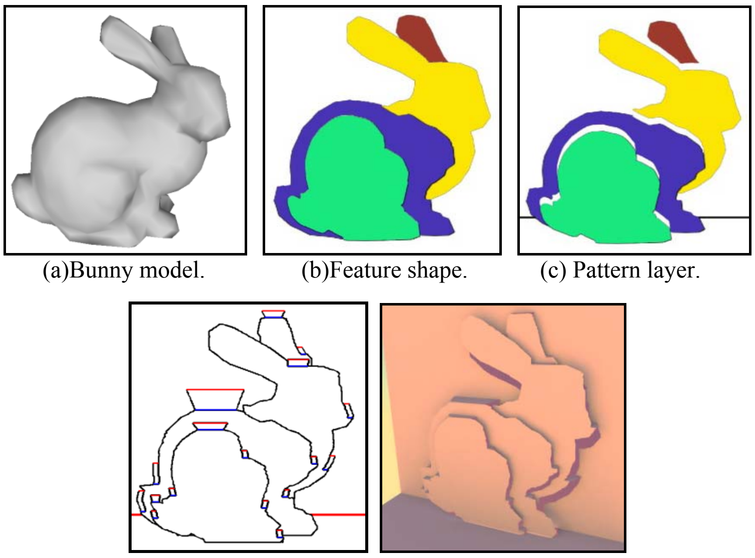
Cuts and folds are used to create a paper architecture from a single piece of paper. The planar of the feature shape, such a paper architecture, consists of a folding line and 2D shape patterns that divide the paper into various regions. Typically, two outer regions, called the backdrop and ground, meet at a fold line. Building architecture models have regular shapes and consist of parallel straight lines or planes. The stability of an origami architecture is defined on patches, which may have two directions. However, for organic models consisting of smooth curves and irregular surfaces, such as a creature whose normal direction varies on the surface of 3D object, the approximation result will not be accurate (Fig. 3). Therefore, this work proposes the concepts of layers and connections. The layers are parallel to the backdrop and represent the shape features of 3D models; the connections are parallel to the ground between two layers.

To construct pop-up planar layouts, this paper propose a novel algorithm for constructing an origami architecture following the creation process. This paper also offers a DAG for planar layouts that can stably pop-up into a paper architecture.

3.1 Design Process for Origami Architecture

Analyzing origami architectures [4], the following steps are used to design origami architecture for organic objects: (1) capture 2D patterns from a 3D object, (2) place the patterns onto layers according to their depths and (3) create connections between layers to generate a foldable layout. The proposed method has three steps: (1) feature shape segmentation, (2) pattern layer conversion and (3) foldable layout generation.

Fig. 5 shows the design flow of the proposed scheme. First, 2D feature shapes are segmented from a 3D model according to their orthogonal projections (Fig. 5(b)). After computing the depth of layers using a depth map, the positions of layers on the layout are determined (Fig. 5 (c)). Connections are then constructed between layers to form an origami architecture, as Fig. 5 (d). Finally, the pop-up card is successfully generated according the foldable layout (Fig. 5 (e)).


(a)Bunny model. (b)Feature shape. (c) Pattern layer.

(d) Foldable layout. (e) Pop-up card.

Fig. 5. Designing process of oigami architecture.

3.2 Directed Acyclic Graph

According to the design process for origami architecture, this paper uses a graph structure to form the planar of the feature shape. After user specified a folding line, Fig. 7 (a) shows the graph of the Bunny's feature shape that constructed from Fig. 6 by Definition 1.

Definition 1: Given a graph $G(V, E)$, where $V = \{v_1, v_2, \ldots, v_n\}$ is a set of vertices and $E \subseteq V \times V$, a vertex v_i is a region on the planar paper and an edge $e(v_i, v_j)$ when they are adjacent neighbor.

The crucial goal is to apply computational algorithms to construct planar layouts that can be popped up. To illustrate the algorithm's design and prove its correctness, this work presents a directed graph for planar layouts, particularly those that can be popped up and are physically realistic.

Definition 2: Given a directed graph $G(V, E)$, where $V = \{v_1, v_2, \ldots, v_n\}$ is a set of vertices and $E \subseteq V \times V$, where $e(v_i, v_j) \in E$ vertex v_j is a successor of v_i, that means Layer L_i is shallower than Layer L_j.

