Generating Multi-style Paper Pop-up Designs using 3D Primitive Fitting

Conrado R. Ruiz, Jr. Sang N. Le Kok-Lim Low*
School of Computing, National University of Singapore

Abstract

Paper pop-ups are fascinating three-dimensional books that impart stories and information more vividly to their readers. The design and construction of these pop-up books, however, are generally done by hand, and given the lack of expertise in this area, have necessitated the need for automated or computer-assisted design of paper pop-ups. This paper presents an automatic algorithm and the underlying theory for producing paper pop-up designs from 3D models. Existing studies on paper pop-ups have mainly focused on individual styles and proposed formulations for only a limited set of mechanisms. In our work, we design multi-style pop-ups by combining the formulations of previously studied styles with our derived validity conditions for box-style pop-ups. We use a mesh abstraction technique that fits volumetric primitives unto a 3D mesh, which are later automatically mapped to selected mechanisms. We also preserve important texture and shape contours using a hybrid object- and image-space approach. Finally, we generate printable design layouts and the corresponding assembly instructions to facilitate the actual production, which verifies the feasibility of our pop-ups.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Geometric algorithms, languages, and systems

Keywords: paper pop-ups, physical models, computer art, mesh simplification, shape abstraction

1 Introduction

Paper pop-ups and movable books are three dimensional books containing paper pieces that rise up or move when the book is opened and fold completely flat when the book is closed. Creating a paper pop-up can be a tedious task even for an experienced designer. It usually entails a trial-and-error approach, which may take weeks or months to find a valid configuration of paper pieces that resemble a desired object. Computer-aided tools are now increasingly used in paper pop-up design, after achieving much success in industrial and architectural design.

Most of the works in computational pop-up design focused on developing interactive design tools for individual pop-up styles. They were meant to replace the actual paper cutting, gluing and folding by providing the user with virtual design environments. However, in practice artists may incorporate multiple styles of pop-ups to better abstract a desired scene, although such a process requires considerably more experience and skill. In our approach, we generate multi-style pop-ups automatically by selecting suitable mechanisms that match the 3D primitives extracted from the input model. Our system requires minimal skill from the user as it automatically generates the final pop-up design layout from an input model that can be easily obtained from online 3D repositories.

*email: {conrado,lnsang,lowkl}@comp.nus.edu.sg

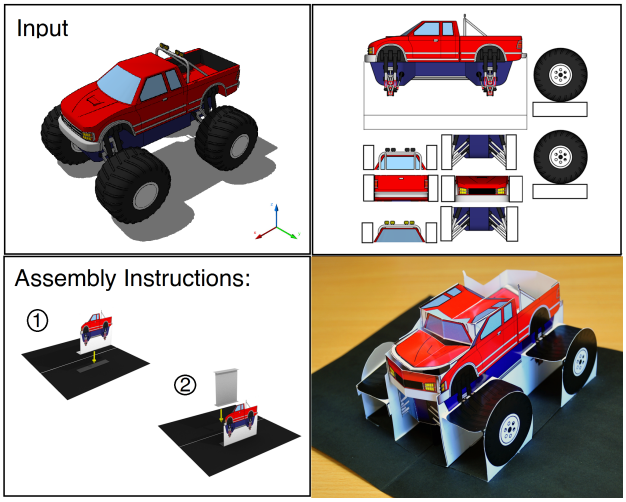

Figure 1: *An input 3D model, its pop-up design layout, assembly instructions and the physical paper pop-up.*

Currently, other fully automated approaches [Li et al. 2010; Li et al. 2011] use voxelization to approximate the input 3D model, which leads to the possible loss of important features. In order to capture accurate contours, a very dense voxel grid is required, which then makes the design infeasible to create in practice because of the excessive amount of cuts and folds. In our work, the mesh abstraction allows us to fit a minimal number of 3D primitives to approximate the model, resulting in fewer patches. We also apply an image-domain approach to preserve a set of important contours and retain the textures of the 3D model.

