Eurographics Workshop on Graphics for Digital Fabrication (2016)
A. Medeiros e Sá, N. Pietroni, and K. Rodriguez Echavarria (Editors)

Foldlings: A Tool for Interactive Pop-up Card Design

Nook Harquail Marissa Allen Emily Whiting

Dartmouth College, USA

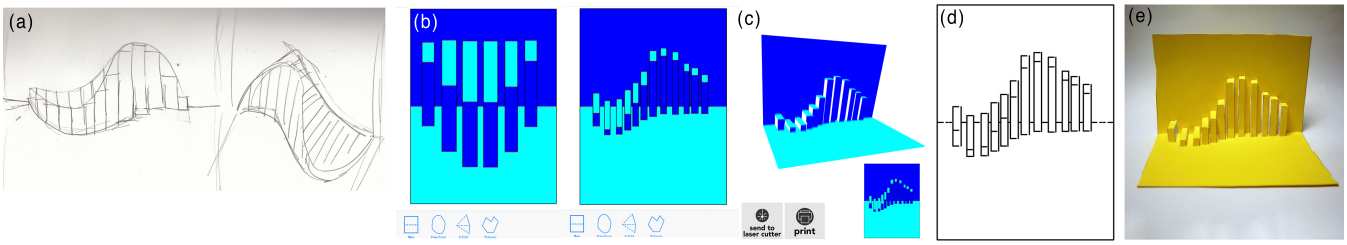

Figure 1: *A full outline of the design process for Foldlings, from initial concept sketches to the full realization of a paper pop-up card. (a) Concept sketches, (b) two 2D designs made in Foldlings' sketch interface, (c) 3D preview of the folded card, (d) SVG output file, (e) final cut and folded card.*

Abstract

Crafting a 3D paper pop-up is a creative and playful experience, which can help develop spatial reasoning skills. However, designing the cuts and folds is often a frustrating trial-and-error process due to the tight set of geometric constraints. We introduce Foldlings: an iPad application that assists in this exploratory process. Our tool-based approach allows users of all skill levels to create complex cards with ease, by separating folding geometries into logical units. We present a novel user interface for creating popup cards from user input, leveraging the practical two-dimensional format of paper. Our approach uses a modular set of simple drawing tools, which can be combined to create complex designs. We guarantee the validity of the folded card at all stages of design, and our interface provides an intuitive set of visual aids to help users understand the relationship between 2D patterns and 3D geometry. We describe the algorithms and data structures used in interpreting 2D user input and visualizing the 3D geometry of the folded card. We discuss results of user studies, which guided early stages of UI design and demonstrate the efficacy of the final prototype.

1. Introduction

Kirigami is the art of papercraft originating from 17th century Japan [TT78]. Kirigami structures vary widely in form and scale — from sub-microscopic creations [GM15] to large sculptures [AFJ*]. Kirigami can represent a wide range of 2D and 3D constructions, e.g., a design can be a 2D lace-like pattern, or a large architectural structure; the defining restriction is that the paper design does not require glue or other attachments for construction. Further constraints define subsets such as origami, which allows folds but no cuts. In this paper we introduce Foldlings: an iPad application that assists in the exploratory process of creating 3D pop-up cards. In the tradition of kirigami, our approach focuses on designs that can be cut from a single piece of paper, without the use of glue or attachments [TT78].

Traditionally, users create pop-up cards manually. Users might sketch out a shape on a card in pencil, and then measure with a ruler to determine where to place folds. Or, they might fold the paper while cutting, discovering correct fold positions experimentally. The second method works well for simple designs, but becomes difficult with complex and nested geometries. Constructing pop-up cards manually is challenging for several reasons:

1. **2D to 3D visualization**. It is nontrivial to understand the structure of a folded 3D card based on its 2D cut and fold pattern.
2. **Geometric constraints**. Pop-up cards present strict constraints for foldability, which are often unintuitive to novice designers.
3. **Physical constraints**. The paper medium presents physical limitations on where edges can be placed.
4. **No "undo"**. Pop-up card design is a trial-and-error process.

Pop-up cards present a constrained problem with opportunities for both interface design and algorithm innovations. Our tool aims

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.

DOI: 10.2312/gdf.20161076

delivered by EUROGRAPHICS DIGITAL LIBRARY www.eg.org diglib.eg.org

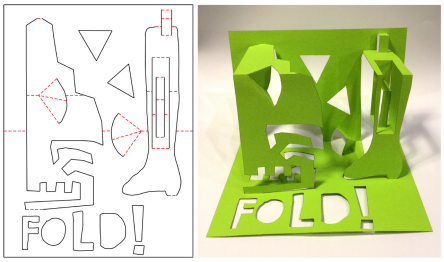

Figure 2: *A pop-up card design created with our software.*

to make the design process fluid and fun for users with all degrees of design experience. Rather than requiring users to draw individual cuts and folds, our system creates modular folding units, which solve geometric constraints transparently. We believe our tools-based system allows for a high level of creativity, while guaranteeing a valid, foldable pop-up card. We demonstrate our work with a prototype iPad application.

