Submission to GMP 2018

Folding Cartons: Interactive Manipulation of Cartons from 2D Layouts

Submission number: 1074

Abstract

Cartons are broadly used in packaging industry and our daily life. While most of traditional packages are axis-aligned cuboids, there is a recent trend of creating paper packages and cartons with more complicated geometry. In this work, we present an interactive system to manipulate 3D cartons from 2D layouts. Two key components, *automatic folding* and *suggestive editing*, make our system significantly easy to use. The automatic folding part automatically computes the 3D coordinates of vertexes in cartons based on simple observation. Then the suggestive editing part smartly provides a group of geometric constraints, such as vertex merging, merging propagation, and face pasting, for users to explore and manipulate 3D shapes. Moreover, automatic updating of 2D layout is supported when the user edits the carton shape in 3D.

1 Introduction

Cartons have been widely used in the packaging industry to organize, ship and deliver various commodities including food, daily necessities and electronic components. Instead of the very basic packaging shapes like cuboids, there exist multiple fantastic cartons to package wedding candies or wine bottles. These various designs increase much popularity, not to mention they are environmental friendly due to their recycling and degradability [21].

Cartons are usually designed based on experience and trial-and-error. Designers often start package design by generating 2D vector artwork, then a 3D mockup is essential to be made for designers and clients to see the real appearance. There are two ways to create a 3D mockup, a digital mockup is a great way of showing how the design would appear with lower cost, and a practical mockup can be helpful when making sure the size is correct and making final decision.

In recent years, there are software packages developed help designers improve efficiency and productivity. For example, a plug-in for adobe illustrator named STUDIO [8] can generate 3D models by manually assigning angles to folding edges, and with the model, users can turn their ideas into beautiful 3D images. However, the existing softwares still need a lot manual work to design packages. In order to create a virtual 3D model and explore layouts' diversity, our idea is to fold the existing layout directly into 3D model, and by manipulating the corresponding model to reach a diverse layout.

However, based on given planar layout of a carton only, it is nontrivial to formulate the problem of 3D model construction mathematically, because the information extracted from the 2D layout such as edge length and vertex degree may cause ambiguities on account of repeated edges and vertexes. Moreover, Biedl et al. [1] proved that given a polygon and a set of creases, it is NP-hard to know whether a polyhedron can be obtained by folding along the creases, which can also illustrate the difficulty of our problem in some way.

In order to solve the folding cartons just from 2D layouts, we solve a optimization problem given on set of shape constraints. Moreover, an interactive design and exploration framework is developed to allow users to visualize the planar layouts and the corresponding model in a 3D view, and edit the 3D model freely. Figure 1 shows our folding results based on the layouts designed by artists.

The contribution of this paper includes the following:

(1) An interactive and explorative system to manipulate the shape of 3D carton: It creates the corresponding 3D realization by initialization and user interaction, and users are allowed to edit the 3D model to explore deformed layouts automatically.

(2) A collection of shape constraints represented through a set of points to implement shape optimization: Observed from existing cartons, constraints are summarized including face rigidity and coplanarity to keep the shape of cartons. Besides, computer-aid detection like vertex merging and face pasting are brought to give users suggestions.

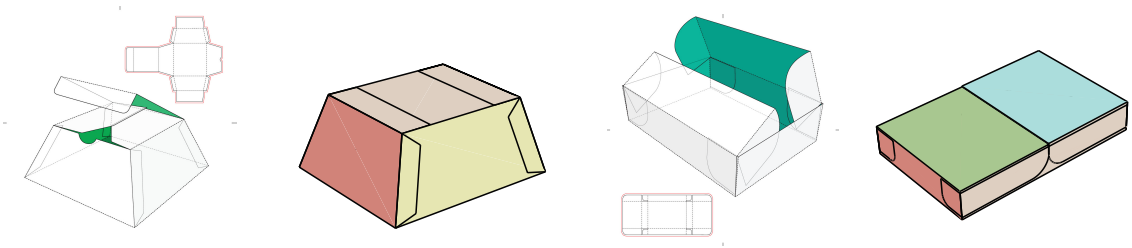


Figure 1: Two cartons and their layouts designed by artist, and the corresponding model folded by our system.

2 Related Work

2.1 Paper folding problem

Various types of paper crafts have been studied in the field of computation and mathematics. Origami is the Japanese traditional paper art of making different kind of objects by a single sheet of paper, and has been long studied hundreds of years ago[10].The simulation of rigid origami is similar to our carton folding problem, while the folding motion is mostly computed based on the given angle of creases, and consider the geometry of origami in kinetic motion [28, 29]. In this paper, cartons are folded into 3D model without knowing the prior of crease angles, and can be deformed through optimization while origami folding cannot deal with the deformation. The thickness problem of origami is also a direction of origami related research these years [6, 15, 30], while thickness is not considered in our problem.

