EUROGRAPHICS 2014 / B. Lévy and J. Kautz
(Guest Editors)

Volume 33 (2014), Number 2

Multi-style Paper Pop-up Designs from 3D Models

Conrado R. Ruiz Jr. Sang N. Le Jinze Yu Kok-Lim Low

Department of Computer Science, National University of Singapore

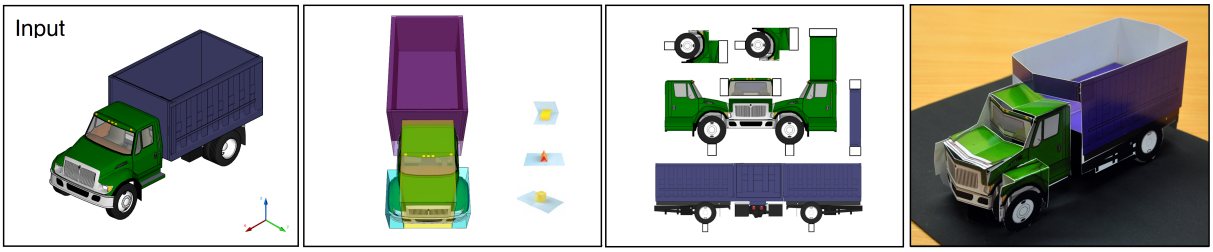

Figure 1: *A multi-style paper pop-up constructed from a design layout that is automatically generated from an input 3D model.*

Abstract

Paper pop-ups are interesting three-dimensional books that fascinate people of all ages. The design and construction of these pop-up books however are done manually and require a lot of time and effort. This has led to computer-assisted or automated tools for designing paper pop-ups. This paper proposes an approach for automatically converting a 3D model into a multi-style paper pop-up. Previous automated approaches have only focused on single-style pop-ups, where each is made of a single type of pop-up mechanisms. In our work, we combine multiple styles in a pop-up, which is more representative of actual artist's creations. Our method abstracts a 3D model using suitable primitive shapes that both facilitate the formation of the considered pop-up mechanisms and closely approximate the input model. Each shape is then abstracted using a set of 2D patches that combine to form a valid pop-up. We define geometric conditions that ensure the validity of the combined pop-up structures. In addition, our method also employs an image-based approach for producing the patches to preserve the textures, finer details and important contours of the input model. Finally, our system produces a printable design layout and decides an assembly order for the construction instructions. The feasibility of our results is verified by constructing the actual paper pop-ups from the designs generated by our system.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Geometric algorithms, languages, and systems

1. Introduction

Paper pop-ups or movable books are three dimensional books that contain paper pieces that pop out or move when the book is opened and fold completely flat when the book is closed. Although now popularly used for children's books, it was historically also employed for illustration purposes in a wider range of topics like philosophy, astronomy, geometry and medicine. Today's pop-up books still continue to fascinate readers of all ages and cultures. Some of the notable artists are Robert Crowther [Cro11], Robert Sabuda [CS03] and David Carter [Car08] (Figure 2).

Recently, there has been much interest in the physical fabrication of 3D models. Paper pop-ups are a practical candidate for this task since they do not require specialized hardware and they can be folded flat for easy storage. Just as algorithms in origami have found applications in protein folding and deploying instruments in space, pop-up algorithms could potentially be used for other applications. Examples include 3D micro-fabrication from 2D patterns and collapsible objects such as foldable furniture.

Creating a pop-up however can be a tedious task even for an experienced designer. It usually entails a trial-and-error

Figure 2: *Sample pop-up books (left to right): Amazing Pop-up Trucks [Cro11], Alice's Adventures in Wonderland [CS03] and Yellow Squares [Car08].*

approach to find a valid configuration of the pieces of paper, so that they resemble a desired structure. A pop-up book may require weeks to design and up to a year to complete. With the proliferation of 3D models on the web and the easy accessibility to 3D authoring software, we propose a fully automated approach for converting 3D models into paper pop-ups.

The only methods that are able to automatically generate pop-up designs are [LSH*10], [LJGH11], [LNLRL13] and [LLLN*13]. In each of these works, each pop-up is made of only a single type of pop-up *mechanisms* (i.e. single-*style* pop-up), and a very specialized method is used to generate the pop-up design. The v-style pop-ups addressed by [LJGH11] seem to be the most versatile in terms of geometry. However, the main focus of [LJGH11] was on the geometric study, and its automatic method can only generate pop-up patches restricted to three perpendicular orientations. As such, it is not able to demonstrate the full potential of the v-style mechanism.

In actual pop-up books created by artists, numerous styles are used to suitably represent different parts of the objects. Our objective is thus to combine multiple styles in a pop-up, and use the most suitable mechanism for each part of the object. Combining multiple styles presents new challenges in the validation of its stability and foldability. In this paper, we provide new geometric conditions for the validity of multi-style pop-ups.

We consider four types of mechanisms, the *step-fold*, *tent-fold*, *v-fold* and *box-fold*. Of these mechanisms, the box-fold has not yet been studied in any previous work. As such we formally include a description of the box-fold and outline the conditions for its validity considering its foldability and stability.

Our method abstracts the input 3D model using suitable primitive shapes that both facilitate the formation of the considered pop-up mechanisms and closely approximate the input model. Each shape is then abstracted using a set of 2D patches that combine to form a valid pop-up.

In the automated approaches of [LSH*10] and [LJGH11], voxelization is used to approximate the input 3D model, which leads to the possible loss of important features. In our work, the shape abstraction allows us to fit a minimal number of 3D primitives to approximate the model, resulting in fewer patches. The final patches are produced using an image-based approach to preserve the textures, finer details and important contours of the input model. Finally, our system produces a printable design layout and decides an assembly order for the construction instructions.

