An Interface for Assisting the Design and Production of Pop-Up Card

Sosuke Okamura[1], Takeo Igarashi[2]

[1] Department of Computer Science, The University of Tokyo,
Bunkyo-ku, Tokyo, Japan
okamura@ui.is.s.u-tokyo.ac.jp
[2] Department of Computer Science, The University of Tokyo / ERATO, JST
takeo@acm.org

Abstract. This paper describes an interface for assisting the design and production of pop-up cards by using a computer. A pop-up card is a piece of a folded paper from which a three-dimensional structure pops up when it is opened; it can be folded flat again afterward. Many people enjoy this interesting mechanism in pop-up books and greeting cards. However, nonprofessionals find it difficult to design pop-up cards because of the various geometric constraints required to make the card fold flat. We therefore propose an assistant interface to help people easily design and construct pop-up cards. In this paper, we deal with pop-up cards that open fully to 180°.

We have designed a prototype that allows the user to design a pop-up card by setting new parts on the fold lines and editing their position and shape afterward. At the same time, the system examines whether the parts protrude from the card or whether the parts collide with one another when the card is closed. Users can concentrate on the design activity because the results are continuously fed back to them.

We created several pop-up cards using our system and performed an informal preliminary user study to demonstrate its usability.

1 Introduction

A pop-up card is a piece of a folded paper from which a three-dimensional (3D) paper structure pops up when it is opened. The card can be folded flat again afterward. Many people enjoy this interesting mechanism in pop-up books [1] [2] [3] and greeting cards, and receiving and viewing pop-up cards appeals to people of all ages. Fig. 1 shows an example of pop-up book.

Constructing a pop-up card is relatively easy; anyone can simply cut out the pieces and glue them together if a template is available. Unfortunately, it is much more difficult for nonprofessionals to design a pop-up card from scratch. The first problem is correctly understanding the pop-up card mechanism. The second problem is determining the positions of objects so that pop-up parts do not collide. This usually requires repetitive trial and error during design: cutting out component parts out of paper, pasting them on the card, and checking

Fig. 1. "Alice's Adventures in Wonderland", a typical pop-up book [1].

whether they collide. If an error is found, re-thinking the design and starting over from the beginning. This process requires a lot of time, energy, and paper. Design and simulation in a computer eliminate the boring repetition and save time.

Glassner proposed methods for designing pop-up cards [4] [5] [6]. He introduced several simple pop-up mechanisms and described how to use these mechanisms, how to simulate the position of vertices as an intersecting point of three spheres, how to check whether the structure sticks out beyond the cover or if a collision occurs during opening, and how to generate templates. His work is quite useful in designing simple pop-up cards.

Our work builds on Glassner's pioneering work and introduces several innovative aspects. Our system has two new mechanisms based on the V-fold: the box and the cube. We provide a detailed description of the user interface, which Glassner did not describe in any detail. In addition, our system provides real-time feedback to the user during editing operations by examining whether parts protrude from the card when closed or whether they collide with one another during opening and closing. Finally, we report on an informal preliminary user study of our system involving two inexperienced users.

2 Related Work

Glassner described a stable analytical solution for the location of important points in the three main pop-up techniques (single-slit, asymmetric single-slit, and V-fold mechanisms) when the pop-up card folds and unfolds in interactive pop-up card design [4] [5] [6]. He also implemented his solution in a small design program. However, he did not describe the interactive behavior of the program in detail, nor did he report any user experience.

Mitani and Suzuki proposed a method for creating a 180° flat fold Origamic Architecture with lattice-type cross sections [7]. This system creates pieces from a 3D model. Those pieces can be used in our system because the base mechanism is same as the angle fold open box mechanism described in the following section.

However, a 3D model is not always available. Furthermore, the structure is set only on the center fold line and cannot be combined with other pieces.

Lee et al. described calculations and geometric constraints for modeling and simulating multiple V-fold pieces [8]. However, that system is limited to V-fold mechanisms and is not designed as an interactive system.