Curved folding from a single sheet of paper is a variation of computational origami which considers curved lines as a part of creases, and can be treated as a practical instance of developable surface. Kilian et al. [12] presented an optimization based framework to approximate the given geometric data. Solomon et al. [26] provided a subdivision based modeling scheme involving curved paper structure with the folding angle on creases as input. Kilian et al. [13] studied the deformation of curved folded surfaces after the folding motion actuated by pulling a network of string. Despite the 2D expanded layout as an input, curved folding problem always has extra information, such as approximate shape, folding angles or external force, while the carton design layout is our only input.

While researching on origami in the field of computational algorithm and geometric analysis, the simulation system is developed to visualize the folding behaviour of a single piece of paper. Thiel [32] provides a virtual origami system including the user interface to model folded paper and show animations of folding process. Kishi et al. [14] allowed users to create and edit the origami properties over the Web. Nimnual et al. [22] presented an application for package folding practices in a virtual space. Although these applications can model folded paper well, they need given parameters to construct the model. How to align constraints to structural designs by implementing shape optimization is our main concern.

There are also some methods to solve related problems of carton folding. Song et al. [27] modeled foldable objects as tree like multilink objects and used PRMs (probabilistic roadmap methods [11]) to find a sequence of motions to transform one configuration of a foldable object into another configuration. Mullineux et al. [21] provided a simulation framework of the carton during erection using a constraint-based approach. Both these work required the target state as a premise, while our work aim to generate the target configuration.

The complexity of folding to polyhedra problem has also been studied for decades, Lubiw [18] provided an dynamic programming algorithm based on Aleksandrov's theorem to test whether a polygon can be folded into polyhedra which takes $O(n^2)$ time and space. O'Rourke [23] examined three open problems on the subject of folding and unfolding. Biedl et al. [2] studied in polynomial time to solve the question of when is the graph orthogonally convex polyhedra given a graph, edge length and facial angles, also shown that it's NP-hard to decide whether the graph is orthogonally polyhedra or not. Rather than the given graph, Biedl et al [1]. proved that if given a net along with the dihedral angle at each crease, we can know whether a net can be folded to a polyhedron in polynomial time, but it becomes NP-hard without the angles even adding constraints on orthogonal polyhedron, which results in more difficulties on more complex input. Compared to our desired result, a polyhedron is a set of polygons without overlap, nevertheless, our 3D model contains small faces that needs to be fixed to another panel. These works above justify our problem being hard to solve caused by even more intricate inputs.

2.2 Reconstruction from single line drawings

Although with the same goal that constructing 3D models from 2D layouts including vertices and edges as line drawing problem, the construction process is actually different.

A line drawing is defined as a 2D projection of object containing its vertexes and edges. Line drawings of three-dimensional objects have long been studied, and the main problem is still in object reconstruction given its projection on two-dimensional planes. Some researchers treat this task as an optimization problem. Marill [19] proposed MSDA (Minimize the Standard Deviation of Angles) principle to emulate the interpretation of line drawings as 3D objects. This new criterion is used by many other researchers later. Leclerc et al. [16] combined MSDA with the deviation from planarity as objective terms. Cao et al. [5] added a symmetry measure of the objects to get more complicated results. Some other researchers try to solve this problem from the information theoretic point of view. Marill [20] minimized the description length of objects based on the idea that we usually pick the simplest one from infinite possibilities when we see the line drawing. Shoji et al. [25] implemented the principle of minimizing the entropy of angle distribution between line segments using genetic algorithm. Later, the strategy of splitting and merging has been used to solve line drawings of complex 3D objects [17, 33].

Different from the input above, ours are expanded structural layout of three-dimensional objects in 2D planes, the final model is constructed based on the folding process instead of mathematical presentation.

2.3 Shape optimization

The basic idea of our paper is shape optimization by a set of simple algorithms, multiple algorithms have been proposed to enforce shape constraints, and have been used successfully for interactive tools and physical simulation [3, 9]. Bouaziz et al. [4] unified a large variety of geometric constraint into one optimization framework, and provided simple, and robust implementation. Poranne et al. [24] described an interactive method to manipulate and optimize polyhedral meshed with constraints with a linear-time algorithm. Tang et al. [31] solved constrained equations by Newton-type method in a fast way, and provided an interactive system to model meshed constrained by equalities and inequalities. Deng et al. [7] developed an interactive tool to explore architectural design with shape constraints, and provided an optimization method to enforce hard constraints and soft constraints at the same time.

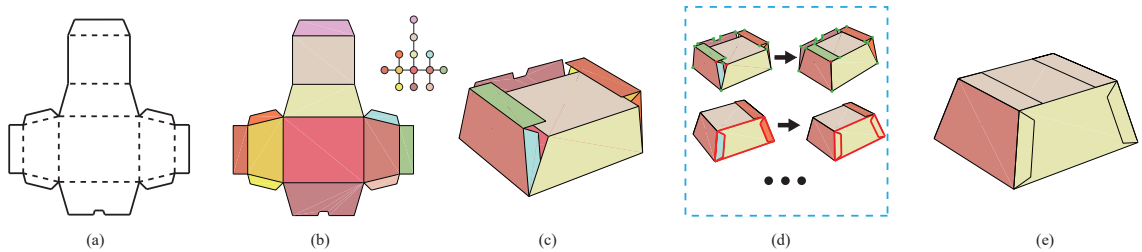


Figure 2: Given a 2D layout (a), we first extract its 2D mesh (b), with different face in different color. By providing each folding edge with a specific angle, we can construct an initial 3D model (c). The final carton model (e) is built through the shape optimization based on the information acquired from user interactions (d).