2. Related Work

Papercraft. Since the invention of paper itself, paper crafting has fascinated people and has been the subject of several studies. Mitani and Suzuki [MS04] proposed an automatic method for creating paper toys by simplifying 3D meshes into paper strips. Origami, the traditional Japanese art of paper folding, is another popular type of papercraft that has been extensively studied in literature [Hul06, DO07, O'R11]. In particular, folding algorithms and conditions for foldability are of much interest. Recently, Tachi [Tac10] proposed an algorithm for designing origamic structures automatically from a polyhedral surface. Although sharing one of the central problems of pop-ups, which is foldability, the physical constraints are significantly different for origami. As a result, their formulations cannot easily be used for paper pop-ups.

Mesh Simplification and Abstraction. Automatic generation of paper pop-up designs from 3D models can be considered a form of geometry simplification. [LNLRL13] uses generalized cylinders to approximate 3D models in order to automatically generate lattice-style pop-ups or sliceforms designs. Our approach is inspired by the promising results in shape abstraction using 2D planes [DDSD03, MSM11] and 3D primitives [YK12]. Our approximation aims to preserve the input contours, as they have been shown to capture important shape information of the objects [MZL*09]. Nonetheless, the existing works in shape abstraction cannot be readily applied to our domain, since an arbitrary set of 2D planes may not necessarily be a valid pop-up.

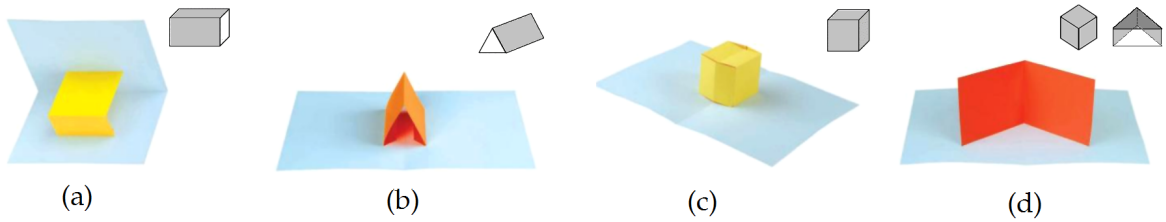

Figure 3: *Pop-up mechanisms and the corresponding 3D primitives: (a) step-fold, (b) tent-fold, (c) box-fold and (d) v-fold. The shaded faces are the principal faces.*

In addition, artists consider the overall structure of an object and its parts to determine which mechanisms to use. As such, our work is also related to semantic segmentation, like the recent work of [vKXZ*13] on the hierarchical analysis of shape structures. Other approaches are detailed in the work of [MWZ*13].

Computational Pop-ups. A number of studies on paper pop-ups have been presented. [LTS96, Gla02a, Gla02b] describe the use of simple geometry to model various pop-up styles. [HE06, LTS96] present interactive design systems that simulate the opening and closing of paper pop-ups. Iizuka et al. [IEM*11] also presented an interactive system that supports v-folds and parallel folds. Hoiem et al. [HEH05] developed an application to create simple pop-ups from photographs based on scene or image understanding [HEH08]. Abel et al. [ADD*12] proposed an algorithm for creating pop-ups by subdividing an input polygon into a single-degree-of-freedom linkage structure.

A few studies have focused particularly on *Origamic Architectures* (OA), a special type of paper pop-up that is made of a single sheet of paper [MSU03,LSH*10,LLLN*13]. One notable recent work is that of [LSH*10], in which they provided a theoretical foundation and a voxel-based algorithm for automatic design of origamic architectures.

[LJGH11] later extended the notion of validity to a more general class called *v-style pop-ups*. They also developed an interactive tool to convert a 3D model to a v-style paper pop-up. Similar to [LSH*10], they used voxelization to approximate the input model, which may cause important features to be lost. In order to capture accurate contours, a very dense voxel grid is required, which then makes the design infeasible to construct in practice because of the excessive amount of cuts and folds. In addition, their v-style pop-ups are limited to only three patch orientations.

Note that most of the computer-aided design systems before [LSH*10, LJGH11] are limited to a few known mechanisms, and do not guarantee the validity of the designs. Research on the geometric conditions for the validity of a general paper pop-up is scarce, primarily because determining the foldability of a general pop-up is NP-hard, as shown in [UT06].

3. Pop-up Fundamentals

3.1. Terminology

A pop-up is created by folding and gluing together pieces of paper. It comprises a set of planar polygons called *patches* that are connected at straight line segments called *hinges* or *fold lines*. A pop-up has to sit between two primary patches called the *ground* and *backdrop*, which are connected by a special hinge called the *central fold*. The interior angle between these two patches is called the *fold angle*.

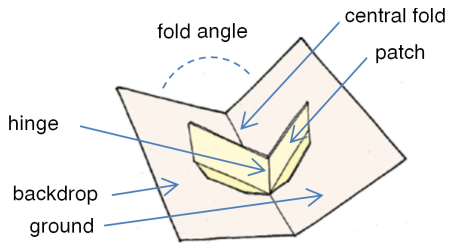

Figure 4: *Basic pop-up terminology.*

We define a *mechanism* as the most basic geometric structure that, together with other structures of the same type, gives a pop-up a unique style. Pop-up artists use several types of mechanisms to make their works come to life. In this paper, we focus on four mechanisms: *step-fold*, *tent-fold*, *v-fold*, and *box-fold*. Among these mechanisms, step-fold structure pops up at 90° fold angle, while the others pop up at 180°. Illustrations of these mechanisms and their 3D primitive pairs, which are used in our algorithm, are shown in Figure 3.

When creating a pop-up, we also require it to be valid, which means it needs to be both *foldable* and *stable*. A pop-up is said to be *foldable* if the structure can fold completely flat when the ground and backdrop patches are fully closed. Note that during the folding process, the rigidity and connectivity of the patches need to be maintained at all times and it should not introduce new fold lines. On the other hand, a pop-up is said to be *stable* if all its patches are stationary when the ground and backdrop patches are held still at any fold angle. In other words, the closing and opening of a pop-up do not need any extra external force besides holding the two primary patches.