In addition, we study the geometric properties of valid paper pop-ups, specifically the conditions for their foldability and stability. These conditions serve as the foundation of our algorithm and guarantee the validity of our designs. Finally, besides generating the design layout, our system provides step-by-step instructions for assembling the physical pop-up.

2 Related Work

A number of studies on paper pop-ups have been presented. [Lee et al. 1996; Glassner 2002] described the use of simple geometry to model various pop-up styles. [Hendrix and Eisenberg 2006] developed an interactive design system that simulate the opening and closing of paper pop-ups. Work on the validity conditions of a general paper pop-up is still scarce. This is primarily because determining the foldability of a general pop-up is NP-hard, as shown in [Uehara and Teramoto 2006].

A few studies focused particularly on *Origamic Architectures* (OA), a special type of paper pop-up that is made of a single sheet of paper [Mitani et al. 2003; Li et al. 2010; Le et al. 2013]. [Li et al. 2011] 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 [Li et al. 2010], they used voxelization technique to approximate the input model, which may cause important features to be lost. In addition, their v-style pop-ups were limited to only four possible patch orientations. In our work, we allow a variety of pop-up styles and orientations by using an automatic fitting and selection approach that maps a 3D primitive to a suitable pop-up mechanism. We also propose an image-domain method to preserve important contours and textures that may otherwise be lost in other simplification techniques.

Automatic generation of paper pop-up designs from 3D models can be considered a form of model simplification. [Le-Nguyen et al. 2013] used 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 [McCrae et al. 2011] and 3D primitives [Yumer and Kara 2012]. Our approximation aims to preserve the input contours, as they have been shown to capture important shape information of the objects [Mehra et al. 2009]. 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.

3 Pop-up Basics

A pop-up structure is composed of planar pieces of paper that can be folded and glued together. We formally call this structure a *scaffold*. It comprises a set of planar polygons, called *patches* that are connected at straight line segments called *hinges*. Pop-up structures have 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*.

To facilitate the creation of multi-style pop-ups, 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 mechanisms to make their works come to life. In this paper, we focus on four mechanisms: *v-fold* and *step-fold* (see mechanisms D1 and D2 in [Li et al. 2011]), *tent-fold* (see [Lee et al. 1996]), and *box-fold*, which will be described in this paper. 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 will be used in our algorithm, are shown in Figure 2.

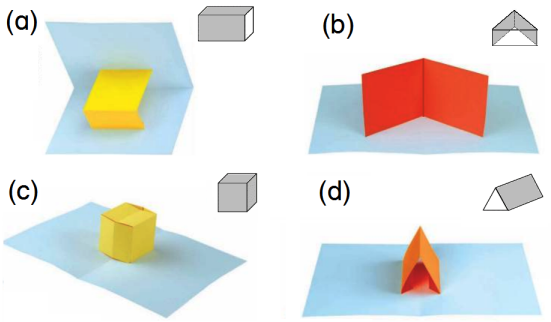

Figure 2: *Pop-up mechanisms and the corresponding 3D primitive: (a) step-fold, (b) v-fold, (c) box-fold and (d) tent-fold structures. The shaded faces are the principal faces.*

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 the rigidity and connectivity of the patches need to be maintained at all times. On the other hand, a pop-up is said to be *stable* if all its patches cannot move when the ground and backdrop patches are held stationary at any fold angle.

The foldability and stability of the step-fold, v-fold and tent-fold mechanisms have been studied in [Li et al. 2011] and [Lee et al. 1996]. In the next section, we describe the box-fold structure and its validity.

3.1 Box-Fold Structure

Consider a pop-up opened at 180°. A box-fold structure is a scaffold with 11 patches labeled as G, B, L, R, F_L, F_R, B_L, B_R, T_G, T_B and C. Patches G and B are 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 being equidistant to the central fold. Patch C is a special backbone glued perpendicularly to G and B at the central fold. On top of the box, patches T_G and T_B are glued to C at their common hinge and connected to L and R, respectively. In our work, all the hinges connecting B, R, T_B, T_G, L and G are made parallel to the central fold.