Overview. Figure 1 overviews the design process, from concept sketches to a physical pop-up card. To begin, a user draws a design using the Fold Feature tools: Box, Polygon, Freeform and V-fold. These tools create a pattern of cuts and folds, displaying an interactive preview of the 2D design. The cuts and folds remain associated with their containing Feature, and can be modified or deleted as a unit. Each time a new shape is added to the design, it is evaluated for validity. The sketch also maintains hierarchies of the planes within each Feature, which determines fold orientations (mountain vs. valley) in the constructed card.

During the design process the user may bring up a 3D preview, which displays an animation of how the design will fold. The user is free to continue editing his/her design or output a SVG vector file. To fabricated the design, the user may print and cut the final pattern using a laser cutter or other cutting tool, then fold the card by hand.

1.1. Related Work

Generating 3D geometry from 2D sketches is an active and vibrant area of HCI and graphics research [IMT07, PJT02, WWY03]. We solve a far more tightly-constrained problem, in that we are concerned with foldable pop-up kirigami, rather than free-form 3D meshes. Previous work in pop-up card design has involved many variations on the definition of the art style. The seminal work on pop-up card design uses glue to connect components [Gla98], and so does not meet our criteria for kirigami. Glue-based design requires an assembly stage that disconnects the initial 2D pattern from the final pop-up geometry.

Alternatively, previous work also approaches pop-up design by modeling directly in 3D space [RLYL14, LSH*10, ADD*13], where user-defined 3D models are transformed into paper pop-up patterns. [WHS13] segments a 3D model and then uses shape recognition to create paper models in pop-up cards. Because these algorithm modify the input geometry, the end result is not guaranteed to preserve the user's design intent. Users are also less likely to gain an intuitive understanding of the geometric constraints imposed by paper. Further, these methods require the user to have expertise with 3D-modeling software, and thus are not accessible to novice users.

Other approaches present algorithms for creating v-style pop-ups [LJGH11]. V-style planes have different angle constraints than orthogonal pop-ups, allowing for a different set of designs fulfilling geometric constraints. [OI09] describe a program to design elaborate 180 degree pop-ups, they also implement collision testing between card components. However, both these approaches require multiple sheets of paper attached via glue and therefore do not fall under our style constraint of kirigami.

Closest to our work, [HE*06] aims to help users create valid designs and visualize their constructions in 3D. However, the tool was built for educational purposes rather than aiding artistic design. Thus the UI is more restrictive, e.g. only rectilinear cuts are supported. While the software includes Feature-based tools, it does not permit modular modification of design elements after creation.

Related to pop-up design, [XKM07] describe algorithms for creating 2D cut patterns from input images, allowing designs to be composed from multiple composited shapes. Although their result is strictly two-dimensional, we share their interest in producing paper art through software. Similarly, [JGDH12] present an interface for creating precise laser-cut designs by interpreting and smoothing user sketches. The Paper3D system [PDRK14] offers a similar motivation to ours of using 2D gestural interfaces to make modeling tasks more accessible. In contrast to our folded single-sheet designs, their focus is on modeling developable surfaces that are assembled into complex scenes.

In addition to pop-up card design, several authors create tools for origami design (only folds, no cuts). [Fas09] describe "eGami," which provides a toolset of common operations and displays an interactive preview. [KI08] present a web-based tool for origami design, built on the work of [Zam94]. In [JBF*02] the user physically folds the card, while a sensing system provides feedback on how closely the card matches a target design. Such feedback helps reproduce an existing design, but does not facilitate new creations.

1.2. Design Approach

Throughout the UI design process, we explored the balance between creativity and rigid geometric constraints. One of the primary goals of our software is to keep the user's sketch in a valid state while still providing flexibility and creativity. Our primary contributions are:

- Tool-based feature creation allowing for complex geometry while solving geometric constraints transparently.
- Visual aids to assist the user in spatial reasoning both during the design process, and when folding the physical card into the pop-up configuration.
- A prototype application built for iPads, demonstrating sketching capabilities in an intuitive 2D environment.

We chose to design for tablets to leverage the touch-based interface and create interactions that are "natural", mimicking the act of sketching on paper. We strived to design an interface that is modular

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.

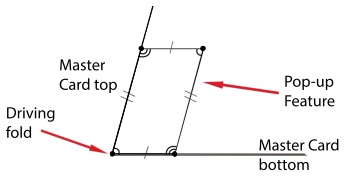

Figure 3: *Cross-section of a pop-up card illustrating parallelogram constraint for valid foldability.*

and friendly. Modularity stems from the conception of the pop-up card as a collection of discrete units that can be acted on individually. Modularity allows users to think in terms of shape constructions, without concern for individual cuts and folds. Friendliness is seen in the careful structuring of our experience to make getting started as painless as possible and evoking the spirit of exploration associated with casual papercraft.

2. Feature-Based Design

Foldlings uses a tool-based system for card design. Each tool creates a specific type of *Feature* — a collection of planes that has a validated folding behavior. Each fold Feature is a modular design element that can be individually created, modified, and deleted.

2.1. Overview

Master Card. Each sketch is initialized with a single Master Card, which is the ancestor of all drawn Features. The Master consists of top and bottom planes with a valley fold between them, representing the blank folded card.