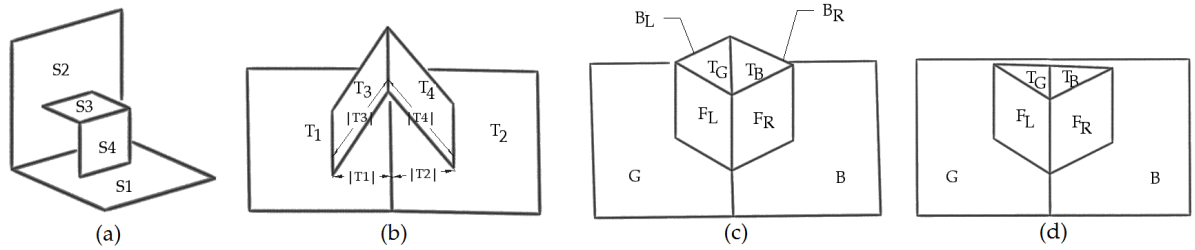

Figure 5: *Pop-up mechanisms and patches: (a) step-fold, (b) tent-fold, (c) type-1 v-fold and (d) type-2 v-fold.*

To facilitate our formulation, we assume that paper is rigid and has no thickness or weight. These assumptions are similar to those used in the previous computational studies of paper art [BH02, LSH*10, LJGH11]. We also use the same formal definitions of foldability and stability as used in [LSH*10, LJGH11].

3.2. Pop-up Mechanisms

In this section, we describe the patch arrangement and constraints in the considered pop-up mechanisms.

3.2.1. Step-fold

Step-fold, which is referred to as mechanism D2 in [LJGH11], comprises 4 patches, S_1, S_2, S_3, and S_4, folded together such that S_1 is parallel to S_3 and S_2 is parallel to S_4 (Figure 5a). Patches S_3 and S_4 are bounded by their cut lines and common hinge.

3.2.2. Tent-fold

Tent-fold, as described in [HS09], is more general than step-fold in that it does not require its patches T_1, T_2, T_3 and T_4 to form parallel pairs, although the fold lines need to be parallel. We construct tent-fold so that $|T_1| = |T_2|, |T_3| = |T_4|$ and $|T_3| > |T_1|$, where $|T_i|$ is the distance between two hinges on T_i (Figure 5b).

3.2.3. V-fold

Our *v-fold* is also known as mechanism D1 in [LJGH11]. However, it may comprise more patches to depict more shapes. In this work, we use two types of v-fold.

In the first type, when the primary patches are opened at 180°, the v-fold structure forms a complete box as shown in Figure 5c. More specifically, our *v-fold* structure consists of 8 patches, G, B, F_L, F_R, B_L, B_R, T_G, T_B, in which we glue $\{B, F_R\}$, $\{G, F_L\}$, $\{F_L, T_G\}$, $\{F_R, T_B\}$, and $\{F_L, F_R, B_R, B_L\}$. When B and G are opened at 180°, T_G and T_B are parallel to B and G, while F_L, F_R, B_L and B_R are all perpendicular to B and G, and form 45° with the central fold. In addition, the hinge between T_G and T_B lies in the bisecting plane of B and G. Note that, in order for the top patches to fold up, we do not glue $\{T_B, B_R\}$ or $\{T_G, B_L\}$.

The second type of v-fold (Figure 5d) is similar to the first type, except that it forms only a triangular half of a box when B and G are fully opened at 180°, and it does not have the back patches B_L and B_R.

3.2.4. Box-fold

Unlike the other pop-up mechanisms, *box-fold* has yet to be formulated in detail in previous work. Box-fold is the most complex of the four mechanisms considered. We choose to include it because of its ability to capture rectangular objects aligned with the central fold line and its common usage in many artworks, such as [Cro11].

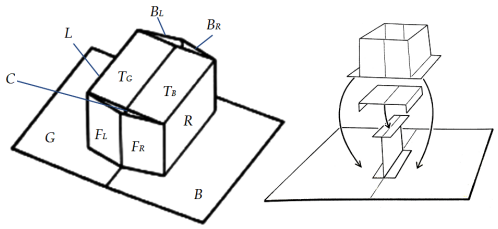

Figure 6: *The box-fold mechanism.*

Consider a pop-up opened at 180°. A box-fold structure comprises 11 patches labeled as G, B, L, R, F_L, F_R, B_L, B_R, T_G, T_B and C (see Figure 6). Patches G and B are parallel to the ground and backdrop. Patches L and R are the left and right sides of the box structure, forming equal angles with G and B, and are equidistant to the central fold. Patch C is a special backbone glued perpendicularly to G and B at their common fold line. On top of the box, patches T_G and T_B are connected to L and R, respectively, and are glued to C at their common fold line. In our work, all the fold lines connecting B, R, T_B, T_G, L and G are made parallel to the central fold.

On the front side of the box, patches F_L and F_R share a fold and are connected to L and R, respectively. Similarly, B_L and B_R are equivalent patches on the back of the structure. The folds between F_L, F_R and between B_L, B_R must be coplanar with patch C. Note that the front and back patches are not glued to C. In principle, only one of the two pairs, either (F_L, F_R) or (B_L, B_R), is needed. However, in practice, box-style pop-ups normally contain both sides for better symmetry and sturdiness. In addition, each side is allowed to have multiple pairs of (F_L, F_R) or (B_L, B_R), as long as their fold lines do not intersect.

3.3. Pop-up Validity

As described in Section 3.1, a pop-up is said to be valid if it is both foldable and stable. In this section, we discuss the conditions for a valid pop-up.

3.3.1. Validity of Single-style Pop-ups

Step-fold, tent-fold and v-fold pop-ups have been proven valid in earlier work [LJGH11, HS09]. We now show that box-fold is also valid.