Several interactive interfaces have been proposed for 90° pop-up cards. Mitani et al. proposed a method to design Origamic Architecture [9] models with a computer using a voxel model representation [10] [11]. Using this method, the system can store 90° pop-up card models and display them using computer graphics. User operations are the interactive addition and deletion of voxels. This system was used for graphics science education [12]. Mitani et al. also proposed a method for designing Origamic Architecture models with a computer using a model based on a set of planar polygons [13]. That system computes and imposes constraints to guarantee that the model can be constructed with a single sheet of paper. Thus, it enables the user to make more complex 90° pop-up cards interactively from the beginning through to the pattern printing stage. Hendrix and Eisenberg proposed a pop-up workshop system [14] [15] that enables the user to design pop-up cards by making two-dimensional (2D) cuts. The result in 3D can be opened and closed on the Viewer window. They also showed that children could design pop-up cards using their system [16]. However, their interfaces are not available for 180° pop-up cards. 90° pop-up cards are made with single sheet of paper and their system works well given this constraint. However, it is difficult to design 180° pop-up cards using a voxel model or planar polygon model. Moreover, it is difficult for people to edit 3D structures in 2D because they can not imagine the resulting 3D shape.

3 Assisted Pop-up Card Design

Fig. 2 shows a screenshot of our prototype system.

Our system allows the user to design pop-up cards using a mouse and a keyboard. The system consists of a single window for the design and simulation of the pop-up card. Fig. 3 shows the system functions. The user positions parts, adjusts their properties (e.g., length, angle, position and pattern), and generates templates. During interactive editing, the system always examines whether parts protrude from the card or whether they collide with one another.

The user can use five mechanisms in our system. The V-fold mechanism, the even-tent mechanism, and the uneven-tent mechanism were introduced by Glassner [4]. The angle fold open box mechanism and the angle fold cube mechanism [17] based on the V-fold are new. The former forms a box, which is an empty rectangular parallelepiped without a top and a bottom as shown in Fig. 4 (a). The latter makes an angle fold open box with a lid as shown in Fig. 4 (b). The lid is folded toward the inside of the box when the card is closed.

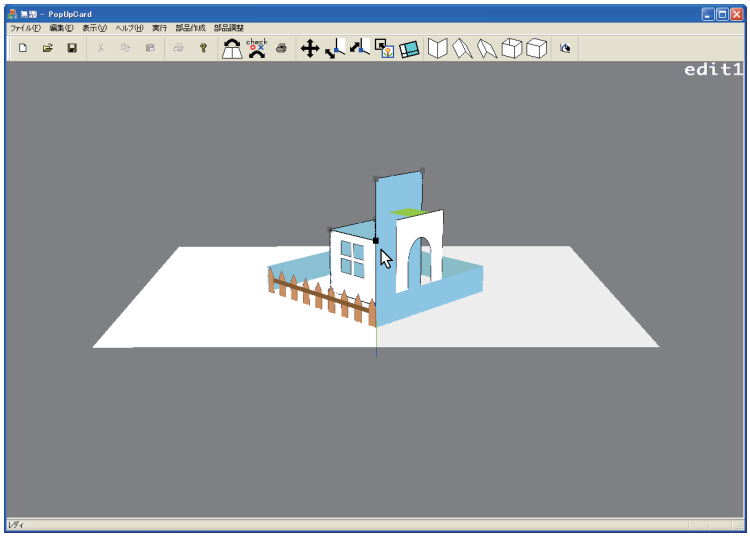

Fig. 2. Prototype system.

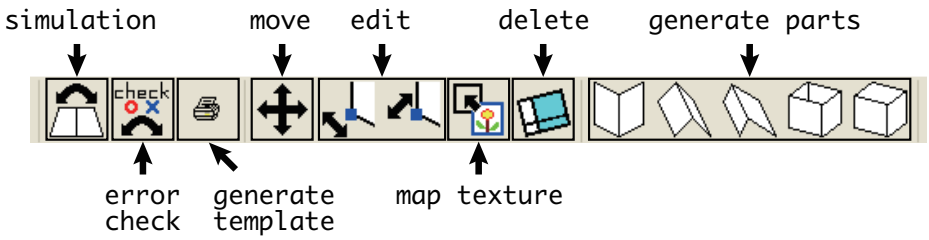

Fig. 3. Prototype system functions.