Driving Fold. Each Feature is associated with a *driving fold*, which plays a central role in defining the geometric constraints for how a Feature will fold (Fig. 3). The driving fold is identified as an edge that is spanned by the Feature. A Feature spans a folding edge when its top and bottom folds lie on either side as shown in Fig. 5.

There is a many-to-one relationship between Features and their driving fold: an edge can be the driving fold for more than one Feature, but each Feature has only one driving fold. Driver relationships are also captured in parent-child hierarchies (described in Sec. 3): a Feature's parent always contains its driving fold. In the special case when an input shape does not span a driving fold, the shape is interpreted as a hole rather than a folding Feature.

2.2. Input Methods

We offer four input methods for sketching Features: Box, Freeform, Polygon, and V-fold (Fig. 4). These type classifications help users design complex cards while maintaining valid foldability at all stages of design.

Box. The Box Feature is a set of two planes, axis aligned with the Master Card and connected by its top and bottom edges (Fig. 4 top-left). This Feature is defined by its diagonally opposite corners which specify the bounds of the box. The user specifies size and positioning by a simple dragging gesture, diagonally across the touch screen. Side edges are denoted as cuts, and horizontal edges are folds. A middle fold line is placed according to geometric constraints of foldability. Boxes may fold either inward or outward from the card, depending on its placement relative to other Features (e.g. a set of nested Box Features will alternate).

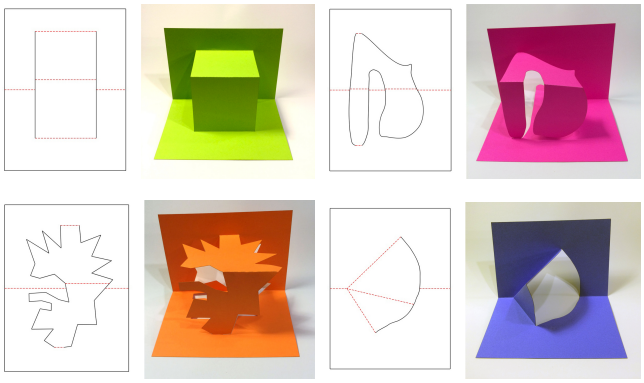

Figure 4: *Fold & cut pattern of features alongside the laser-cut model. Clockwise from top-left: Box (green), Freeform (pink), V-fold (blue), and Polygon (orange).*

Freeform. Freeform Features are defined by a single, closed path (Fig. 4 top-right). We construct a smooth boundary using spline curves [CR74] and capture interpolation points as a function of touch velocity. Free-form shapes are created by a single continuous drag gesture, defining a closed shape.

In contrast to the Box Feature, Freeform shapes must be modified to create top and bottom fold lines. When the initial Freeform shape is completed by the user (by releasing the touch), the shape is then *truncated*: horizontal folds are added at the top and bottom of the shape (Fig. 5) and the boundary is trimmed accordingly. The top and bottom folds are selected to yield fold lines longer than the minimum fabricatable edge length. If the user desires different truncation positions, they can drag folds after the Feature is added to the sketch. Holes are not truncated since they are cut out from the final design and do not require fold lines.

Polygon. Polygons are created from a list of "tap points" constructed from user input. The boundary path consists only of straight line segments. Users can also drag existing points in the polygon to modify the shape. As in Freeform shapes, the path is truncated (if a driving fold exists) in order to create the top and bottom fold lines. Like Freeform shapes, Polygons that do not have a driving fold are considered holes.

V-Fold. V-folds are defined by a vertical cut that crosses a driving fold. This path can be any arbitrary shape that crosses the driving fold once. The fold lines to complete the Feature are specified by adding a point **p** on the driving fold (see Fig. 5). From this input, we construct three diagonal folds fanning out from **p**, analogous to the top, bottom and middle folds seen in previous Features.

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.

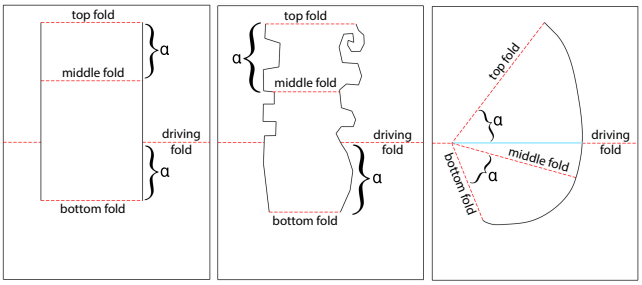

Figure 5: *Geometric constraints. (Left) Box Feature, (middle) Freeform and Polygon Features, and (right) V-fold Feature.*

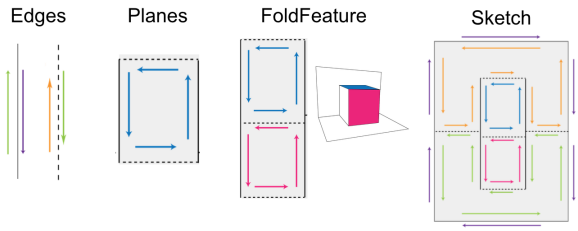

Figure 6: *Data structures used to represent card designs.*

2.3. Constraints

Constraints are central to the implementation of Foldlings' algorithms, which ensure that designs will fold correctly. These constraints are also the core challenge in creating designs manually. Our primary contribution is developing an interface that resolves constraints automatically while giving flexibility to the user.