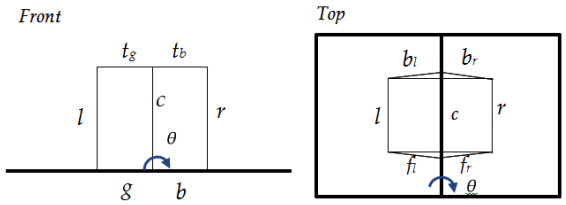

Figure 7: *Two cross sections of a box-fold.*

Foldability of a box-fold. According to [HS09], if a 2D quadrilateral with 4 sides a, b, c and d, in that order, has $a+b=c+d$, then it can fold completely flat when a and c are fully closed. Hence, a box with patches L, R, T_G, T_B, and C is foldable if $b+r=c+t_b$ and $g+l=c+t_g$. In addition, because the folds between F_L, F_R and between B_L, B_R are coplanar with the central fold and bisect the angle between L and R, the front and back patches, F_L, F_R, B_L, and B_R, can also fold completely flat following the motions of L and R.

Stability of a box-fold. We show that a box-fold as constructed in Section 3.2.4 is stable. Assume that the structure is opened at an arbitrary angle in $(0,180^\circ]$. If the angle between C and G may change while G and B are held stationary, then the box will undergo a shearing effect. However, it also contains the front and back patches, F_L, F_R, B_L, and B_R, which keep L and R from shearing. In other words, C cannot rotate while G and B are opened at an arbitrary angle. As a result, a box-fold structure is stable.

3.3.2. Validity of Multi-style Pop-ups

Although each mechanism is foldable, the combined multi-style pop-up may not be foldable, due to the possible intersections between the mechanisms. In particular, during the closing of a pop-up, patches F_L, F_R, B_L, B_R of a box-fold or F_L, F_R of a v-fold emerge and may intersect with the corresponding patches in another box-fold, v-fold, or tent-fold. To make the whole pop-up foldable, it is important to position the mechanisms so that such intersections do not occur.

First, we define a right-handed Cartesian coordinate system, in which the z-axis is perpendicular to the central fold and bisecting the interior fold angle between the primary patches. The x-axis lies along the central fold and the y-axis points in the direction of the ground patch.

Based on the structures defined in Section 3.2, a tent-fold, v-fold and box-fold can only be positioned along the central fold or between T_G and T_B of a box-fold. Step-fold can only lie between a base patch and patch L or R of a box-fold. A base patch can be a primary patch, or T_B, T_G of a box-fold. For instance, we can position a box-fold on top of another box-fold, then a step-fold between patch R of the upper box and T_B of the lower box.

Note that a step-fold always forms parallel pairs of patches, a tent-fold always remains symmetric to the central hinge, and they do not move along the x-axis during folding. Hence, we only need to consider the range of movement of box-folds and v-folds for intersection checking.

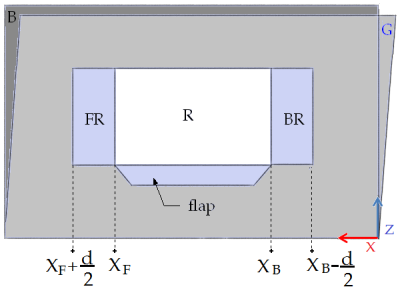

Figure 8: *A fully-closed box-fold.*

Box-fold. We consider a box-fold lying on patches B and G, in which the hinge between F_L and F_R is h_F. Let d be the thickness of the box along the y-axis, and x_F be the x-coordinate of h_F when the box is opened at 180°. Then when the box is fully closed (see Figure 8), the x-coordinate of h_F becomes $x_F+d/2$. Similarly, the x-coordinate of the hinge h_B on the back is $x_B-d/2$ when the box is closed. Hence, to avoid intersection, no other mechanism should be placed on B and G within the range $[x_B-d/2, x_F+d/2]$ along the x-axis.

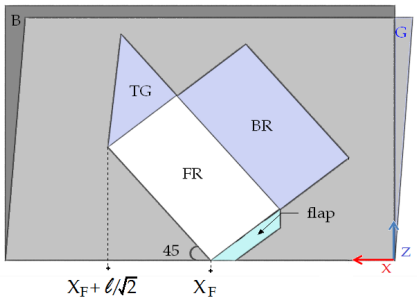

Figure 9: *A fully-closed type-1 v-fold.*

V-fold. In a type-1 v-fold, let x_F and x_B be the x-coordinates of the hinges between F_L, F_R, and between B_L, B_R when the fold is opened at 180°. Note that, when B and G, the two base patches of the v-fold, are being closed, only F_L and F_R still touch G and B, and the v-fold leans toward the positive direction of the x-axis. Hence, the small-

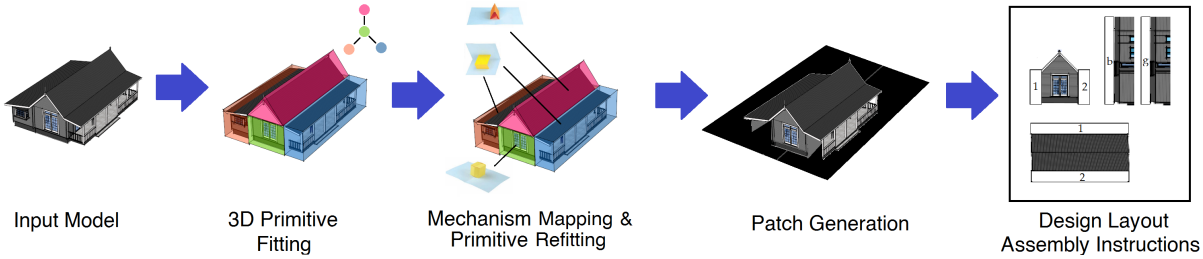

Figure 10: *Overview of the automatic pop-up design algorithm.*

est x-coordinate of the v-fold is x_B. On the front of the v-fold, the intersecting point between F_L, F_R, T_B and T_G has the greatest x-coordinate, which becomes $x_F + l/\sqrt{2}$ at 0° fold angle, where l is the height of the v-fold along z-axis when the fold angle is 180°. To avoid intersection, no other mechanism should be placed on B and G within the range $[x_B, x_F + l/\sqrt{2}]$ along the x-axis. The range for a type-2 v-fold is similar, with x_B being the x-coordinate of the right cut lines of F_L and F_R when the fold is fully opened.