To construct stable origami architecture, the first task is to construct the hierarchical structure of layers. A stable layer in origami architecture should be supported by a ground or a neighboring layer that is shallower than this layer at contacting points [22, 26]. This work constructs a directed graph corresponding to graph of feature shape (Fig. 7 (b)), which records connections between layers. Each directed edge point from node L_i to node L_j indicates that L_i and L_j are neighboring layers and L_i is shallower than L_j. The graph root is ground node G, which represents the shallowest layer, and pseudo backdrop node B represents the deepest layer.

Li *et al.* developed a formal, geometric formulation of planar layouts that pop-up into stable and rigid paper architecture [22]. They discussed the behavior of patches parallel to the backdrop and ground in an origami architecture, and then constructed origami architecture using these patches to approximate the shape of an input model under the stability criterion. If all layers are stable, the origami architecture is stable. According stability rules, the directed graph of a planar layout must be a DAG and meet the following conditions:

(1) If a node is connected with both the ground node and backdrop node, it is stable.
(2) If a node is connected to the ground node or backdrop node and is connected to a stable node, it is stable.
(3) If a node has no fewer than two connections with different stable layers, it is stable.

Definition 3: A DAG is a directed graph $DAG(V, E)$ with no directed cycles.

According to the DAG definition, an edge represents a connection between two layers. The DAG is stable when each node is stable with bread-first traversal node-by-node. For instance, the bunny DAG (Fig. 7 (b)), L_3 is stable with a connected edge by rule 1; L_2 and L_4 are stable with a connected edge by rule 2; and L_1 is also stable by rule 2.

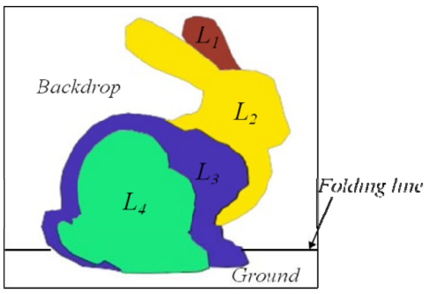
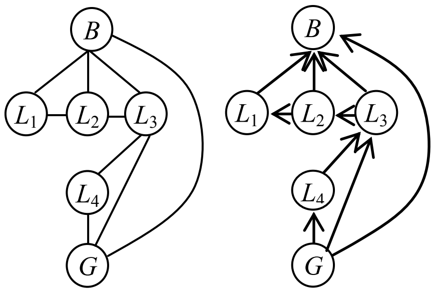


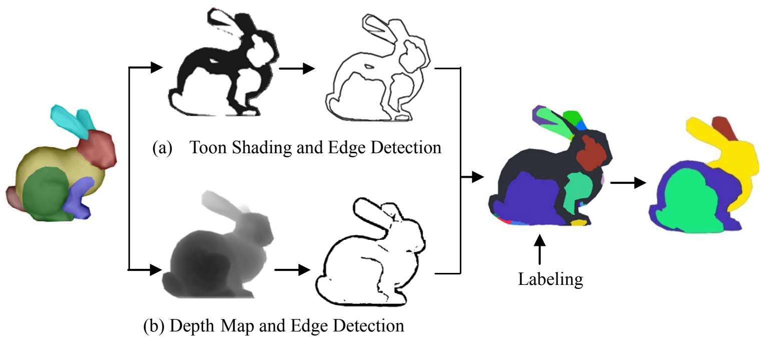
Fig. 6. Feature shape.



(a) Graph of feature shape. (b) Directed acyclic graph.
Fig. 7. DAG formulation of bunny feature shape.

4. FEATURE SHAPE SEGMENTATION

Segmentation and labeling of 3D shapes into meaningful parts is fundamental to shape understanding and processing. Many mesh segmentation techniques are described in section 2. In this paper, the segmentation and labeling of the input model is manually specified. Normally for creature objects are labeled into head, neck, torso, hand (or wing), leg, tail, ..., *etc*. In order to segment origami architectural layers, each layer has 2D shape features of the object model. Feature shape segmentation is view dependent and separates a labeled model into layers while preserving model features. Fig. 8 shows the processing flow of the feature shape segmentation.



(a) Toon Shading and Edge Detection
Labeling
(b) Depth Map and Edge Detection

Fig. 8. The processing flow of the feature shape segmentation.