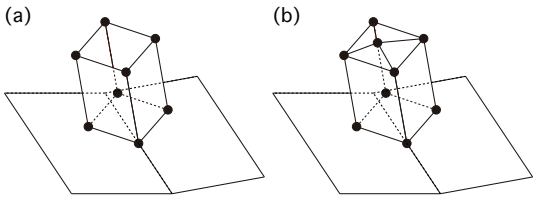

Fig. 4. (a) Angle fold open box mechanism. (b) Angle fold cube mechanism.

3.1 Constrained Editing

Setting Parts The user first selects a desired mechanism and moves the mouse cursor to a fold line where the black point then appears. If the user clicks the mouse button, the system automatically generates the new part at that position with default lengths and angles that can be adjusted later (see Fig. 5).

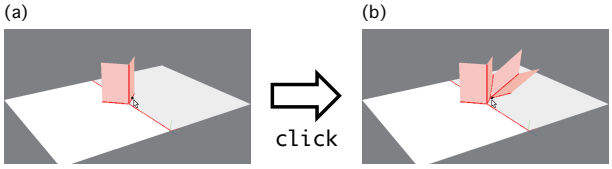

Fig. 5. User interface for setting parts. (a) When the user clicks on a fold line, (b) the system generates a new part there.

Editing Shapes The user drags the vertices of a part to edit its shape as shown in Fig. 6. The vertices of a part are highlighted when the user selects that part. The user can then drag the vertices one at a time. When editing lengths, the user can maintain the parallelogram shape by pressing the Shift key while dragging as shown in Fig. 6 (c). The user can also extend a panel of a part as shown in Figs. 6 (d) and (e) or change the angle between two planes or the inclination of the part as shown in Figs. 6 (f)-(i).

Deleting Parts The user clicks on a part to delete it. If other parts exist on the part being deleted, they are also deleted.

Mapping Textures The user prepares images in advance and can use them as textures. To put a texture on a part, the user selects a panel of a part and an image as shown in Figs. 6 (j) and (k).

Generating Templates The system automatically generates 2D templates from the 3D model designed by the user. The system also creates glue tabs and generates guidelines to tell the user where to glue them on.

3.2 Realtime Error Detection

During interactive editing, the system examines whether the parts protrude from the card when closed or whether they collide with one another during closing and opening. When the system detects such a protrusion or collision, it displays a message on the lower right portion of the window and highlights the parts that caused the error as shown in Figs. 7 (c) and (d). This occurs in real time. The message and the highlights disappear immediately once the error is resolved.