Foldability of the Box, Freeform and Polygon Features are constrained by the vertical positions of its fold lines (Fig. 5). The vertical distance, a, between the bottom fold and the driving fold must equal the distance between the top fold and the middle fold. For Freeform and Polygon Features, this constraint is applied after truncation is performed.

The constraint on V-fold Features is based on angle rather than height. The simplest case is a symmetric V-fold. In this case, the top and bottom angles of the fold are equal. In the general case, Glassner [Gla98] demonstrates that for any "single slit mechanism," the angle from the driving fold to either the top or bottom fold must be equal to the angle between the opposite fold and the middle fold. This constraint is only solvable if the sum of the angles between each diagonal fold and driving fold is less than 180 degrees.

2.4. Feature Operations

Additional operations can be performed on Features, either during the initial sketching phase or when returning to a Feature to edit.

Self-intersecting Paths. In order to be fabricatable, paths cannot have self intersections. We attempt to repair occurrences by trimming the boundary at the point of self-intersection.

Overlapping Features. We observed the need for overlapping Features during a preliminary user study. To allow these kinds of operations, we draw the new Feature "on top" of existing Features in the Sketch, occluding existing edges. See Fig. 13 for examples where nested Features are used, which is a form of overlap.

Feature Edit. To support design exploration, we allow the user to edit a Feature after it is created. Different options are presented based on the Feature type and state. For example, leaf nodes in the Feature tree have a "Drag Folds" option. That is, you can move folds within a Feature that has no children. The restriction to leaf nodes avoids invalidating child Features, which may occur depending on their fold positions. Implementing fold dragging in Features with children is left for future work.

3. Data Structures

We refer to several data structures which are primary means of storing user input, and are processed by our algorithms to draw designs in 2D and preview them in 3D.

Edges. An *Edge* represents a cut or fold, and is the basic building block of sketch patterns. An Edge can alternatively represent a straight line segment or a Bezier path in the case of Polygons. We use a doubly-connected Edge list (DCEL) to support connectivity relationships (Fig. 6). Traditionally, kirigami patterns indicate fold direction as "mountain" or "valley" where mountains fold outward from the Master Card and valleys folds inward [Cha86]. We determine edge orientations by traversing the Plane tree structure.

Planes. Planes are an enclosed shape, defined and bounded by a closed loop of Edges. They are detected from edges by traversing the directed Edge graph. In order to preview a folded design in 3D, we construct parent-child relationships between the Planes, which determine fold orientations. A Plane's parent is identified by the Plane that contains its topmost edge (Fig. 7). Orientations alternate when the parent connects across a fold. We color code Plane orientations (horizontal or vertical) to better help the user's spatial reasoning of their design in 3D. For visualization purposes, holes are also considered distinct planes. They are bounded by a continuous cut (no folds) and are parented by their containing Plane.

Fold Features. The central data structure of Foldlings is the fold Feature: a representation of a shape drawn by the user that folds in 3D. A Feature is a collection of Planes and has a validated folding behavior. Each fold Feature is a single design element and can be individually created, modified, and deleted. There are four subclasses of FoldFeature: Box Fold, FreeForm, Polygon, and V-Fold, representing differences in drawing behavior, geometry, and appearance.

Hierarchy. We maintain two types of hierarchies as illustrated in Fig. 7. The first is Feature hierarchy: each Feature can have other Features as children. The relationship is determined by the concept of the driving fold. When a Feature is drawn spanning a fold f, its driving fold is set as f, and its parent is the Feature that contains this driving fold.

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.

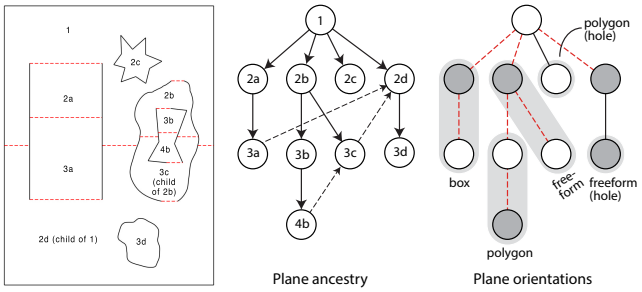

Figure 7: *(Left) Cut and fold pattern of a valid sketch, with Planes numbered by ancestry. (Middle) Ancestry illustrated in tree hierarchy. Dotted lines show alternative pathways. (Right) Plane orientations assigned: white is vertical, gray is horizontal when folded. Fold vs. cut lines are indicated by red dashed lines vs. black solid lines. Orientations alternate when the parent connects across a fold. Feature groupings are also indicted (e.g. Box).*