First, the model is rendered using toon shading in orthogonal projection with user-specified the direction and position of the light source. Color C of the model is defined as follows:

$$Cs = \begin{cases} 0, & \text{if} \quad v_n \cdot v_l < k \\ 1, & \text{if} \quad v_n \cdot v_l \geq k \end{cases}. \tag{1}$$

Where v_n is the normal of a point on the model surface; v_l is the direction of light; and k is the threshold value in the range of 0–1. Image filtering is applied to detect edges of the toon shading result, and the model image is separated into pieces (Fig. 8 (a)).

Secondly, this work also detects the depth of map edges, and the depth of map shading will be changed as follows:

$$C'_d = \text{clamp}(\beta(\alpha - C_d), 0, 1). \tag{2}$$

Where C_d and C'_d are the current and new magnitude of a pixel in a depth map respectively, and α and β are parameters that change the depth map shading, $\alpha = 4.12$ and $\beta = 0.23$ in Fig. 8 (b).

Then, the image of the input model is separated into segments using toon shading edges, and the erode operation is applied to each segment to remove insignificant pieces. Finally, based on the pre-segmented of labeled 3D model and depth map, each surviving segment is iteratively expanded or merged until segments touch each other or the border of the pre-segmented in labeled model or the detected edge of depth map.

5. PATTERN LAYER CONVERSION

For each feature shape segment obtained in Section 4, the depth value of a shape segment is the average of each pixel's depth value. *Depth*(L_i) = Σ(*each pixel's Depth in* L_i)/*NumberPixels*(L_i). Let *Depth*(G) = 0 and *Depth*(B) = 1. In order to construct DAG, the direction of $e(v_i, v_j)$ is determined by $depth(v_i) < depth(v_j)$. Fig. 7 (b) illustrates the Bunny's DAG from Fig. 7 (a).

To allocate the position of layers onto the layout, this work defines the coordinate systems for a 3D structure and 2D pattern as model coordinate and pattern coordinate, respectively (Fig. 9). Since an origami architecture is made from a single sheet of paper, a one-to-one map exists between a 3D structure and 2D pattern. The coordinate's conversion is achieved using the following equations:

$$\begin{cases} X_{(2D)} = X_{(3D)} \\ Y_{(2D)} = Z_{(3D)} + Y_{(3D)} \end{cases} \tag{3}$$

Where $Z_{(3D)}$ represents the depth value of a layer. Fig. 9 shows the coordinate systems of the 3D origami architecture and 2D layout.

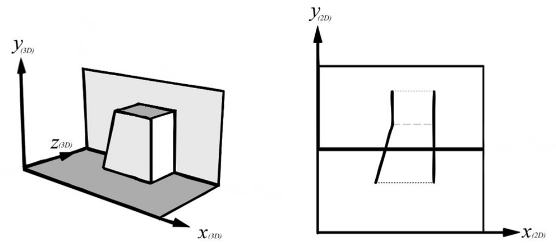
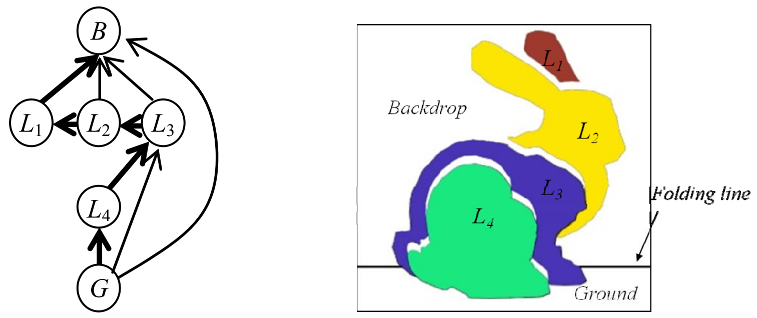


Fig. 9. Coordinates of 3D origami architecture and 2D layout.

The depth of each layer must be transferred into $Z_{(3D)}$ to construct a 3D origami architecture. The new depth value D_{Li} was defined as

$$D_{Li} = D_G + k\Delta d,\ 0 \leqq k \leqq N. \tag{4}$$

Where N is the number of nodes of the longest path in the DAG, except for the backdrop and ground node; k corresponds to the kth node of the longest path and Δd is the thickness of layer.