The studies mentioned above focus on the architectural design with constraints, and proposed different solutions, while our constraints do not need satisfy properties like fairness. As a result, we implement the shape optimization by the method introduced in [4] for its robustness and simplicity.

3 Overview

Figure 2 shows the overview of our algorithm. As shown in Figure 2(a), the input 2D design layout of a carton consists of a set of cutting edges (solid lines) and folding edges (dashed lines). Given the 2D layout, an undirected graph is built and faces are extracted by finding minimum cycles in the graph, as Figure 2(b) shows. The 2D layout therefore can be represented by a polymesh $\mathcal{L} = (V, E, F)$, where V is the set of vertexes, E is the set of all edges, and F is the set of faces. The edge set $E = E_c \cup E_f$, where E_c is the set of cutting edges, and E_f is the set of folding edges. To build a 3D carton model $\mathcal{M} = (V, E, F)$ from the 2D layout L, while they share the same topology, we compute the 3D coordinates of all vertexes in V.

However, it is not intuitive to analytically define the desired final 3D shape from a 2D layout. One possible way is to detect all the possible geometric constraints such as vertex merging, face parallelism, adjacent face orthogonality, and then to integrate all these possible constraints together to form a large equation system. But the challenge is that these local constraints can not well describe complicated and creative designs. Moreover, there are many ambiguities when detecting these constraints in a 2D layout that consists of many repeated edges and faces. We propose a two-step algorithm based on the observation that humans usually fold edges with a right angle to get a rough shape and then merge vertexes or edges to get a stable 3D carton. First, an initial 3D model (Figure 2(c)) is constructed based on a specific angle along each fold edge, as described in Sec. 4.1. The user can then manipulate and explore 3D shapes of the canton based on a series of suggestive operations provided by our system. The final model is shown in Figure 2(e).

4 Algorithm

In this section, we explain our algorithm in detail. Initially, assume that the 2D layout $\mathcal{L}$ lies on the XOY plane, so that each vertex $\mathbf{v}_i(x_i, y_i, z_i)$ in V has $z_i = 0$. Each face in F has a normal $\mathbf{n}_i = (0, 0, 1)^T$. In the first step of our algorithm, our system automatically folds the original flat mesh into a rough model by assigning a specific angle to each folding edge. In the second step, the carton shape is refined based on a set of potential geometric constraints.

4.1 Shape Initialization

A general process for humans to fold a carton starts from folding each edge by a rough angle and then connecting close vertexes to obtain a box-like shape. Naturally, we first fold the 2D layout by assigning a rotation angle to each pair of adjacent faces to get a rough shape, instead of directly computing the vertex coordinates. Observed from existing data in the Internet, most of the traditional cartons are cuboid for holding files or delivering daily supplies. Although there is a recent trend to design more complicated layouts to attract consumers, the shape of these unusual cartons is similar to boxes as their functions are still packaging commodity. Therefore, we choose $\pi/2$ as the initial value of the rotation angle to each folding edge. First, a face graph of a layout is constructed, as shown in Figure 3 (a). Each face is a node, and there is an edge between two faces if they are connected by a folding edge. The faces in the 2D layout are recursively folded in a breadth-first manner. Starting from the face with the maximal area, our system folds all its adjacent faces by rotating them around their connecting edge by $\pi/2$, and then continues to others. Figure 3 illustrates the folding process of a cuboid carton.

A group of results is shown generated by simply folding faces by $\pi/2$. Traditional cartons in cuboid shapes could reach an ideal state, as Figure 4 shows. However, many complicated designs still need refinement, such as the four examples in Figure 5. Take the hexagonal box as an example, the 3D model can be improved by simply snapping the small paste faces to its nearby faces which ensures the model is stable without manual intervention.

As a result, we provide an suggestive interface by automatically detecting potential geometric modifications in the rough carton shape for users to explore.