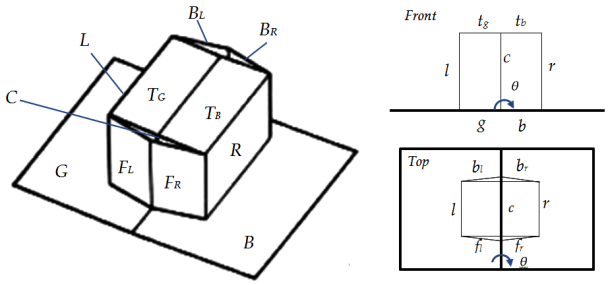

Figure 3: *The box-fold mechanism and its cross sections.*

On the front side of the box, patches F_L and F_R share a hinge and are connected to L and R, respectively. Similarly, B_L and B_R are equivalent patches on the back of the structure. We create the hinges between (F_L, F_R) and (B_L, B_R) so that they are coplanar with patch C. Note that the front and back patches are not glued to C. Theoretically, only one of the two pairs, either (F_L, F_R) or (B_L, B_R), are needed. However, in practice, box-style pop-ups normally contain both sides for better symmetry and sturdiness.

3.2 Validity of Box-Fold

A scaffold is said to be valid if it is *foldable* and *stable*. Here we examine the geometric conditions for a valid box-fold structure.

Foldability. According to [Hara and Sugihara 2009], if a two-dimensional 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 hinges between (F_L, F_R) and (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, B_R can also fold completely flat following the motions of L and R.

Stability. We show that the box-style pop-up as constructed in section 3.1 is readily 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, 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, the box structure is stable.

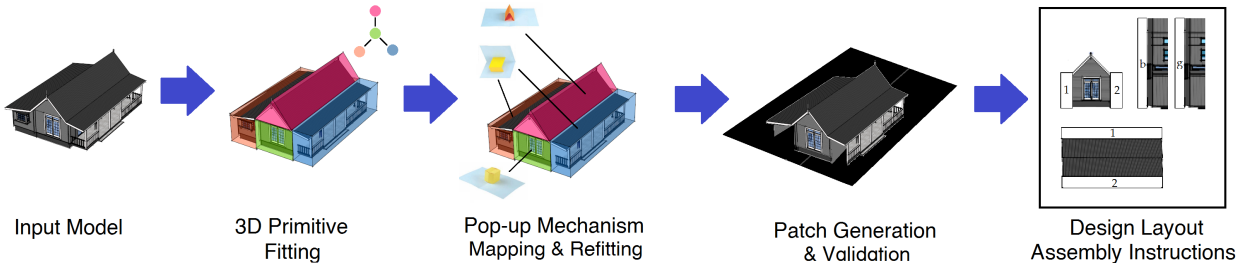

Figure 4: *Overview of the automatic pop-up design algorithm.*

4 Automatic Pop-up Design Algorithm

We employ a divide-and-conquer approach to convert a 3D model into a paper pop-up. Our main idea is to subdivide a 3D model into sub-volumes, and then we handle each part independently with a subsequent step that combines them to form one valid pop-up scaffold. After the patches have been generated and validated, we then produce the printable design layout and decide the assembly order for the instructions.

4.1 3D Primitive Fitting

Initially, we align the input model by using Normal Principal Component Analysis (NPCA) and use the symmetry plane to position the mesh on the primary patches. Then we do a RANSAC primitive fitting similar to the techniques used for point clouds [Schnabel et al. 2007]. However, in our case we fit volumes rather than surfaces like in the work of [Yumer and Kara 2012].

In addition, our primitives also have some constraints on their orientation and connectivity. For example, the primitives must be aligned along the principal direction and they must be connected with other primitives. Furthermore, certain connected pairs are not permitted. A triangular prism is permitted on top of a rectangular box, but not at its side. This is due to the fact that there is no possible mapping of primitives that would allow the final structure to become valid. Consider a box-fold on the central hinge and a v-fold connected to its side. The v-fold will not be foldable since its main fold line will not coincide with another fold line.

Because of these restrictions on the combinations of primitives that are adjacent to each other, we must search though a constraint space of valid combinations of primitives to minimizes the difference of volume to the original mesh using dynamic programming. Once we have the best valid configuration, we store the adjacency information in a graph.