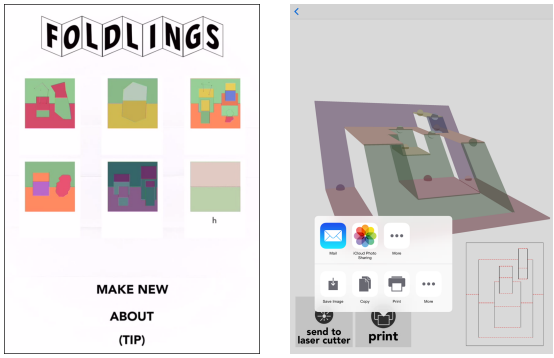

Figure 8: *Prototype UI/UX components: (Left) Saved sketches displayed on the main screen. (Right) Options for sharing a fold pattern from the 3D preview.*

The second type of hierarchy is parent-child relationships between Planes. Each Plane has one or more children, forming a branching tree that starts at the top Plane of the Master Card and ends with the Master Card's bottom plane as one of its leaf nodes. The Plane hierarchy is essential for rendering the scene in 3D.

Sketches. A *Sketch* is the representation of the user's drawing, it contains all of the Features, Planes, and Edges our system creates in defining a pop-up card design. It also contains information about the current drawing state, and the state of user interaction (for example, which features are currently being modified, and which tool is selected).

When a Sketch is initialized, we immediately create the Master Card, which consists of the top and bottom driving Planes that allow the card to fold. The user draws a design by adding new Features to the Sketch, progressing toward a novel pop-up card design.

4. User Interface Implementation

A core component of Foldlings is the interface. To create Features, we capture touch input, display a preview of fold Features as the user creates them, and visualize the 3D design.

Touch Handling. In order to create Features, we first need to capture touch input. Foldings handles pan gestures and tap gestures. The implementation of pan delegate methods varies depending on which Feature is selected. For example, a diagonal pan with the Box tool selected would form a Box between the start and end point, whereas the same touch in the Freeform tool would simply create a diagonal line defining a section of the boundary.

Fold Feature Preview. While the user is drawing, the SketchView displays a 2D preview of the *active* (in-progress) Feature to give interactive feedback. For Box Features, we display a preview of all the edges, but do not occlude the middle fold until the feature is completed. For Freeform Features, we do not perform truncation until the feature is completed (and spans a driving fold). As a preview, the SketchView shows the user's touch path as a cut. For Polygons, we display control circles at vertices. These circles indicate that the vertices are draggable, modifying edge endpoints dynamically. For V-folds, we display a dynamic preview of the vertical cut and top and bottom diagonal folds. The middle diagonal angle is calculated when the feature is added to the sketch. The accompanying video demonstrates various stages of user input.

3D Visualization. Our tool shows a 3D preview of the user's pop-up design, which facilitates spatial reasoning and allows users to preview and iterate through designs quickly. To create the 3D preview, we traverse the Plane tree using a depth-first search and insert joints between each Plane. Each fold alternates between valley and mountain orientation.

We create the folding animation using forward kinematics. Each joint is placed on the Plane's top fold. When the angle of the driving fold changes, the joints rotate according to the parallelogram constraint. Each Plane's transformation is in the frame of reference of their parent joint. To preview the opening and closing of the card interactively, the user performs a pinch gesture. The angle of the card is based on the distance between the user's fingertips.

Visual Aids. Foldlings' utility relies on the user understanding how their design will fold. Using data from the visual aids user study (Sec. 6.2), we include cues to help users visualize the 3D design. The primary visual aid is the interactive 3D preview as described in Sec. 4. In addition, we shade planes based on orientation (e.g. cool colors for planes that will be vertical when folded). In the SVG file, we adjust line patterns for folds based on orientation: dotted lines are mountain folds, dot-dash lines are valley folds. From the visual aids user study, we have data suggesting that displaying fold orientation is very helpful to users when folding pop-up cards.

5. Fabrication

Constraints. The Sketches are bound by physical constraints, we take these into account during the sketching process so the user's design is fabricatable. The primary constraint is the precision of the cutting tool, which limits how closely cuts and folds can be drawn

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.

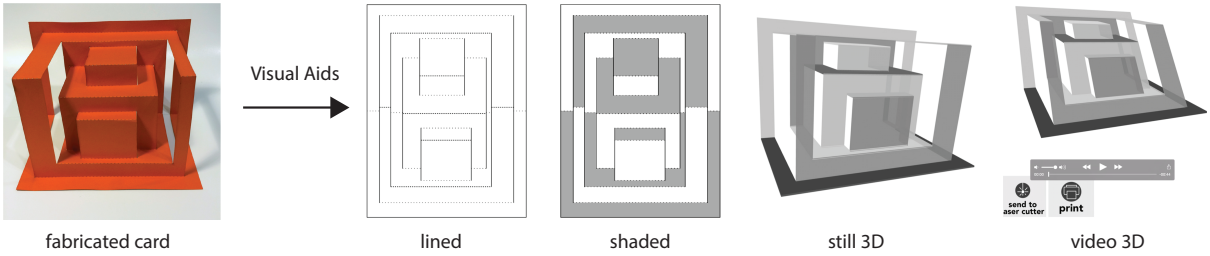

Figure 9: *The four visual aids. "Lined": Edges patterned based on mountain or valley fold. "Shaded": Planes shaded by orientation. "Still 3D": Image of folded card. "Video 3D": Video of folding card.*

to each other. We define a minimum line length, and a minimum distance folds and cuts must be apart. These depend on a number of variables ranging from paper thickness to manufacture method (for example, laser cutters have a higher tolerance for closely-drawn lines). For folds, we adjust the line pattern to consist of short dots spaced out slightly further than the minimum edge distance.