By following the two sufficient conditions above, we avoid intersections between the considered mechanisms and guarantee that the combined pop-up is fully foldable. Furthermore, because all the patches of each mechanism are stable, their resulting multi-style structure is also stable.

4. Automatic Pop-up Design Algorithm

Our algorithm takes a 3D mesh as input, then generates a 2D design layout of a valid paper pop-up. Our main idea is to abstract a mesh into sub-volumes by fitting 3D primitives, and then choose the best mechanism to represent each primitive. Each primitive-mechanism pair has its own set of steps to convert a shape into a set of valid pop-up patches that is guaranteed by our formulation. Once the patches have been generated and stabilized, we produce a design layout and determine an assembly order for the printable instructions.

4.1. 3D Primitive Fitting

First, we align the input model using Normal Principal Component Analysis (NPCA). We obtain the principal axis of the 3D model using the surface normals weighted by the area of the faces. Our aim is to automatically align the model such that the principal axis will point in the same direction as the normal of the faces with the most cumulative surface area. We set this principal axis as our y-axis, and the x- and z-axis to the vectors orthogonal to this axis. It is also possible to allow the user to align the model based on his own preference. Alternatively, we can also use the dominant long edges in the model to determine the alignment to facilitate foldability.

Then we obtain the symmetry plane and the tightest axis-aligned bounding box of the model. The base of the bounding box forms our initial backdrop and ground patches. They

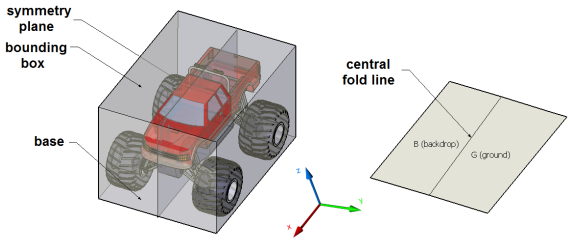

Figure 11: *Model aligned with NPCA and the corresponding bounding box.*

are connected to form the central fold line, where the symmetry plane intersects the base.

We then perform a RANSAC primitive fitting similar to the techniques used for point clouds [SWK07]. However, in our case we fit volumes rather than surfaces, similar to the work of [YK12]. Since some meshes may only have a few sparse vertices, we also employ a preprocessing step to add pseudo vertices to the input mesh in a technique similar to [SLCH11] for grid-based PCA. This ensures that our vertices are evenly distributed on the object's surface.

Our primitives also have some constraints on their orientation and connectivity. We primarily have two basic 3D primitives, which are the rectangular and triangular prisms. Because of the constraints on these primitives, we only need a minimum of two points to specify a rectangular prism and four for a triangular prism. In these prisms, the sides edges are always parallel to one another, while the top and bottom edges form a rectangle or a triangle.

An important part of the RANSAC primitive fitting is the error metric used to judge how well the fitting is. The error of a set of points to a primitive is given by:

$$error_{fit} = outliers * w_1 + dist * w_2 + coverage * w_3 \quad (1)$$

where *outliers* is the percentage of points that are not part of the consensus set of the current model, *dist* is the cumulative normalized distance of the points to their nearest surface on the primitive, and *coverage* is the volume covered by the primitive relative to the bounding box of all the points. The weights w_1, w_2 and w_3 are in the range from 0 to 1 and add

up to 1. They control the influence of each metric to the total error. These weights affect the number of primitives fitted by the RANSAC algorithm. By putting more weight on *coverage* and *outliers*, it will favor bigger volumes that encompass more vertices and in turn produce bigger and fewer patches.

The spatial constraints on our primitives are:

Constraint 1. Angles: Every angle on the rectangular prism must be 90°. Similarly, the angle between a top or bottom edge and a side edge in a triangular prism must also be 90°.

Constraint 2. Orientation: The rectangular prism's four side edges should only be either parallel or orthogonal to the central fold line (x-axis). The triangular prism's three side edges, must similarly be either parallel or orthogonal to the central fold line (x-axis).

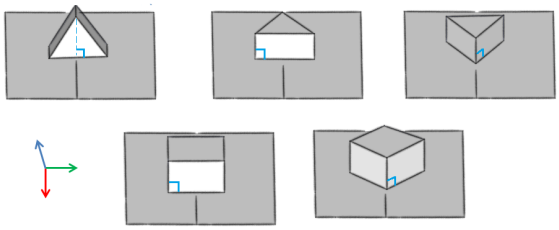

Figure 12: *Valid orientations of the 3D primitives.*

Constraint 3. Placement: A new primitive can only be placed on a *Base Patch Pair*. A Base Patch Pair (BPP) is a set of two patches that share a common hinge and can act as intermediary backdrop and ground patches. For example, in Figure 13, we start with our first BPP, which are the primary patches $\{B,G\}$. Once we add the rectangular prism in the center, three more base pairs are generated: $\{F1,F2\}$, $\{F3,B\}$, $\{F4,G\}$. Adding another rectangular prism on $\{F4,G\}$ generates two more base pairs $\{F4,F5\}$ and $\{F6,G\}$. Finally, we add a triangular prism on $\{F1,F2\}$ pair, and this does not add any new BPP.

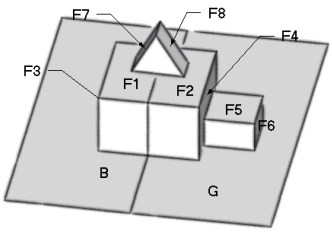

Figure 13: *Base Patch Pairs.* $BPP = \{\{B,G\}, \{F1,F2\}, \{F3,B\},\{F4,G\},\{F4,F5\},\{F6,G\}\}$.