4.2 Shape Refinement

Though not perfect, the initial 3D shape provides a good start point for generating the final carton model. To further refine the 3D shape, the 3D coordinates of all the vertexes are then refined based on a set of shape constraints, such as vertex merging, face pasting, and so on. In this step, the coordinate of vertexes are chosen as our objective instead of

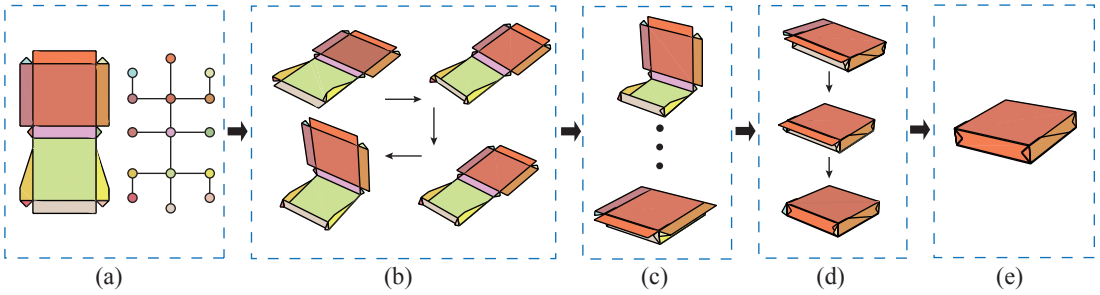


Figure 3: A rough 3D shape is obtained by folding each face by $\pi/2$ in the 2D layout (a) in a breadth-first manner, starting from the face with the maximal area. (b), (c), (d) and (e) are the different folding stages in the initialization step.

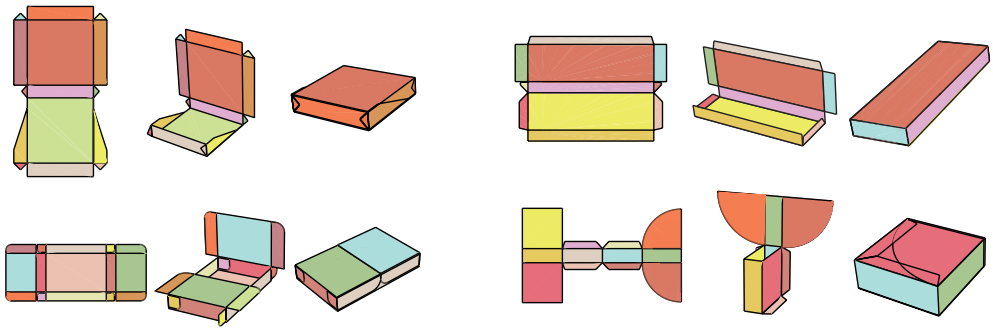


Figure 4: Four examples of carton models that can be fully automatically generated from 2D layouts by folding each edge with a fixed angle $\pi/2$.

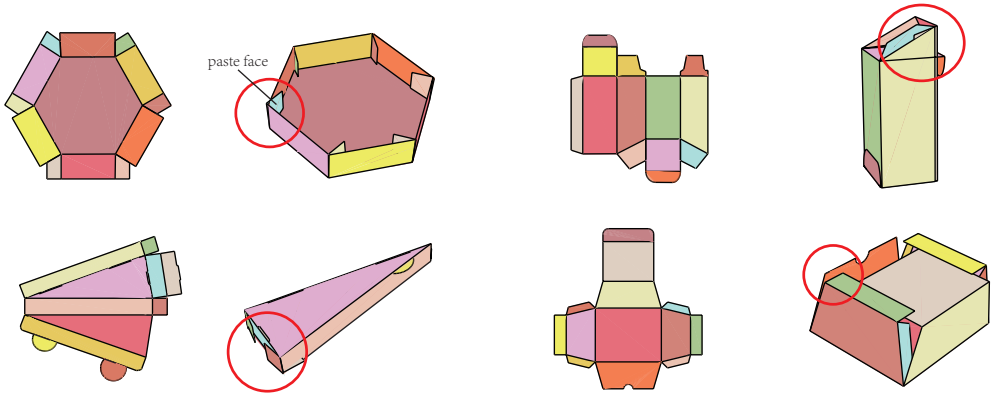
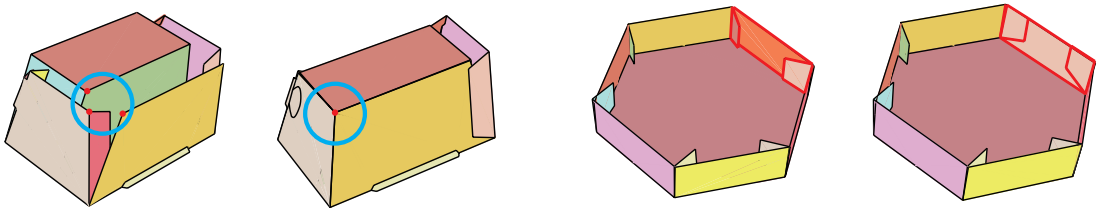


Figure 5: Four examples of the carton models that need shape improvement. While the shapes are not cuboid, using a fixed angle $\pi/2$ leads to non-closed shape and loose structure.



(a) Vertex Merging (b) Face Pasting

Figure 6: 3D shape refinement based on merging vertexes (a), or pasting faces (b). The locations where the 3D shape changes are highlighted in blue.

angles on folding edges, mainly because that the geometric constrains in 3D shapes can be more simply and intuitively represented by 3D vertexes.