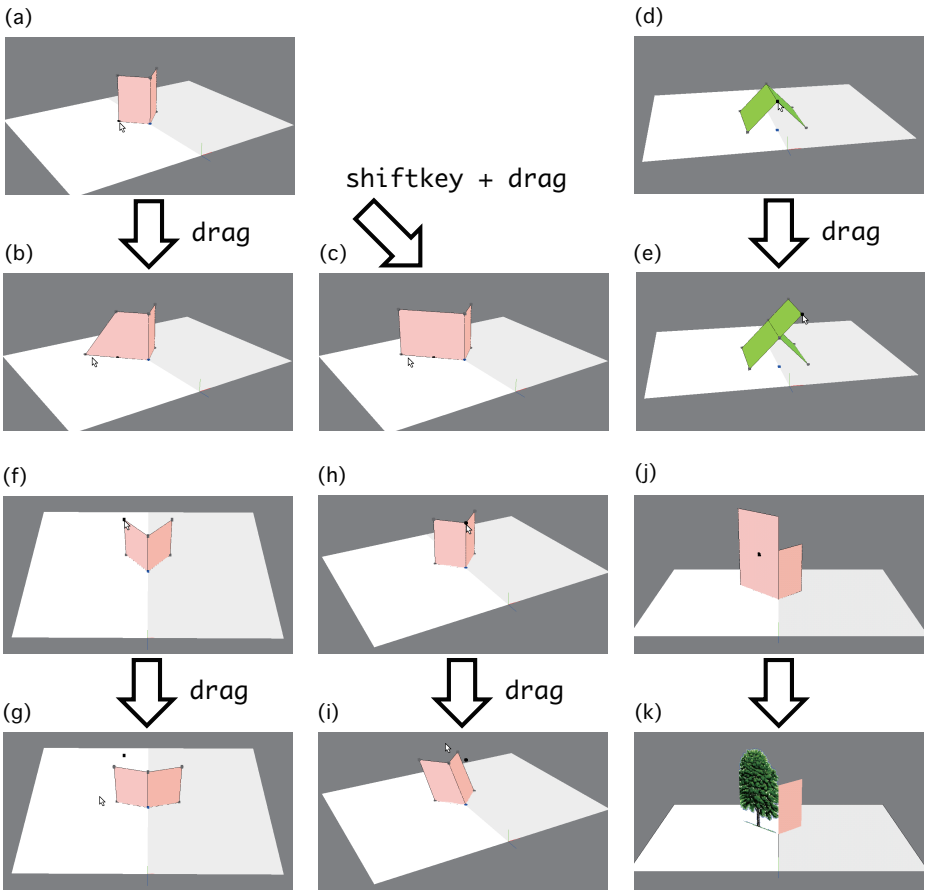

Fig. 6. User interface to edit shapes. The user can (a)-(e) change lengths; (f), (g) change open angles; (h), (i) change gradient angles; and (j), (k) apply textures.

4 Implementation

We implemented our prototype system based on Glassner's work [4] [5] [6] using Microsoft Visual C++ with the MFC platform. We also used OpenGL to render the scene and OpenCV to read images. Fig. 3 shows the functions that were implemented. Note that the *Undo* function has not yet been implemented. A pop-up card is represented as a tree data structure. Each node of the tree corresponds to a part and contains its relative position on the parent part and various parameters. The system first computes the card position based on its open angle and updates its center fold line information. It next updates the 3D coordinates of the parts on the fold line. The system repeats this procedure recursively to determine the 3D coordinates of all parts.

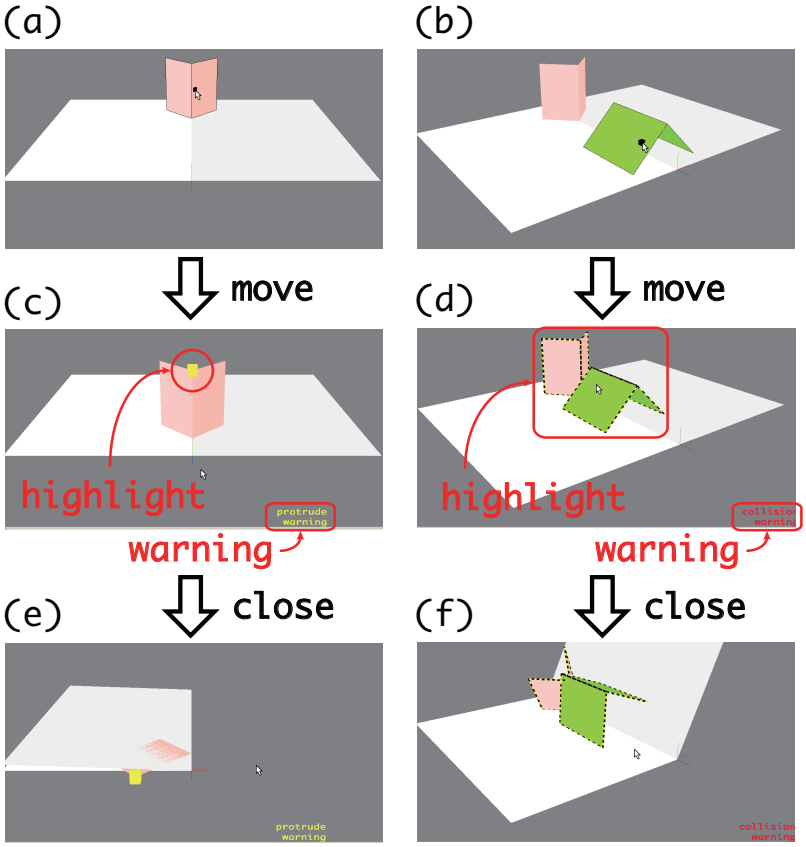

Fig. 7. User interface when the system detects errors. (a), (b) The user moves a part. (c), (d) The system displays a warning message and highlights the part if the system detects an error. (e) The part protrudes from the card if the card is closed. (f) The part collides with one another if the card is closed.

The system checks for protrusion and collision errors while dragging and placing parts. The system checks for collisions at every 10°. This may miss a minor collision, but this is not an issue due to the inherent flexibility of paper. We initially tried checking every 1°, but this was too slow when the number of parts increased.