These constraints are based on the possible combinations of the different mechanisms, as described in Section 3.3.2. For example, a triangular prism is permitted on top of a rectangular prism, but not on its side, since such combination may not map to a foldable pop-up.

RANSAC is used to fit the vertices to individual primitives. In order to get the best combination of primitives that satisfy the aforementioned constraints, we formulate it as a combinatorial optimization problem. We then use dynamic programming to minimize the accumulated individual primitive errors ($error_{fit}$).

4.2. Mechanism Mapping and Primitive Refitting

After the 3D primitive fitting step, we now have a 3D model that is abstracted using primitives that best approximate the original shape and at the same time can be mapped to pop-up mechanisms. Using the valid primitive-mechanism pairs shown in Figure 3, we derive Table 1.

Primitive	Step	V-fold	Box	Tent
Rectangular Prism	X	X	X	
Triangular Prism		X		X

Table 1: *Possible primitive-to-mechanism mappings.*

Most of the abstraction and approximation is done in the previous step. Normally, there will only be a few possible mappings of the primitives and we can do an exhaustive search of all the possibilities. We select the combination that minimizes a certain error criterion. Our error criterion is based on the coverage of the mechanism or how well its patches can approximate the primitive:

$$error_{cover} = 1 - (\sum Area_{principal\,face} / \sum Area_{face}) \quad (2)$$

Unlike [LJGH11] that uses all the faces of the voxel as patches, we only use some faces of the primitives called the *principal faces* (the shaded faces in Figure 3). The error is based on the surface area covered by only the principal faces over all the surface area of the faces of the primitives in the structure.

During the mechanism mapping, we also examine the validity of the generated structure based on the conditions in Sections 3.3.1 and 3.3.2. These formulations allow us to test the foldability and stability of the structure without the need for simulations.

The primitives can also be refitted as long as they satisfy the foldability of the mechanisms. For example, the side patches for box-folds do not necessarily need to be orthogonal to the ground or backdrop patches, as long as the lengths of the patches still satisfy $b+r=c+t_b$ and $g+l=c+t_g$, such as the case of the ship in Figures 14 and 15, where the sides of the box are refitted to slant along the body of the ship. In addition, if sufficient gap between different box-folds along a common hinge is not achievable, the front and back patches of the box-folds can be partially trimmed to avoid the intersection.

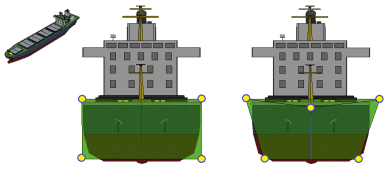

Figure 14: *Refitting a rectangular prism.*

4.3. Patch Generation

Once we know the mapping of the primitives, we now convert each principal face of the primitive into a pop-up patch. Unlike previous approaches that simply consider the entire face as a patch, we use an image-based approach to better approximate the shape and contours of the object. In general, we render an orthographic projection of the mesh unto a plane co-planar with the principal face. This produces an image that becomes the basis of the shape and texture of the patch.

Box-folds are of particular interest because the front, back and top patches (F_L, F_R, B_L, B_R, T_G, T_B) can be composed of multiple layers of patches, which can better approximate the volume of the original mesh. For example, the box-folds used in the car and monster truck in Figure 15 are composed of several front, back and top patches at different depths or heights. These patches are obtained by performing image segmentation on the depth and normal maps taken from an orthographic projection of the front, back and top view.

Basically, the image segmentation works by locally fitting a quadratic surface on the segmented pixels in the neighborhood of a candidate pixel, similar to [LLLN*13]. It determines whether a pixel p should be part of the current segment by thresholding $f(p)-q(p)$, where $f(p)$ is the depth value and the x-, y-, z-components of the normal vector at p, and $q(p)$ is the quadratic approximation from the previously segmented pixel p_0:

$$q(p) = f(p_0) + f'(p_0)(p-p_0) + \frac{1}{2}f''(p_0)(p-p_0)^2 \quad (3)$$

4.4. Design Layout Generation

After we obtain the patches, we continue to generate the design layout for cutting, folding and gluing. We position the patches separately on a sheet of paper, adding a flap to each edge where it connects with another patch. When two adjacent patches are bounded by cut lines and their common hinge, we connect them in the layout so that the user will only need to create a fold without gluing. We finally assign IDs to the patches and flaps accordingly.

The order of assembling the patches is important. Some orders are not feasible, while some are easier to construct in practice. In our work, we employ a method similar to [APH*03], which produces assembly instructions for rigid components. While we achieve feasible assembly orders for our pop-ups, further study is needed to improve the ease of assembly.

5. Results

We run our algorithm on an Intel Core i7 PC with 8GB of RAM. The entire process completes in a few seconds to a few minutes with the primitive fitting accounting for most of the running time. We have tested our approach on several 3D models from the Google 3D Warehouse [Goo13], where some models contain up to 30000 vertices. Figure 15 shows some of the models we have used.

Our approach works well for block-like models, such as man-made objects (e.g. cars, trucks, buildings), since they can be closely approximated using our set of primitives. Although our primitives are symmetric, the combinations of these primitives can be asymmetric structures. Organic and rounded models might require more complex primitives and higher levels of abstraction (see Figure 17). Objects that are hard to align to the principal axes may also be difficult to approximate exactly. In some cases, like the T-shape, necessary additional supports may also distract from the original shape.

We compare our results with those of [LSH*10] and [LJGH11] in Figure 16. Because of the voxelization, their results contain many small box structures, which are difficult to make in practice. Our pop-ups consist of only two box-folds for the Waldorf-Astoria Hotel and a single tent-fold for the Eiffel Tower. Although our geometry is not as detailed, we are able to use textures to capture the details and shape information.