(a) The longest path of DAG. (b) Pattern Layer.
Fig. 10. Allocating layers onto layout of origami architecture.

Finally, all layers were transferred into the layout according to the relationships between layers and folding line in origami architecture. The position of user-specified folding line represents backdrop depth. As the backdrop is deeper than all other layers, backdrop depth is $D_B = D_G + N\Delta d$. The depth of layer D_G is 0 essentially. Fig. 10 (b) shows the result of allocating layers onto layout according to the result of model segmentation and depth map.

6. FOLDABLE LAYOUT GENERATION

The 2D patterns parallel to the backdrop are layers representing the features of 2D model shapes; and connections are parallel to the ground between two layers. The contour of each shape segment is the connection between two layers. These connections will be put onto the origami architectural layout by taking the overlap of a connection created with Δd by formula (4). Each directed edge $e(v_i, v_j)$ of DAG is a foldable connection except the ground node's edge. All edges $e(G, v_k)$ of the ground node are only for folds. The contour of layer L_i is cut into several fragments based on degree of node v_i. Normally, the number of fragments is equal to or greater than the degree of node v_i. Each out-arrow edge of node v_i will be computed the overlap connections.

To ensure that an origami structure can be folded, two types of directional connections are used (horizontal connections and vertical connections). The types of connections generated are based on the slope of the contour between layers. Fig. 11 shows the slope of contours for a horizontal connection (white color) and vertical connection (dark color) in the bunny model.

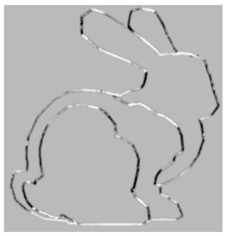
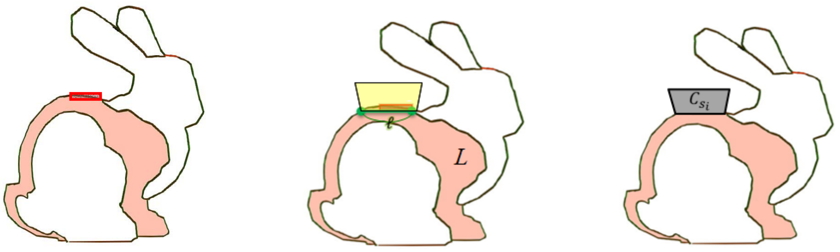


Fig. 11. Slope of contour of bunny model.

6.1 Horizontal Connection

After tracing each layer's node in the DAG, the out-arrow edges of node v_i represents the overlapping connections between two regions. The horizontal connection is created with an $n \cdot \Delta d$ height and as wide as possible, where n is the distance between two nodes in the longest path. For instance, given an out-arrow edge $e(L_3, B)$ of node L_3 and one horizontal connecting fragment extracted from the slope of a contour. A subfragment $frag(L_3, B)$ has the widest axis-aligned bounding box (Fig. 12 (a)). Then a connection area $E(L_3, B)$ is created with height as $3 \cdot \Delta d$ that the slope is smaller than a threshold (Fig. 12 (b)). Finally, there is a small area near the bottom of $E(L_3, B)$ that includes the $frag(L_3, B)$ intersects layer L_3. The horizontal connection $C(L_3, B)$ is generated with a top wider than the intersection area (Fig. 12 (c)).



(a) Widest $frag(L_3, B)$. (b) Connection area $E(L_3, B)$. (c) Horizontal connection $C(L_3, B)$.
Fig. 12. Generating process of a horizontal connection.

This intersection erosion area is defined by the $\Delta Erosion = N(x)/l$, where l is the width of connecting base, and $N(x)$ is the number of pixels of erosion area, where $N(x) = C(L_i, L_j) \cap L_i$. To protect the shape feature, $\Delta Erosion$ must be smaller than a threshold value ε; otherwise, one must shrink the erosion area by reducing length l. $C(L_3, B)$ will be moved upward slightly until $\Delta Erosion$ is smaller than a threshold value ε. After generating a connection, the system examines the layout and eliminates connections according to the following rules:

(1) A connection must not overlap any layer that does not belong to source layer or target layer.
(2) If target layer will be broken into pieces after generating connections, the connection will be eroded horizontally (Fig. 13) to make the target layer continuous.