5 Results

Fig. 8 shows examples of pop-up cards designed using our system. The first was based on "The Wonderful Wizard of OZ" [2] pop-up book shown in Fig. 8 (a). We designed the piece shown in Fig. 8 (b) on our system in approximately

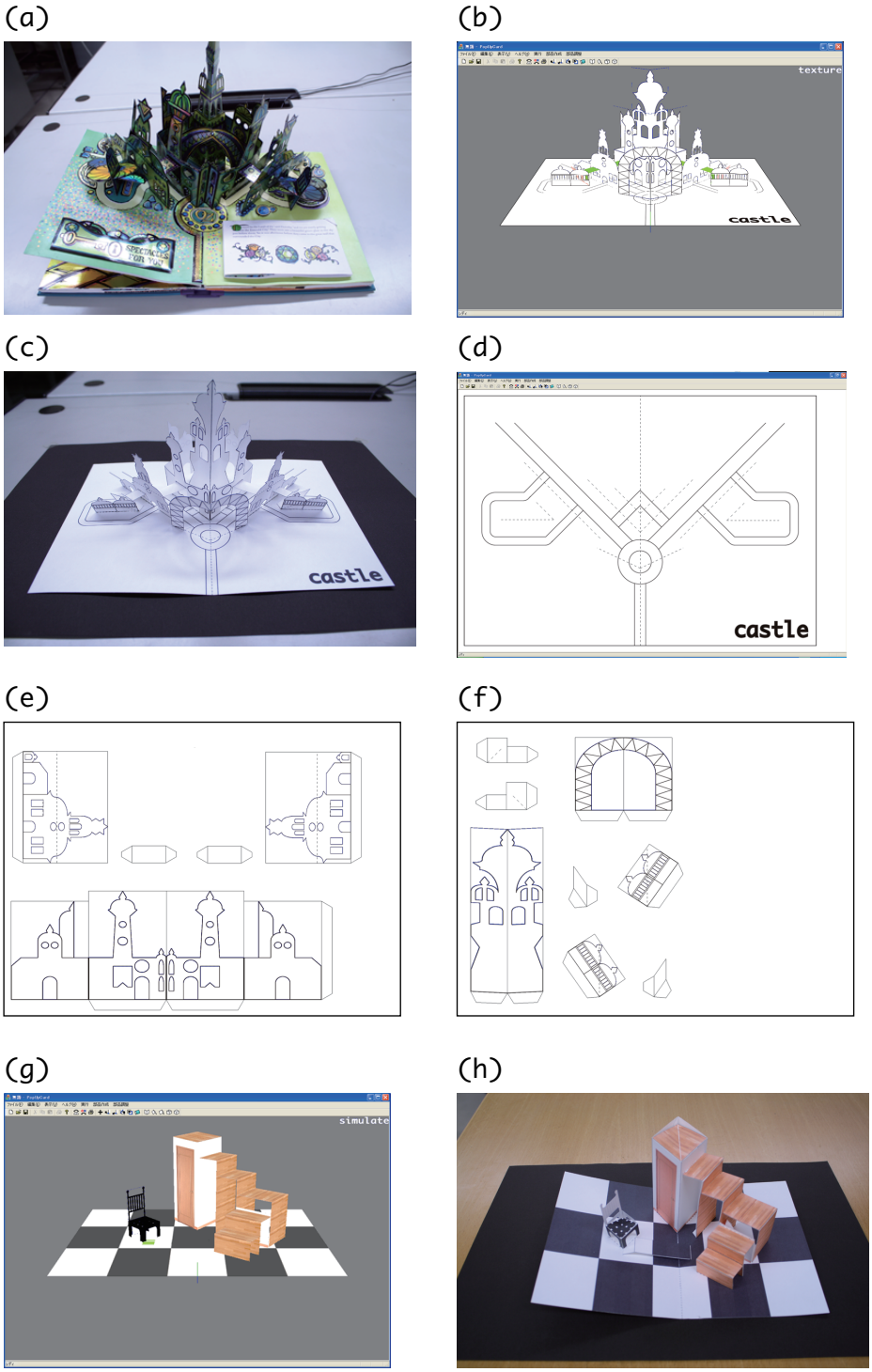

Fig. 8. Pop-up cards designed using our system.