The constraints also prevent users from creating features that are too small to tap, and ensures that edges can be easily cut by hand or using automated methods.

Laser Cutter Input. Our tool is compatible with laser-cutters. We generate an SVG file from the Edges in the Sketch. This file can be fed to a laser cutter, and can be opened in a vector graphics editor to make further changes. Given the vector line data, laser-cutting a design typically takes less than one minute.

6. User Studies

The user tests described in this section were larger-scale studies of key pieces of our software, and were a major driving force in our design process.

6.1. Study: Exhibition Workshop

Method. We tested our system with attendees of a Digital Arts Exhibition at a local College. iPads were provided with Foldlings pre-installed. After a brief demonstration of how to create Sketches and preview their design, users designed cards independently. Upon completion, designs were laser-cut as they waited, so that participants could fold their card as part of the study. Over the course of two hours, users cut and folded 31 pop-up cards.

Figure 10: *(Left) Foldlings at the Digital Arts Exhibition. (Right) A Participant folding a card in our study on 2D to 3D visual aids for pop-up cards.*

People typically spent around 20 minutes at our booth, which was sufficient to design and fabricate a complete pop-up card. The system we demonstrated at the exhibition contained the Box fold and Freeform shape tools, but did not include some advanced features of the final software, such as drag-editing folds or shading based on plane orientation.

Results and Discussion. Sketches generally contained between two and five fold features, the most complex contained ten. Despite their simplicity, sketches showed a wide range of designs, ranging from abstract shapes to representational scenes — users sketched symbols, Chinese characters, and geometric forms. Most of the sketches utilized both Freeform and box fold features, mixing the two element types to create a composition. Roughly one third of the designs took advantage of nesting (constructing fold features inside each other).

Common complaints were the lack of a delete/undo button and that the UI did not show which tool was currently selected. An unexpected behavior was that several users attempted to construct overlapping features. This demonstrated a desire to construct more complex geometry, and we later implemented functionality to resolve intersecting Features. Users also relied on the 3D preview to differing degrees. Some viewed the preview after every operation, while others only switched to the preview occasionally.

Folding the laser-cut sheet also presented difficulties. Although users were able to see a 3D preview of their design while creating it, they had often relinquished the iPad by the time they folded their design. Users often struggled to discover the correct fold orientations. In some cases, it took longer for users to fold their creation than to design it. We conducted further user studies to determine the effectiveness of methods for displaying 3D information, discussed in the next section.

6.2. Study: Visual Aids

The goal of this study was to determine whether users understand the mapping of 2D fold patterns to 3D, and test the degree to which plane shading, edge patterning, and a 3D preview help users understand how a pop-up card will fold.

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.

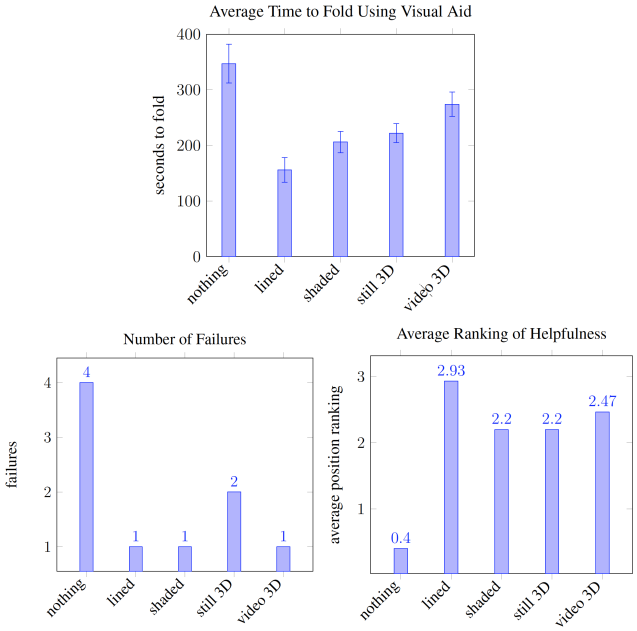

Figure 11: *(Top) Average Time to Fold Using Visual Aids. (Bottom-Left) Number of failed folding attempts. (Bottom-Right) Average ranking of helpfulness.*

Method. We performed this study with 22 participants, who spent an average of 18 minutes with us. Each subject received a set of five laser cut cards, and we recorded the time for them to successfully fold the card. Although there was some variety in age and background, the largest demographic was undergraduate College students. 10 of the participants were male, 12 were female.

For each card, each subject was randomly given one of the following five options (Fig. 9):

1) A 2D design, showing planes shaded according to horizontal or vertical orientation when folded.
2) A 2D design, with edges patterned based on whether they are "hills" or "valleys" (fold toward or away from the card).
3) A video showing an animation of the card folding in three dimensions. Note that we did not test the *interactive* 3D preview.
4) A still image of the card folded in perspective view.
5) No visual aid.