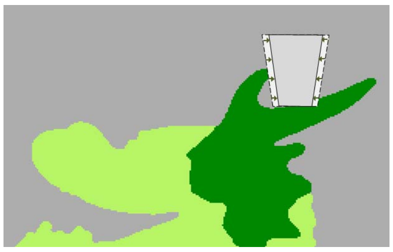
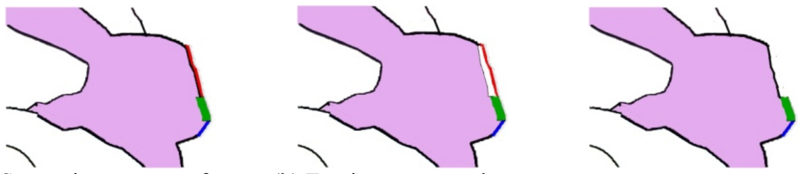


Fig. 13. When target layer (gray) is broken by the connection, the connection will be eroded horizontally to make the target layer continuous.

If the eroded connection touches the source layer and target layer without breaking, the connection is retained; otherwise, the connection will be abandoned.

6.2 Vertical Connection

As the same out-arrow edge of node v_i, the vertical connection will be created with an $n \cdot \Delta d$ height and as narrow as possible. Therefore, this work splits a vertical connecting fragment into three parts: base part; connecting part; and eroding part. Fig. 14 (a) shows an example of fragment separation. The vertical connecting fragment is split into three parts. The parts, from the bottom to the top, are the base part (blue), connecting part (green), and eroding part (red) (Fig. 14 (a)).


(a) Separating process of connecting fragment. (b) Erosion to source layer. (c) Result.
Fig. 14. Generating process of vertical connection.

First, this work eliminates the bottom side of the connecting fragment to ensure that the space of the source layer is under the connection. This work then puts the connection from the bottom of the remaining fragment with a height of $n \cdot \Delta d$ and pre-defined width. Consequently, the eroded part was generated for the rest of the fragment S. The width of eroded part $W(x) = W_c \times h(x) / h_s$, where x is a point of S, h is the height of x from bottom to S, and h_s is the height of S. Fig. 14 shows the process for generating a vertical connection.

7. RESULTS

All input sources are 3D triangular meshes, and output results are 2D layout images. The algorithm is implemented in C++ language using OpenGL and OpenCV. The experiment was carried out on an Intel® Core™ i7 personal computer with a 3GHz CPU and 12GB memory.

Example 1 is a simplified Stanford bunny consisting of 1,068 triangles; Fig. 15 shows experimental result. The experimental result by Li for the Stanford bunny is complex and lacks some features of the bunny [22]. Conversely, the origami architecture in this paper represents features of the Stanford bunny with relatively simpler structure. Example 2 is a simplified Stanford dragon consisting of 4,588 triangles; Fig. 16 shows the experimental result. Example 3 is a horse model; Fig. 17 shows the experimental result. Example 4 is a cow model; Fig. 18 shows the experimental result. Example 5 is a cat model; Fig. 19 shows its experimental result.

The proposed algorithm is flexible; that is, users can design an origami architecture using multiple models. Example 6 puts four models in an origami architecture (Fig. 20).

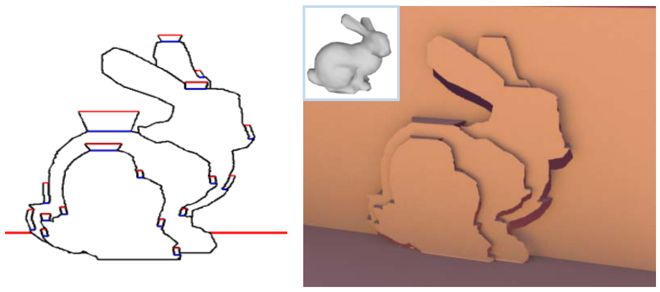


Fig. 15. Origami architecture and layout for Stanford bunny.