4.2 Mechanism Mapping and Refitting

After the 3D primitive fitting step, we now have a 3D model partitioned into sub-volumes. We now determine how to convert each sub-part into a valid pop-up structure using the valid primitive-mechanism pairs shown in Figure 2. 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 configuration 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 sub-volume. Unlike [Li et al. 2011] 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 2).

In some cases, the primitives can be refitted based on the geometric conditions for the foldability of the mechanism, e.g. box-fold side patches do not necessarily need to be orthogonal to the ground or backdrop patches. The box-fold side and top patches could be further refitted using least-squares approximations, as long as the lengths of the patches still satisfy $b + r = c + t_b$, $g + l = c + t_g$. Such as the case of the ship in Figure 5, where the side of the box-fold is refitted to slant along the body of the ship. Note that the refitting can affect adjacent primitives and as such is not always allowed.

4.3 Patch Generation and Validation

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. Each primitive-mechanism pair will have its own rules, but in general we get 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 they can be composed of multiple layers of patches that can better approximate the volume of the original mesh. In order to generate these patches we use an image segmentation based on the depth map and normal map of the orthographic projections of the top, front and back view. Basically, the surface segmentation works by locally fitting quadratic surfaces on the pixels.

Next we have to verify that the scaffold is valid. We check each primitive locally and the interconnections between primitives using our geometric conditions for foldability and stability. Locally, it may be possible that some patches are floating because they are not touching any adjacent patches. In order to handle these patches, we could merge it with a patch that is already stable or create support patches. For example in Figure 5, the airplane model has additional supports on its tail fins for stability. Globally, the final positions of the patches can cause disconnections or intersections with other mechanisms that we need to fix. We also check if the neighboring patches can be merged into one patch based on the difference of their normals and locations.

4.4 Design Layout Generation

Once we have obtained the patches, we continue with the design layout generation. We position the patches separately on a sheet of paper, adding flaps to the edges where it will connect with another patch. In some cases we can do some optimization to pre-connect the patches, similar to unwrapping a 3D mesh. We then label all the patches numerically and tag the flaps accordingly.

The order of assembly of the patches is important. There are some assembly orders that are not feasible and there are certain orders that are easier to construct in practice. As such, the generation of the step-by-step instructions is not as trivial. We employ an approach similar to the method proposed by [Agrawala et al. 2003] for producing assembly instructions given the geometry and orientation of its parts. Eventually we want to adapt their approach to be more suited for our case.

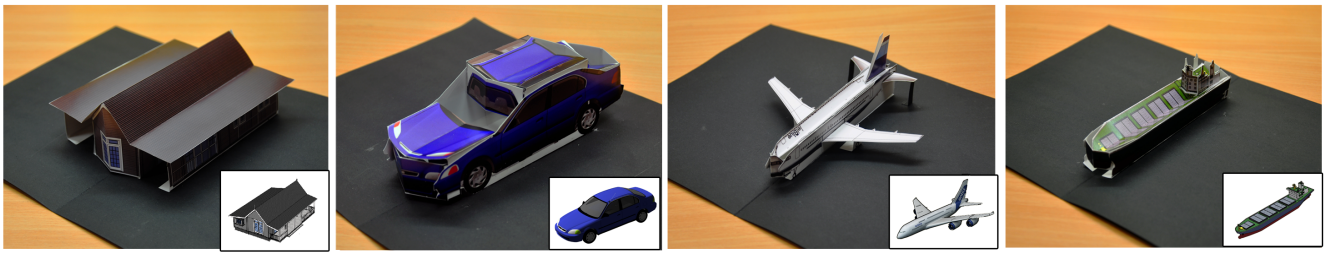

Figure 5: *Sample results. Input models and their corresponding actual pop-ups.*

5 Results

We have tested our approach on several 3D models from the Google 3D Warehouse [Google 2013]. Figure 5 shows some of our preliminary results. Our approach works well for block-like models, such as man-made objects (i.e. cars, trucks, buildings) since they can be closely approximated using our set of primitives. The primitives are symmetric although the combination of these primitives may be asymmetric. More organic models might require more complicated primitives and higher levels of abstraction. We compare our results with [Li et al. 2011] in Figure 6. Because of voxelization, their results contain many tiny box structures in the detailed parts of the model and may be difficult to make in practice. Our pop-up consists only of two box-folds. Although the geometry is not as detailed, the texture is able to capture some of the lost shape information.