The order of aids and cards was shuffled, and then balanced to ensure an equal distribution of orderings. I.e each visual aid has an equal chance of being the first aid presented to a user and the last aid presented. Finally, we asked subjects to rank the visual aids from most to least helpful.

Results and Discussion. The time to fold varied significantly depending on the visual aid used to fold the card (see Fig. 11). Error bars indicate a 95% confidence interval around the mean, showing a relatively narrow variance within each visual aid data set. From this, we can conclude that trials with no visual aid were slower than all trials with visual aids, and that the lined pattern showing fold

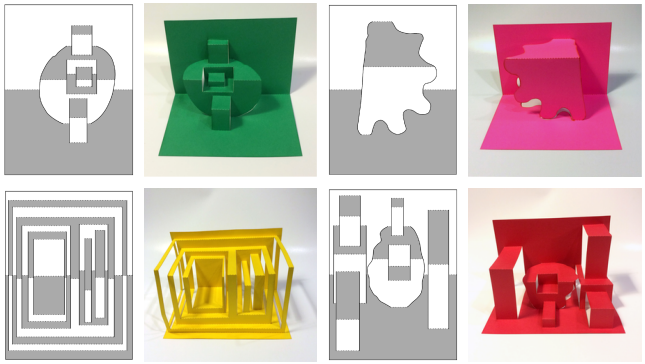

Figure 12: *Examples used in visual aids user study. Folded card shown with shaded visual aid.*

orientation was more helpful than any other preview method. While the shaded visual aid and still 3D preview were indistinguishable, video 3D was slightly worse than all other visual aids.

Some trials were not completed successfully. In some cases, users folded the design incorrectly — in others, they refused to fold the card, feeling lost without a visual aid. Unsurprisingly, users were far more likely to fail to fold the card when they did not have a visual aid. The times for these failures were not recorded in the Number of Failures graph above.

We asked subjects to rank the visual aids from most to least helpful. In this graph, the data is averaged and inverted (5 being the most helpful), to show the relative helpfulness of each aid. Participants rated the lined aid as the most helpful — with small differences among the other aids. The lack of visual aid rated very poorly.

This study gives persuasive evidence for the relative helpfulness of lines showing fold orientation. A hypothesis is that the lined visual aid was most valuable to this task because it was closest to the action participants performed: fold orientation is closely related to the action of folding a card.

In Foldlings, we implement many of these visual aids. In the 2D sketch, we shade planes based on orientation, and display a 3D preview that users can manipulate interactively. As a result of our findings in this study, we also implemented line patterning based on fold orientation for the SVG export.

6.3. Future Work

Currently, our software operates on a single piece of paper. In order to support combinations of interlocking sketches and complex arrangements of Features [H*07], we would like to create an interface for connecting pop-up card elements in layered 2.5D or 3D. As a practical addition, we would also like to add constraints so that Features do not extend beyond the bounds of the Master Card when fully folded. Such "safe area" guidelines would allow e.g. greeting cards that are guaranteed to fit in standard envelopes when folded.

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.

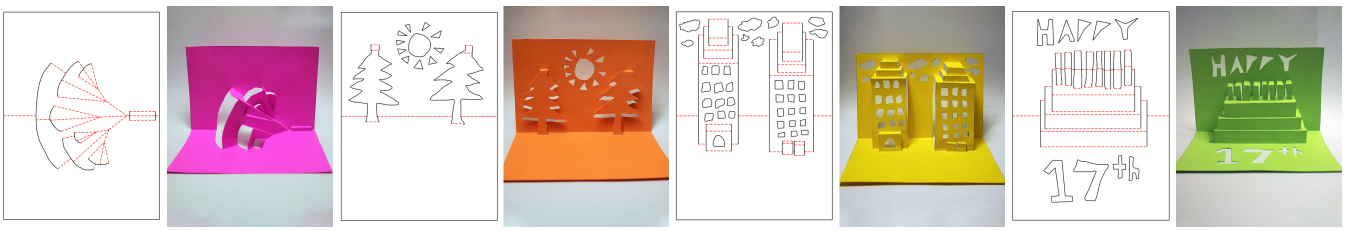

Figure 13: *A sample of cards designed using Foldlings. The cards were created using a combination of Polygon features, nested Box-folds, nested V-folds, and Freeform shapes.*

7. Conclusion

In summary, our tool helps users visualize and iterate custom 3D pop-up designs. Users found the process of designing complex pop-up cards more intuitive with Foldlings than with traditional methods. We have demonstrated that novice users can create a wide range of precise cards, enabled with preview capabilities, an easy to learn UI, and a final pattern appropriate for laser cutter fabrication.

Acknowledgments

We thank Tim Tregubov for his early role in the project and helpful input throughout; Lorie Loeb, Jodie Mack and Xing-Dong Yang for insightful advice; Runi Goswami for initial user tests; and the Digital Arts Masters students for their support. Kevin Baron and the Thayer Machine Shop provided laser cutter facilities. The DALI Lab provided an iPad for user studies. This project is partially supported by the National Science Foundation under Grant No. 1464267.