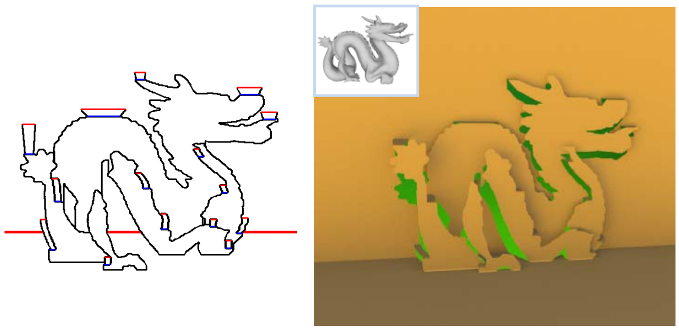


Fig. 16. Origami architecture and layout for Stanford dragon.

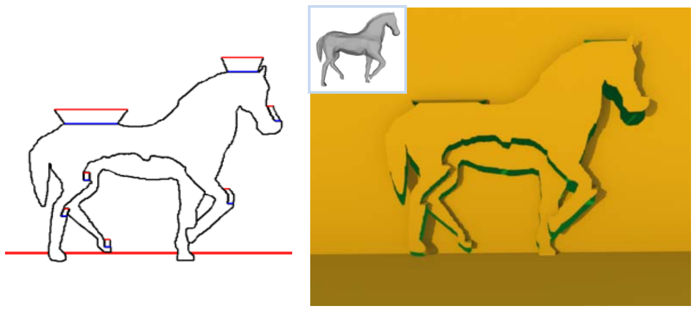


Fig. 17. Origami architecture and layout for horse.

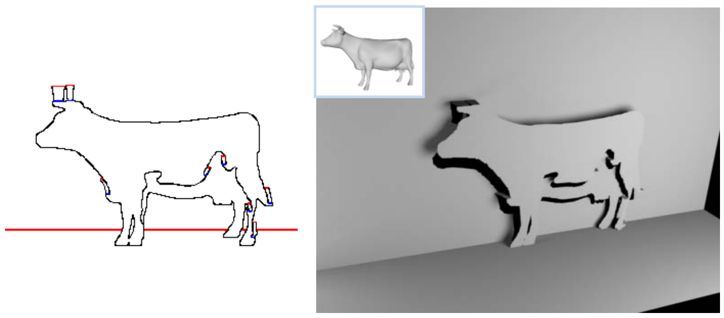


Fig. 18. Origami architecture and layout for cattle.

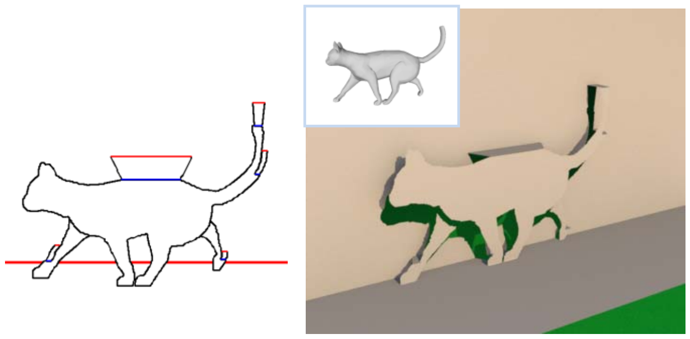


Fig. 19. Origami architecture and layout for cat.

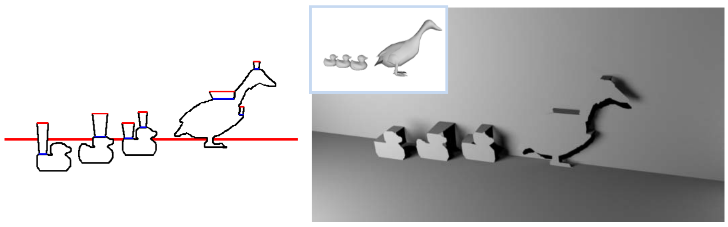
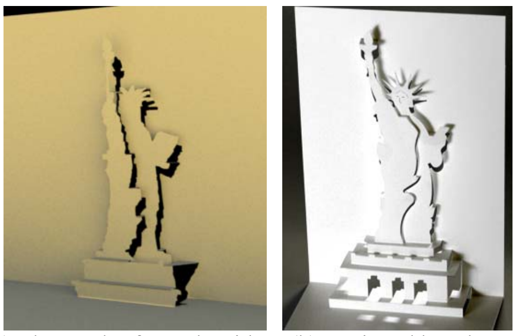


Fig. 20. Origami architecture for multiple models.