2h. Determining the heights and position was easy due to the error detection mechanism. Fig. 8 (d)-(f) show the templates generated by our system, which required approximately 2h for assembly in the final form shown in Fig. 8 (c). Fig. 8 (g) shows another example. This required approximately 2h for the design and 2.5 h for assembly into the final form shown in Fig. 8 (h). These results show that our system can handle reasonably complicated pop-up structures.

We also conducted an informal preliminary user study. The test was conducted using a typical notebook personal computer with a keyboard and a mouse. We assigned two tasks: create a pop-up card of a building without using our system (Task 1) and perform the same task using our system (Task 2). Two test users participated in the study. They first completed Task 1. We then explained our system to them and had them practice with it for approximately 5 min before attempting Task 2. After they had finished both tasks, we conducted a brief interview to receive their feedback.

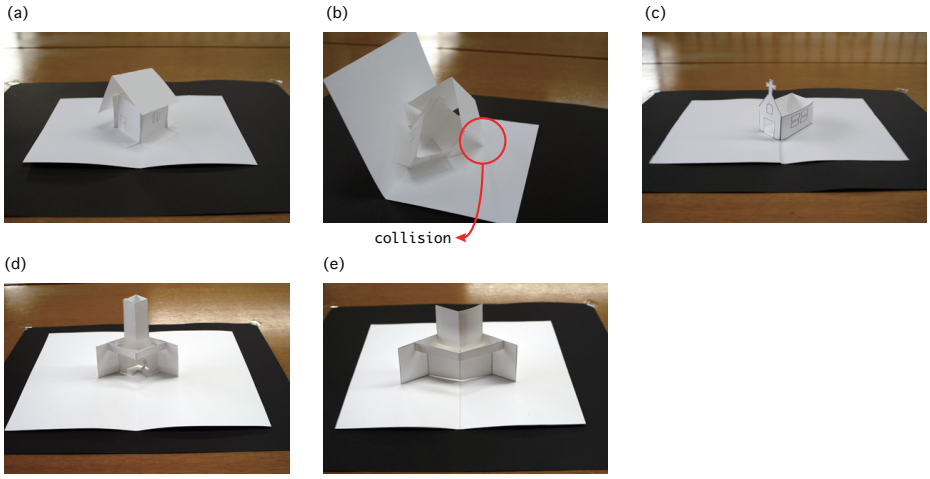

Fig. 9. User study results. (a) The card produced by the first user in Task 1. (b) Collision occurs when the object in (a) is closed. (c) The card produced by the first user in Task 2. (d) The card produced by the second user in Task 1. (e) The card produced by the second user in Task 2.

The first test user had never created pop-up cards before and had no knowledge of them. Fig. 9 (a) shows the card created by the first user in Task 1. It looks good, but it is difficult to close because of the collision shown in Fig. 9 (b). Fig. 9 (c) shows the card created by the first user in Task 2; it is simple but it closes correctly. The first user required 30 min to complete Task 1 (15 min for design and 15 min for construction) but only 20 min to complete Task 2 (10 min each for design and construction).

The second test user had also never created pop-up cards before and had little knowledge of them. The cards created by the second user are shown in Fig. 9 (d) (Task 1) and Fig. 9 (e) (Task 2). They have mostly the same structure and close correctly. However, the second user required 70 min to complete Task 1 (40 min for design and 30 min for construction) but only 40 min to complete Task 2 (20 min each for design and construction).

The preliminary user study and the feedbacks from the users show the following advantages of our system. First, it is easier to design a pop-up card using our system than using paper, scissors, and glue. Second, the pop-up cards designed using our system close correctly without protrusion or collision. This is due to the protrusion and collision detection integral to our system. The second user spent less time in the design stage because it was easy to adjust the height of the pop-up part so that it would not protrude from the card, whereas that user spent a long time adjusting the height in Task 1. The first user failed to make an acceptable pop-up structure manually but succeeded using our system.

Even so, there are several limitations to our system. First, even through a symmetrical design is frequently used in pop-up cards, symmetric editing is not directly supported in our current implementation. Second, because our system uses displacement in two dimensions on the screen for editing, the direction that the user desires to move and the one that the user must move may be different.