Most of our pop-up designs employ only a few mechanisms and as such only a few patches. This is intentional since our main goal is to generate the simplest design that is still able to capture the general shape of the input 3D model. We have determined that in most cases capturing the side contours of a 3D model is enough to give us a good abstraction of the model.

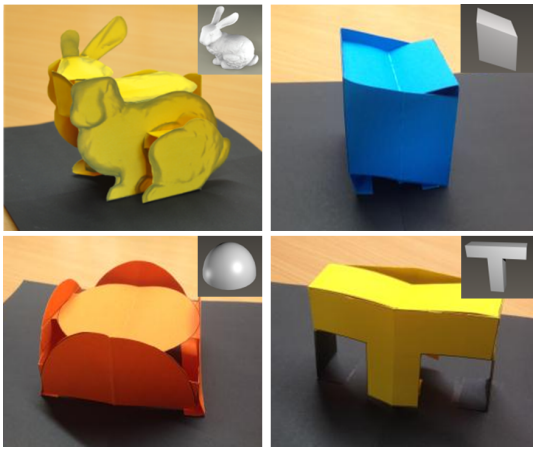

Figure 17: *Our actual paper pop-ups for Stanford Bunny (textured using the rendered model), skewed cube, half-sphere and T-shape.*

Figure 15: *Sample results. Input models and their corresponding actual pop-ups.*

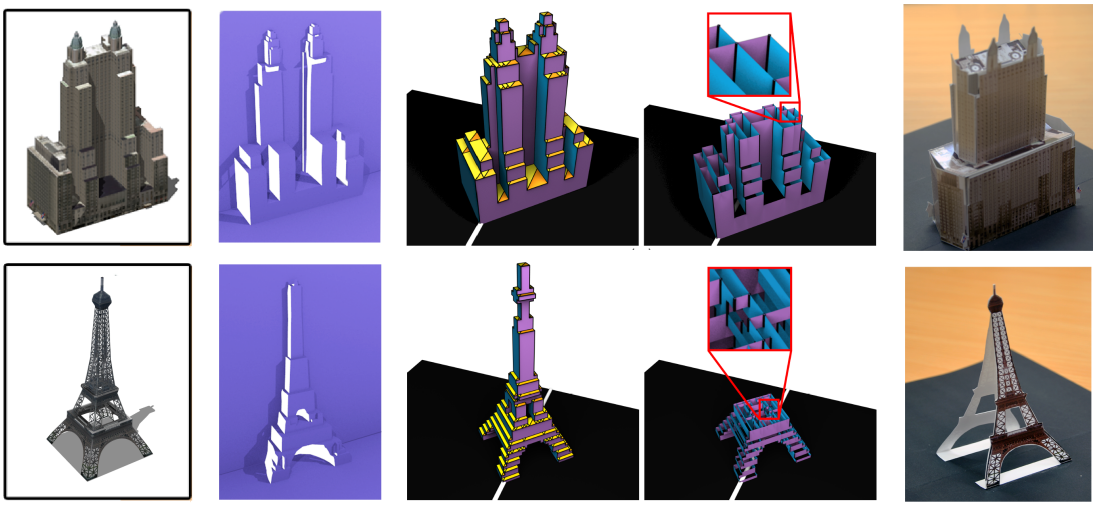

Figure 16: *Waldorf-Astoria Hotel and Eiffel Tower models. (a) Input 3D model, (b) [LSH*10] results, (c) [LJGH11] results (from paper) and (d) our actual paper pop-ups.*

6. Conclusions and Future Work

We present an original algorithm to generate multi-style pop-up designs automatically. Some of the individual steps (e.g. primitive fitting) are well-known graphics techniques, but their combination and their use in similar context, with explicit consideration of pop-up validity, have not been done in previous work. We believe our approach provides a novel and reasonable framework for combining multiple pop-up mechanisms.

Our approach has difficulty in approximating curved and rounded parts, since we only have rectangular and triangular prisms as primitives. Professional artist are able to approximate these shapes in their work using very specialized rounded or curved mechanisms. These specific mechanisms however are not included because their formulations would require a different definition of stability, under the assumption of non-rigid and bendable paper.

Our system is currently fully automatic. However, it may be important for the user to be given creative control over the final design. We may achieve that in the future by incorporating our approach into other interactive systems, such as those by [Gla02a, Gla02b, HE06]. In addition, the presented geometric formulations here do not take into account the physical characteristics of paper. In actual pop-up design, the thickness, mass, strength and elasticity of paper are important considerations. Lastly, we do not have quantitative assessment of the aesthetic quality of our pop-ups. Such measurements can be very beneficial in creating more visually appealing paper pop-up designs.

Acknowledgements

This work is supported by the Singapore MOE Academic Research Fund (Project No. T1-251RES1104). We would also like to thank Armandarius Darmadji for his help in the implementation.

References

[ADD*12] ABEL Z., DEMAINE E. D., DEMAINE M. L., EISENSTAT S., LUBIW A., SCHULZ A., SOUVAINE D. L., VIGLIETTA G., WINSLOW A.: Universality results for pop-up cards. Preprint, 2012. 3

[APH*03] AGRAWALA M., PHAN D., HEISER J., HAYMAKER J., KLINGNER J., HANRAHAN P., TVERSKY B.: Designing effective step-by-step assembly instructions. *ACM Trans. Graph. 22*, 3 (July 2003), 828–837. 8

[BH02] BELCASTRO S.-M., HULL T. C.: Modelling the folding of paper into three dimensions using affine transformations. *Linear Algebra and its Applications 348* (2002), 273–282. 4

[Car08] CARTER D. A.: *Yellow Square: A Pop-Up Book for Children of All Ages*. Little Simon, 2008. 1, 2

[Cro11] CROWTHER R.: *Amazing Pop-Up Monster Trucks*. Walker & Company, 2011. 1, 2, 4