A suggestive interface is provided to users for efficiently exploring better carton shapes, as described later in Sec. 5. Once the user selects a suggested shape refinement operation, we optimize the 3D shape based on a series of geometric constraints. Given the current mesh whose vertex positions are defined as $\{\hat{\mathbf{v}}_i\}_{i=1}^N$, a new mesh with the same topology but new vertex positions $\{\mathbf{v}_i\}_{i=1}^N$ will be constructed. The geometric constraints can be classified into two groups, shape rigidity constraints and shape modification constraints. First, to keep the rigidity of each face, the constraints including face rigidity and coplanarity, which are corresponding to similarity constraint and plane constraint described in [4]. Second, once vertexes or faces are confirmed to be merged to modify the rough shape, more constraints are added. Each constraint is defined as following.

Face rigidity Each face keeps its shape unchanged during folding. This constraint is defined by keeping the length of each line connecting any pair of points $\mathbf{v}_a, \mathbf{v}_b$ on the face the same.

$$||\mathbf{v}_a - \mathbf{v}_b||^2 = ||\hat{\mathbf{v}}_a - \hat{\mathbf{v}}_b||^2. \tag{1}$$

Coplanarity For each face in $\{f_k\}_{k=1,\dots,P}$, the coplanarity constraint specifies all vertexes in a face should always lie on a plane. We can compute the sorted eigenvectors $\mathbf{U} = [\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3]$ of the 3×3 covariance matrix $\mathbf{C}^T\mathbf{C}$ where $\mathbf{C} = \{\mathbf{v}_{kj}\}_{j=1}^{N_k}$, $\mathbf{v}_{kj}$ is the jth vertex among N_k vertexes in the face. By removing the last column of $\mathbf{U}$, we can implement plane projection as [4] described.

If a set of shape modification operations, such as vertex merging and face pasting, are selected, more constraints are added to optimization vertex positions.

Merging vertices For any two vertexes $\mathbf{v}_p$ and $\mathbf{v}_q$ that are selected to be merged as the same vertex, we have

$$\mathbf{v}_p - \mathbf{v}_q = \mathbf{0}. \tag{2}$$

Face pasting If two faces f_a and f_b need to snap together, the whole vertexes of these two faces should satisfy the coplanarity constrain.

In order to reduce the solution space of the optimization problem, soft constraints will be added to keep the original positions of the vertices that are not relative to the shape modification.

Irrelevant vertexes For the vertices $\{\mathbf{v}_i\}$ are not in the same plane with any vertex for merging or face pasting, they should stay at their original location. We add these soft constraints by adding a small weight w, which is set 0.001 in our experiments to the following equation.

$$\mathbf{v}_i - \hat{\mathbf{v}}_i = 0. \tag{3}$$

Combining the above constraints, the new vertex locations can be solved by minimizing the proximity function based on projection operators[4]. For a single point $\mathbf{v}$, the proximity function is defined as

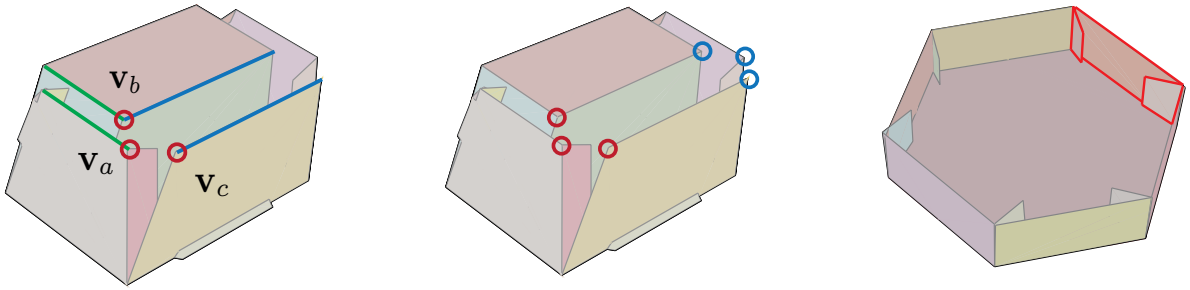
$$\phi(\mathbf{v}) = \sum_{i=1}^{m} \omega_i d_i(\mathbf{v})^2, \tag{4}$$

where ω_i are non-negative weights, and d_i is the distance between the point $\mathbf{v}$ and its projection onto the constraint set, m is the number of constraints. For all the vertices $\mathbf{V} = \{\mathbf{v}_i\}_{i=1\dots n}$, the shape proximity function is formulated as

$$\phi(\mathbf{V}) = \sum_{i=1}^{m} \omega_i ||\mathbf{N}_i\mathbf{V}_i - P_i(\mathbf{N}_i\mathbf{V}_i)||_2^2, \tag{5}$$

where ω_i are weights and $P_i(\cdot)$ is the projection onto the constraint. The matrix $\mathbf{N}_i$ is used to center the vertices at their mean.

Figure 6 shows two examples of shape refinement by merging three vertexes and pasting three faces respectively.