References

[ADD*13] ABEL Z. R., DEMAINE E. D., DEMAINE M. L., EISENSTAT S. C., LUBIW A., SCHULZ A., SOUVAINE D. L., VIGLIETTA G., WINSLOW A.: Algorithms for designing pop-up cards. 2

[AFJ*] ANDREWS B., FELIX D., JOSHI K., MEHTA K., NOVINYO E., HIGGINS L., SHOCKEY I., SINGH O., BALAKRISHNAN V.: Creating a kirigami shelter prototype for migratory populations of himachal pradesh, india. 1

[Cha86] CHATANI M.: *Pop-up greeting cards: a creative personal touch for every occasion ; origamic architecture*. Ondoriha, Tokyo, 1986. 4

[CR74] CATMULL E., ROM R.: A class of local interpolating splines. *Computer aided geometric design 74* (1974), 317–326. 3

[Fas09] FASTAG J.: egami: Virtual paperfolding and diagramming software. In *Origami4: Fourth International Meeting of Origami Science, Mathematics, and Education* (2009), pp. 273–283. 2

[Gla98] GLASSNER A.: *Interactive pop-up card design*. Tech. rep., Miscrosoft Technical Report, 1998. 2, 4

[GM15] GROSSO B. F., MELE E.: Bending rules for nano-kirigami. *arXiv preprint arXiv:1507.01805* (2015). 1

[H*07] HART G., ET AL.: Modular kirigami. *Proceedings of Bridges Donostia* (2007). 7

[HE*06] HENDRIX S. L., EISENBERG M., ET AL.: Computer-assisted pop-up design for children: computationally enriched paper engineering. *Advanced Technology for Learning 3*, 2 (2006), 119–127. 2

[IMT07] IGARASHI T., MATSUOKA S., TANAKA H.: Teddy: a sketching interface for 3d freeform design. In *Acm siggraph 2007 courses* (2007), ACM, p. 21. 2

[JBF*02] JU W., BONANNI L., FLETCHER R., HURWITZ R., JUDD T., POST R., REYNOLDS M., YOON J.: Origami desk: integrating technological innovation and human-centric design. In *Proceedings of the 4th conference on Designing interactive systems: processes, practices, methods, and techniques* (2002), ACM, pp. 399–405. 2

[JGDH12] JOHNSON G., GROSS M., DO E. Y.-L., HONG J.: Sketch it, make it: sketching precise drawings for laser cutting. In *CHI'12 Extended Abstracts on Human Factors in Computing Systems* (2012), ACM, pp. 1079–1082. 2

[KI08] KASEM A., IDA T.: Computational origami environment on the web. *Frontiers of computer science in China 2*, 1 (2008), 39–54. 2

[LJGH11] LI X.-Y., JU T., GU Y., HU S.-M.: A geometric study of v-style pop-ups: theories and algorithms. In *ACM Transactions on Graphics (TOG)* (2011), vol. 30, ACM, p. 98. 2

[LSH*10] LI X.-Y., SHEN C.-H., HUANG S.-S., JU T., HU S.-M.: Popup: automatic paper architectures from 3d models. *ACM Transactions on Graphics-TOG 29*, 4 (2010), 111. 2

[OI09] OKAMURA S., IGARASHI T.: An interface for assisting the design and production of pop-up card. In *Smart Graphics* (2009), Springer, pp. 68–78. 2

[PDRK14] PACZKOWSKI P., DORSEY J., RUSHMEIER H., KIM M. H.: Paper3D: Bringing casual 3D modeling to a multi-touch interface. In *UIST 2014: User Interface Software and Technology Symposium* (October 2014), ACM. 2

[PJT02] PATRICK M., JOYCE C., THOMAS F.: A 2d sketch interface for a 3d model search engine. In *Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH* (2002). 2

[RLYL14] RUIZ C. R., LE S. N., YU J., LOW K.-L.: Multi-style paper pop-up designs from 3d models. In *Computer Graphics Forum* (2014), vol. 33, Wiley Online Library, pp. 487–496. 2

[TT78] TEMKO F., TAKAHAMA T.: *The Magic of Kirigami: Happenings with Paper and Scissors by Florance Temko and Toshie Takahama*. Japan Publications, Incorporated, 1978. 1

[WHS13] WAY D.-L., HU Y.-N., SHIH Z.-C.: The creation of v-fold animal pop-up cards from 3d models using a directed acyclic graph. In *Advances in Intelligent Systems and Applications-Volume 2*. Springer, 2013, pp. 465–475. 2

[Woo01] WOOD D.: Scaffolding, contingent tutoring, and computer-supported learning. *International Journal of Artificial Intelligence in Education 12*, 3 (2001), 280–293.

[WWY03] WANG C. C., WANG Y., YUEN M. M.: Feature based 3d garment design through 2d sketches. *Computer-Aided Design 35*, 7 (2003), 659–672. 2

[XKM07] XU J., KAPLAN C. S., MI X.: Computer-generated papercutting. In *Computer Graphics and Applications, 2007. PG'07. 15th Pacific Conference on* (2007), IEEE, pp. 343–350. 2

[Zam94] ZAMIATINA L.: Computer simulations of origami. *Journal/Anthology 3* (1994). 2

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.