[CS03] CARROLL L., SABUDA R.: *Alice's Adventures in Wonderland: A Pop-up Adaptation*. Little Simon, 2003. 1, 2

[DDSD03] DÉCORET X., DURAND F., SILLION F. X., DORSEY J.: Billboard clouds for extreme model simplification. In *ACM SIGGRAPH 2003* (2003), pp. 689–696. 2

[DO07] DEMAINE E., O'ROURKE J.: *Geometric Folding Algorithms: Linkages, Origami, Polyhedra*. Cambridge Uni. Press, 2007. 2

[Gla02a] GLASSNER A.: Interactive pop-up card design, part 1. In *IEEE Comput. Graph. Appl.* (2002), vol. 22, pp. 79–86. 3, 10

[Gla02b] GLASSNER A.: Interactive pop-up card design, part 2. In *IEEE Comput. Graph. Appl.* (2002), vol. 22, pp. 74–85. 3, 10

[Goo13] GOOGLE:. http://sketchup.google.com/3dwarehouse, 2013. 8

[HE06] HENDRIX S. L., EISENBERG M. A.: Computer-assisted pop-up design for children: computationally enriched paper engineering. *Adv. Technol. Learn. 3*, 2 (2006), 119–127. 3, 10

[HEH05] HOIEM D., EFROS A. A., HEBERT M.: Automatic photo pop-up. In *ACM SIGGRAPH 2005* (2005), pp. 577–584. 3

[HEH08] HOIEM D., EFROS A. A., HEBERT M.: Closing the loop in scene interpretation. In *IEEE CVPR* (2008). 3

[HS09] HARA T., SUGIHARA K.: Computer-aided design of pop-up books with two- dimensional v-fold structures. In *Proc. 7th Japan Conf. on Comput. Geometry and Graphs* (2009). 4, 5

[Hul06] HULL T.: *Project Origami: Activities for Exploring Mathematics*. A K Peters, Ltd, 2006. 2

[IEM*11] IIZUKA S., ENDO Y., MITANI J., KANAMORI Y., FUKUI Y.: An interactive design system for pop-up cards with a physical simulation. *Vis. Comput. 27*, 6 (June 2011), 605–612. 3

[LJGH11] LI X.-Y., JU T., GU Y., HU S.-M.: A geometric study of v-style pop-ups: theories and algorithms. In *ACM SIGGRAPH 2011* (2011), pp. 98:1–98:10. 2, 3, 4, 5, 7, 8, 9

[LLLN*13] LE S. N., LEOW S.-J., LE-NGUYEN T.-V., RUIZ C., LOW K.-L.: Surface- and contour-preserving origamic architecture paper pop-ups. *IEEE Trans Vis Comput Graph (TVCG)* (Aug 2013). DOI:10.1109/TVCG.2013.108. 2, 3, 8

[LNLRL13] LE-NGUYEN T.-V., LOW K.-L., RUIZ C., LE. S. N.: Automatic paper sliceform design from 3d solid models. *Trans Vis Comput Graph* (Nov 2013). 2

[LSH*10] LI X.-Y., SHEN C.-H., HUANG S.-S., JU T., HU S.-M.: Popup: automatic paper architectures from 3d models. In *ACM SIGGRAPH 2010* (2010), pp. 1–9. 2, 3, 4, 8, 9

[LTS96] LEE Y., TOR S., SOO E.: Mathematical modelling and simulation of pop-up books. *Comp. & Graph. 20*, 1 (1996), 21–31. 3

[MS04] MITANI J., SUZUKI H.: Making papercraft toys from meshes using strip-based approximate unfolding. *ACM Trans. Graph. 23*, 3 (2004), 259–263. 2

[MSM11] MCCRAE J., SINGH K., MITRA N. J.: Slices: a shape-proxy based on planar sections. *ACM Trans. Graph. 30*, 6 (2011), 168:1–168:12. 2

[MSU03] MITANI J., SUZUKI H., UNO H.: Computer aided design for origamic architecture models with voxel data structure. *Trans. of Inf. Process. Soc. of Japan 44*, 5 (2003), 1372–1379. 3

[MWZ*13] MITRA N., WAND M., ZHANG H., COHEN-OR D., BOKELOH M.: Structure-aware shape processing. In *Eurographics State-of-the-art Report (STAR)* (2013). 3

[MZL*09] MEHRA R., ZHOU Q., LONG J., SHEFFER A., GOOCH A., MITRA N. J.: Abstraction of man-made shapes. In *ACM SIGGRAPH Asia 2009 papers* (2009), pp. 137:1–137:10. 2

[O'R11] O'ROURKE J.: *How to Fold It: The Mathematics of Linkages, Origami & Polyhedra*. Cambridge Uni. Press, 2011. 2

[SLCH11] SHIH J.-L., LEE C.-H., CHAO-HUNG C.: A 3d model retrieval approach based on the combination of pca plane projections. *J. Info. Tech. and Appl. 5*, 2 (2011). 6

[SWK07] SCHNABEL R., WAHL R., KLEIN R.: Efficient ransac for point-cloud shape detection. *Comput Graph Forum 26*, 2 (June 2007), 214–226. 6

[Tac10] TACHI T.: Origamizing polyhedral surfaces. *IEEE Trans. Vis. Comput. Graphics 16*, 2 (2010), 298–311. 2

[UT06] UEHARA R., TERAMOTO S.: The complexity of a pop-up book. In *18th Canadian Conf. on Comput. Geom.* (2006). 3

[vKXZ*13] VAN KAICK O., XU K., ZHANG H., WANG Y., SUN S., SHAMIR A., COHEN-OR D.: Co-hierarchical analysis of shape structures. *ACM Transactions on Graphics (Special Issue of SIGGRAPH) 32*, 4 (2013). 3

[YK12] YUMER M. E., KARA L. B.: Co-abstraction of shape collections. *ACM Trans. Graph. 31*, 6 (Nov. 2012), 166:1–166:11. 2, 6