(a) Vertex Merging (b) Merging Propagation (c) Face Pasting

Figure 7: Three different suggestions on shape modification. (a) A group of vertexes (marked in red) are detected as mergeable. $\mathbf{v}_a$ and $\mathbf{v}_b$ are close to each other and one edge connecting to $\mathbf{v}_a$ has the same length with one edge connecting to $\mathbf{v}_b$ (green edges). $\mathbf{v}_c$ is also considered to be merged with $\mathbf{v}_a$ and $\mathbf{v}_b$ because $\mathbf{v}_c$ is close to $\mathbf{v}_b$ and one edge of $\mathbf{v}_c$ has the same length as $\mathbf{v}_b$ (blue edges). (b) Once the user confirms a suggested vertex merging operation, our system automatically proposes a symmetric group of vertexes that can be also merged (marked in blue). (c) If faces marked in red are detected close to each other and the same with their normals, our system suggests these faces can be sticked together.

5 Suggestive Interface for Shape Modification

For some creative designs, the rough 3D carton model generated based on our heuristic approach still needs some improvement or the user might edit the carton shape. To assist users to refine the carton model, our system is equipped with a set of smart shape refinement operations, such as automatically detecting vertexes that can be merged, propagating modification to similar groups, and providing suggestions for users to explore. While a user edits the carton model in 3D space, which is more intuitive, our system is capable of revising the 2D layout automatically.

5.1 Suggestive Interaction

Our system automatically detects a group of potential shape constraints to improve the 3D carton model. These guiding constraints are provided to the user for quickly selecting and manipulating the carton shape. In our system, three types of shape refinement operations, vertex merging, merging propagation, and face pasting, are automatically detected.

Vertex Merging. For irregular shapes which consist of non-rectangular faces or non-perpendicular adjacent faces, some vertexes are close to each other after the initialization step. A practical carton model can be obtained by simply merging these nearby vertexes together if their have coherent edges. Any pair of vertexes $\mathbf{v}_a$, $\mathbf{v}_b$ in different faces is detected as mergeable, if $|\mathbf{v}_a - \mathbf{v}_b| < \epsilon$, and there is an edge connecting to $\mathbf{v}_a$ having the equal length with one edge connecting to $\mathbf{v}_b$, ϵ is a distance threshold and set as 50mm in our experiments. More vertexes can be merged together, as shown in Figure 7(a) by gradually adding one vertex $\mathbf{v}$ each time if $\mathbf{v}$ is mergeable with at least one vertex in current list.

Merging Propagation. Once the user makes a decision to merge a subset of vertexes $\mathbb{V} = \{\mathbf{v}_i\}_{i=1,\dots,K}$ from automatic system suggestions, or manually selection, our system will detect if there exists another subset of vertexes $\mathbb{V}_b$ symmetric to $\mathbb{V}$ and propagate the merging operation to $\mathbb{V}_b$, as shown in Figure 7(b). A subset $\mathbb{V}_b$ is symmetric to $\mathbb{V}_a$ if $|\mathbb{V}_a| = |\mathbb{V}_b|$, and there exists one-to-one correspondence between the vertexes in $\mathbb{V}_a$ and $\mathbb{V}_b$. Here, $|\mathbb{V}|$ is the number of vertexes in a vertex subset $\mathbb{V}$. Two vertexes $\mathbf{v}_a$ and $\mathbf{v}_b$ are defined as corresponding vertexes if they have the same number of edges and one-to-one correspondence can be found between the edge lengths.

Face Pasting. Merging vertexes sometimes are not enough to generate desired model caused by the paste faces as shown in Figure 7(c). A pair of faces f_a and f_b is considered mergable if their normals $\mathbf{n}_a$ and $\mathbf{n}_b$ satisfy $\mathbf{n}_a \cdot \mathbf{n}_b > 0.5$, and for each vertex in $\{\mathbf{v}_{ai}\}_{i=1}^{N_a}$, $|(\mathbf{v}_{ai} - \mathbf{v}_c^b) \cdot \mathbf{n}_b| < \epsilon_d$, where $\mathbf{v}_c^b$ is the centroid of face f_b, and $\mathbf{n}_b$ is the normal. When involving multiple faces, the detection strategy is same with vertex merging.