(a) The result of our algorithm. (b) Designed by Chatani.
Fig. 21. Two origami architecture of statue of liberty.

The proposed algorithm is also very effective for non-animal models with special shapes. Fig. 21 (a) shows the origami architectural structure of the liberty statue created using our proposed algorithm. Fig. 21 (b) shows the work of Chatani.

8. CONCLUSIONS AND FUTURE WORK

This paper applies a DAG structure to generate an origami architecture that takes a creature's 2D shape feature. This work uses layers and connections, and thereby differs from the work by Li [22] to generate an origami architecture. The proposed algorithm creates an origami pop-up card from a 3D model with a user-specified folding line. First, the algorithm segments the 2D shape from a 3D model and creates layers of a pop-up paper card using a DAG. After adding layers to the layout, the algorithm creates connections between layers, and then outputs the layout of the origami architecture. Based on the methodology of the DAG, the proposed algorithm computes a class of paper architectures containing two sets of layers and connections that approximate the input geometry while guaranteeing that a popup-foldable card is physically realizable.

The algorithm is robust in producing realizable paper architecture and is computationally efficient. A set of examples is presented using architectural models, and real paper crafts are made according to algorithmic results. Thus, users can design an origami architectural structure for an animal without skill or experiment in designing origami architecture. However, the following issues warrant further study.

(1) In cases when an artist has designed an origami architecture, shape features are also considered when generating connections. If the shapes of connections reflect model features, origami architecture will be more attractive.
(2) If an origami architecture has too many layers, input model features will be lost. Therefore, for a stable origami architectural structure that is too complex to construct.

REFERENCES

1. M. Bianchini, I. Siliakus, and J. Aysta, *The Paper Architect*, Crown, NY, 2009.
2. D. Birmingham, *Pop Up! A Manual of Paper Mechanisms*, Tarquin Publications, UK, 1997.
3. D. Carter, *The Elements of Pop-up*, Little Simon, NY, 1999.
4. M. Chatani, *Origami Architecture*, Arira, Tokyo, 1993.
5. C.-M. Cheong, H. Zainodin, and H. Suzuki, *Origami^4: Origami Architecture in the Cartesian Coordinates*, A. K. Peters, Natick, 2009.
6. J. Cohen, M. Olano, and D. Manocha, "Appearance preserving simplification," in *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, 1998. pp. 115-122.
7. D. DeCarlo, A. Finkelstein, S. Rusinkiewicz, and A. Sante, "Suggestive contours for conveying shape," *ACM Transactions on Graphics*, Vol. 22, 2003, pp. 848-855.
8. E.-D. Demaine and J. O'Rourke, *Geometric Folding Algorithms: Linkages*, *Origami*, *Polyhedra*, Cambridge University Press, NY, 2007.
9. E. Eisemann, P. Sylvain, and D. Frédo, "A visibility algorithm for converting 3D

meshes into editable 2D vector graphics," *ACM Transactions on Graphics*, Vol. 28, 2009, Article No. 83, pp. 1-8.