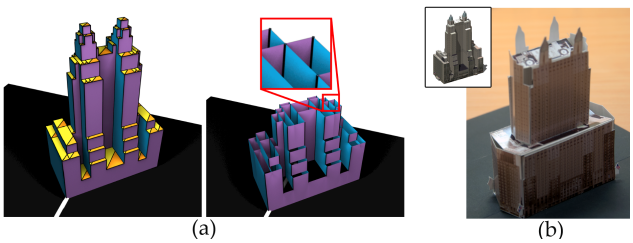

Figure 6: *Waldorf-Astoria Hotel Model. (a) [Li et al. 2011] results. (b) Our actual paper pop-up.*

6 Conclusions and Future Work

This paper presents a method for automatic pop-up design generation using 3D mesh abstraction and an image-domain approach grounded on a set of geometric formulations. We have shown that these designs are feasible in practice by manually constructing the paper pop-ups from the generated output designs. Although we only used four mechanisms, it is possible to use other mechanisms, as long as each of these mechanisms has a set of conditions for foldabilty and stability, a corresponding 3D primitive pair and an algorithm to convert the primitive into a set of pop-up patches. Eventually, we wish to generalize this approach in one unified framework that encompasses most of the known pop-up mechanisms to date.

The system is currently fully automatic. In the future, it is possible to incorporate the approach with other interactive systems, such as those by [Glassner 2002; Hendrix and Eisenberg 2006]. In addition, the geometric formulations presented here do not take into account the actual physical characteristics of paper. In actual pop-up design, the thickness, mass, strength and elasticity of paper are important considerations.

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

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

GLASSNER, A. 2002. Interactive pop-up card design, part 1 and 2. In *Comput. Graph. and Appl.*, vol. 22, 79–86(no.1), 74–85(no.2).

GOOGLE, 2013. http://sketchup.google.com/3dwarehouse.

HARA, T., AND SUGIHARA, K. 2009. Computer-aided design of pop-up books with two- dimensional v-fold structures. In *Proc. 7th Japan Conf. on Computational Geometry and Graphs*.

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

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

LE-NGUYEN, T.-V., LOW, K.-L., RUIZ, C., AND LE., S. N. 2013. Automatic paper sliceform design from 3d solid models. *Trans Vis Comput Graph* (May). DOI:10.1109/TVCG.2013.82.

LEE, Y., TOR, S., AND SOO, E. 1996. Mathematical modelling and simulation of pop-up books. *Comp. & Graph. 20*, 1, 21–31.

LI, X.-Y., SHEN, C.-H., HUANG, S.-S., JU, T., AND HU, S.-M. 2010. Popup: automatic paper architectures from 3d models. In *ACM SIGGRAPH 2010 papers*, SIGGRAPH '10, 1–9.

LI, X.-Y., JU, T., GU, Y., AND HU, S.-M. 2011. A geometric study of v-style pop-ups: theories and algorithms. In *ACM SIGGRAPH 2011 papers*, SIGGRAPH '11, 98:1–98:10.

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

MEHRA, R., ZHOU, Q., LONG, J., SHEFFER, A., GOOCH, A., AND MITRA, N. J. 2009. Abstraction of man-made shapes. In *ACM SIGGRAPH Asia 2009 papers*, 137:1–137:10.

MITANI, J., SUZUKI, H., AND UNO, H. 2003. Computer aided design for origamic architecture models with voxel data structure. *Trans. of Inf. Process. Soc. of Japan 44*, 5, 1372–1379.

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

UEHARA, R., AND TERAMOTO, S. 2006. The complexity of a pop-up book. In *18th Canadian Conf. on Comput. Geom.*

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