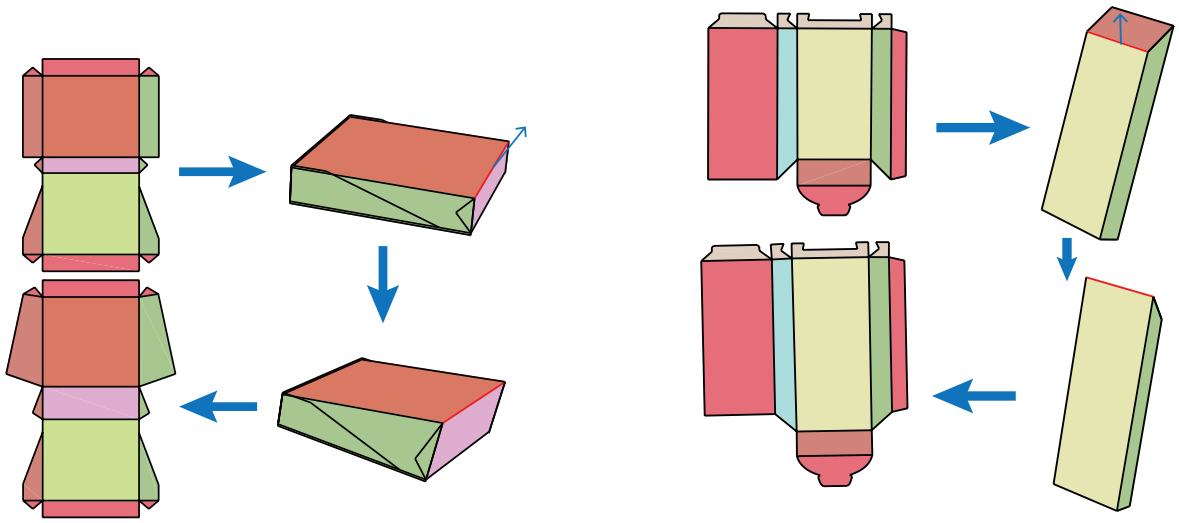
5.2 2D Layout Refinement from 3D Editing

While it is significantly efficient to generate a 3D carton using our approach from an existing layout, our system allows users to creatively edit the current 3D carton model by moving an edge and simultaneously refine the 2D layout automatically according to the 3D editing.

When the 3D mesh changes, the new shape constraints are transferred to the 2D layouts by keeping face rigidity (Eq. 1). Since the layout remains in a 2D plane, the coplanarity is enforced for all the vertexes in the layout. The irrelevant vertexes that do not lie on the moving edge have to stay at their original locations, as described in Eq. 3. Figure 8 shows two examples of automatic layout refinement from 3D editing.

6 Results and Discussion

Based on the two-step approach of folding a carton model as well as the suggestive user interface, our system is natural and easy to use. We will show a series of results of cartons in various shapes. A user study was also conducted to examine the effectivenss of our suggestive system.



(a) (b)

Figure 8: Two examples of layout modification from 3D model editing. For each example, a box-like carton can be generated from the given 2D layout. Users can edit the shape of the 3D carton by dragging edges in our system. By enforce the shape rigidity constrains, our system automatically updates the 2D layout.

Table 1: Statistics on the number of edges N_{edge}, number of faces N_{face}, and the number of user interactions $N_{interaction}$ of the examples shown in this paper.

Examples	Fig.9(a)	Fig.9(b)	Fig.9(c)	Fig.9(d)	Fig.10	Fig.11	Fig.12(a)	Fig.12(b)	Fig.12(c)	Fig.12(d)	Fig.12(e)	Fig.12(f)
N_{edge}	49	62	46	45	54	67	40	43	42	38	48	30
N_{face}	13	13	11	13	14	19	11	13	13	13	12	11
$N_{interaction}$	0	0	0	0	3	9	1	4	1	3	3	3

6.1 Carton Results

For most cuboid cartons, their 3D models can be generated by folding each edge by $\pi/2$, as shown in Figure 4 and Figure 9. In the common design process nowadays, designers typically spend much time on manually creating the 3D carton models in 3D modeling softwares. Our suggestive interface significantly saves the designers' effort so that they can focus on the appearance designing of cartons.

For more complicated cartons, shape refinement is required. An entire process of creating a carton model from a 2D layout is illustrated in Figure 10. The final model can be generated by three clicks from the user to confirm vertex merging two times, and (xuejin:face pasting) in one time. Note that our system automatically detects these possible geometric editing options for users to select. Therefore, the user can get rid of the tedious work of selecting edges or vertexes and assigning precise length or positions to them. A more challenging example is shown in Figure 11. By initially folding each edge as $\pi/2$, our system firstly generates a cube inside the carton for the six square faces. Sometimes, our system fails to detect mergeable vertexes with a small threshold ϵ to merge closeby vertexes. However, the user could manually select two vertexes to merge. Then our systems automatically detects two symmetric vertexes that also can be merged. Finally, a hexagonal carton can be generated in a limited number of user interactions.

More carton models generated using our system with a few user interactions are shown in Figure 12. The statistics of the face number, edge number, and number of user interactions are listed in Table 1. We can see that only a few user interactions on confirming system suggestions are needed for a variety of shapes.

6.2 User Study

We conducted a user study to examine the productivity of our system in constructing the digital 3D mockups from 2D layouts, and the effectiveness of our system of guiding non-expert users to fabricate the physical mockups from 2D layouts. There are two experiments in our user study. In the first experiment, participants were asked to construct the digital 3D model from the same 2D layout using our system and a commercial software STUDIO, respectively, after a brief introduction. For each participant, ten examples were randomly selected from the thirty-four 2D design layouts and presented to the participant for folding using our sytem and Studio. A *two-alternative forced choices* design was used, with the participant asked to choose which of the two systems they prefer to use considering the operation simplicity and modeling efficiency. Furthermore, participants were asked to rate our system from one to five on the necessity of layout optimization, while five indicates it is very necessary.