10. M. Garland and P.-S. Heckbert, "Surface simplification using quadric error metrics," in *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive*, 1997, pp. 209-216.
11. A. Glassner, "Interactive pop-up card design, Part 2," *IEEE Transactions on Computer Graphics and Applications*, Vol. 22, 2002, pp. 74-85.
12. T. Hara and K. Sugihara, "Computer aided design of pop-up books with two-dimentional *v*-fold structures," in *Proceedings of the 7th Japan Conference on Computational Geometry and Graphs*, 2009.
13. S.-L. Hendrix and M.-A. Eisenberg, "Computer assisted pop-up design for children: computationally enriched paper engineering," *Journal of Advance Technology for Learning*, 2006, Vol. 3, pp. 119-127.
14. X.-F. Hong, *The Collection of the Paper Crafts*, Three-Mining Culture, Taipei, 1996.
15. T. Hull, "On the mathematics of flat origamis," *Congressus Numerantium*, Vol. 100, 1994, pp. 215-224.
16. E. Kalogerakis, A. Hertzmann, and K. Singh, "Learning 3D mesh segmentation and labeling," *ACM Transactions on Graphics*, Vol. 29, 2010, pp. 1-12.
17. M. Kilian, S. Flöry, Z. Chen, N.-J. Mitra, A. Sheffer, and H. Pottmann "Curved folding," *ACM Transactions on Graphics*, Vol. 27, 2008, pp. 1-9.
18. Y.-K. Lai, Q.-Y. Zhou, S.-M. Hu, and R.-R. Martin, "Feature sensitive mesh segmentation," in *Proceedings of ACM Symposium on Solid and Physical Modeling*, 2006, pp. 17-25.
19. Y.-K. Lai, Q.-Y. Zhou, S.-M. Hu, J. Wallner, and H. Pottmann, "Robust feature classification and editing," *IEEE Transactions on Visualization and Computer Graphics*, Vol. 13, 2007, pp. 34-45.
20. Y.-T. Lee, S.-B. Tor, and E.-L. Soo, "Mathematical modelling and simulation of pop-up books," *Computers & Graphics*, Vol. 20, 1996, pp. 21-31.
21. Y. Li, J. Yu, K.-L. Ma, and J. Shi, "3d paper-cut modeling and animation," *Computer Animation Virtual Worlds*, Vol. 18, 2007, pp. 395-403.
22. X.-Y. Li, C.-H. Shen, S.-S. Huang, T. Ju, and S.-M. Hu, "Popup: automatic paper architectures from 3D models," *ACM Transactions on Graphics*, Vol. 29, 2010, pp. 1-9.
23. X.-Y. Li, T. Ju, Y. Gu, and S.-M. Hu, "A geometric study of *V*-style pop-ups: Theories and algorithms," *ACM Transactions on Graphics*, Vol. 30, 2011, pp. 1-10.
24. R. Mehra, Q. Zhou, J. Long, A. Sheffer, A. Gooch, and N.-J. Mitra, "Abstraction of man-made shapes," *ACM Transactions on Graphics*, Vol. 28, 2009, Article No. 137.
25. J. Mitani, H. Suzuki, and U. Hiroshi, "Computer aided design for origami architecture models with voxel data structure," *Transactions of Information Processing Society of Japan*, Vol. 44, 2003, pp. 1372-1379.
26. J. Mitani and H. Suzuki, "Computer aided design for origami architecture models with polygonal representation," in *Proceedings of the Computer Graphics International*, 2004, pp. 93-99.
27. A. Shamir, "A survey on mesh segmentation techniques," *Computer Graphics Forum*, Vol. 27, 2008, pp. 1539-1556.
28. I. Shatz, A. Tal, and G. Leifman "Paper craft models from meshes," *The Visual*

Computer, Vol. 22, 2006, pp. 825-834.
29. T. Tachi, "Origamizing polyhedral surfaces," *IEEE Transactions on Visualization and Computer Graphics*, Vol. 16, 2009, pp. 298-311.
30. J. Wei, and Y. Lou, "Feature preserving mesh simplification using feature sensitive metric," *Journal of Computer Science and Technology*, Vol. 25, 2010, pp. 595-605.
31. J. Xu, C.-S. Kaplan, and X. Mi, "Computer-generated paper cutting," in *Proceedings of the 15th Pacific Conference on Computer Graphics and Applications*, 2007, pp. 343-350.

Der-Lor Way (魏德樂) received the BS degree in computer science from Soochow University in 1986, MS degree in computer science from Chung-Yuan Christian University in 1988, Ph.D. degree in 2005 in computer science from the National Chiao Tung University. He stayed in Industrial Technology Research Institute (ITRI) to research multimedia and virtual reality for 11 years. Currently, he is an Associate Professor in the Taipei National University of Arts, Taiwan. His research interests are in the area of non-photorealistic rendering, digital art and virtual reality.

Yi-Shan Tsai (蔡宜珊) received the BS degree in Computer Science and Information Engineering from National Central University in 2009, MS degree in computer science from National Chiao Tung University in 2011. Now, She is an engineer in TSMC. Her current research interests are computer graphics.

Zen-Chung Shih (施仁忠) received the BS degree in Computer Science from Chung-Yuan Christian University in 1980, MS degree in 1982 and Ph.D. degree in 1985 in Computer Science from the National Tsing Hua University. Currently, he is a Professor in the Department of Computer Science at the National Chiao Tung University in Hsinchu. His current research interests include procedural texture synthesis, non-photorealistic rendering, global illumination, and virtual reality.