6 Conclusion and Future Work

Here we proposed an assistant interface to help people to design and create pop-up cards. Our system examines whether the parts protrude from the card or whether they collide with one another during interactive editing, and it displays the result continuously to the user as feedback. This helps the user concentrate on the design activity. We showed sample pop-up cards created using our system and reported the results of a preliminary user study. The preliminary user study shows that the protrusion and collision detection functions are very effective.

We plan to add new functions to the system in the future. First, we would like to make a mirror editing system in which the system changes values to maintain symmetry if the user edits a part. Second, we would like to create a function that the user could use to change lengths and angles by entering numerical values. Third, we would like to make a constraint mechanism so that when the user marks a pair of edges, those two edges are always the same length. Whereas we use textures to add appearance details to a part, preparing the texture in advance using other software is inconvenient. We would like to let the user paint textures directly on a part using our system. Although we have implemented five mechanisms in this system, there are many other possibilities. One of the most interesting mechanisms we plan to implement is curved surfaces. Curved surfaces deform non-linearly unlike a simple planar surfaces and we plan to apply some soft of physical simulation.

We would not claim that our method (direct manipulation interface with continuous feedback) is the best interface for designing physical objects in gen-

eral. There are many other methods for designing physical objects such as quick sketching or tangible interfaces. Each method has its own strength and weakness. Our experience is that sketching and tangible approaches are less constraining, so they are good for very early exploration. In contrast, our method is suitable for later stages of design process or for the design of objects with complicated constraits. Pop-up card is an example of highly-constrained objects where one can not design arbitrary shape and our approach works well. However, it is true that our method is a little bit too constraining for very initial exploration and we would like to work on this problem in the future.

Acknowledgements

We would like to thank Jun Mitani for his helpful comments. We also appreciate the members of Igarashi Laboratory for their useful discussions.

References

1. R. Sabuda and L. Carroll. *Alice's Adventures in Wonderland*. Little Simon, 2003.
2. L.F. Baum. *The Wonderful Wizard of OZ*. Elibron Classics, 2000.
3. R. Sabuda and M. Reinhart. *Encyclopedia Prehistorica: Dinosaurs*. Candlewick Press, 2005.
4. Andrew Glassner. Interactive pop-up card design. *Miscrosoft Technical Report*, 1998.
5. Andrew Glassner. Interactive pop-up card design, part i. *IEEE Computer Graphics and Applications*, 22(1):79–86, 2002.
6. Andrew Glassner. Interactive pop-up card design, part 2. *IEEE Computer Graphics and Applications*, 22(2):74–85, 2002.
7. Jun Mitani and H. Suzuki. Computer aided design for 180-degree flat fold origamic architecture with lattice-type cross sections. *Journal of graphic science of Japan*, 37(3):3–8, 20030901.
8. YT LEE, SB TOR, and EL SOO. Mathematical modelling and simulation of pop-up books. *Computers graphics*, 20(1):21–31, 1996.
9. M. Chatani. Origamic Archtecture Toranomaki, 1985.
10. Jun Mitani, H. Suzuki, and H. Uno. Computer Aided Design for Origamic Architecture Models with Voxel Data Structure. *Transactions of Information Processing Society of Japan*, 44(5):1372–1379, 2003.
11. Pop-up Card Designer PRO. `http://www.tamasoft.co.jp/craft/popupcard-pro/`.
12. J. Mitani and H. Suzuki. Making use of a cg based pop-up card design system for graphics science education. *Journal of graphic science of Japan*, 38(3):3–8, 20040901.
13. J. Mitani and H. Suzuki. Computer aided design for origamic architecture models with polygonal representation. *Computer Graphics International, 2004. Proceedings*, pages 93–99, June 2004.
14. S. L. Hendrix. Popup Workshop: Supporting and Observing Children's Pop-up Design.
15. Popup Workshop. `http://l3d.cs.colorado.edu/~ctg/projects/popups/`.

16. S. L. Hendrix and M. A. Eisenberg. Computer-assisted pop-up design for children: computationally enriched paper engineering. *Adv. Technol. Learn.*, 3(2):119–127, 2006.
17. D.A. Carter and J. Diaz. *The Elements of Pop-up: A Pop-up Book for Aspiring Paper Engineers*. Little Simon, 1999.