In the second experiment, participants were separated equally into two groups, one group was asked to fold a paper sheet with printed 2D layouts into a 3D mockup with the guide video showing the folding sequence provided by our

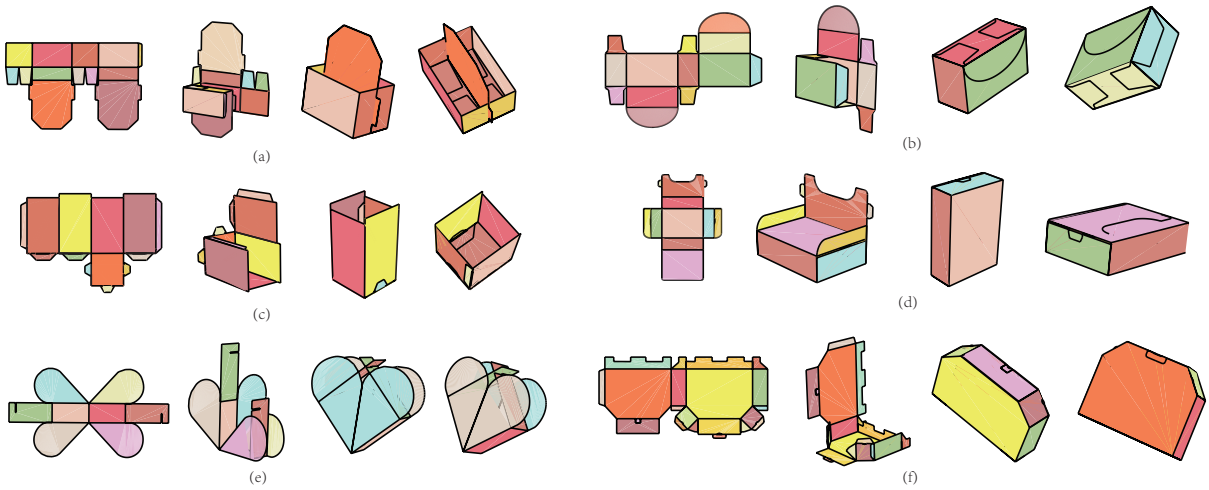


Figure 9: More carton examples that can be automatically generated. For each example, four subfigures are shown from left to right: the input 2D layout, an intermediate stage during folding, and the final 3D shape at two different views.

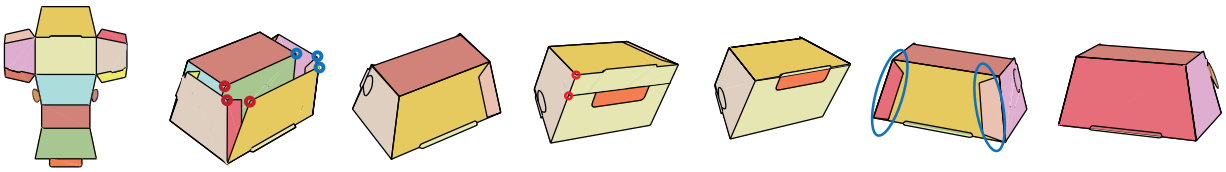


Figure 10: Structural layout is used to generate flat polymesh, and then initialize to a rough model, after two vertex merging confirmations and one face pasting confirmation, we can finally have its corresponding 3D model.

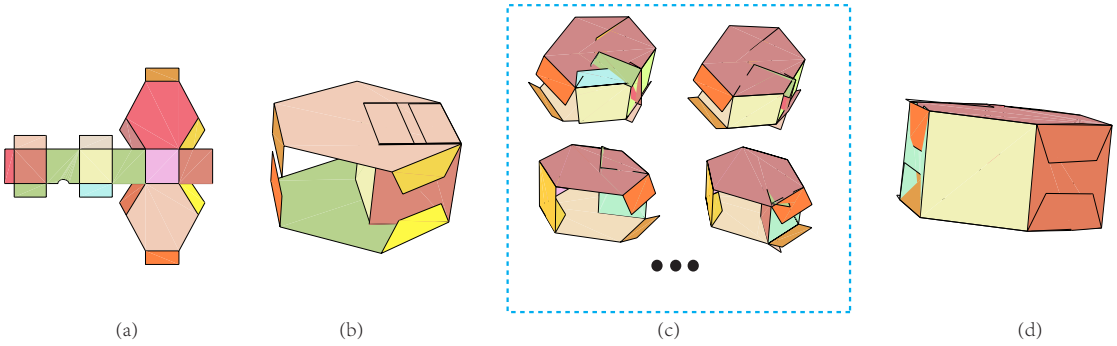


Figure 11: Given a layout as (a), our system can generate an initialized result as (b), and through almost ten steps including selecting merging vertexes and faces need to be coplane , users can reach the final model as (d).

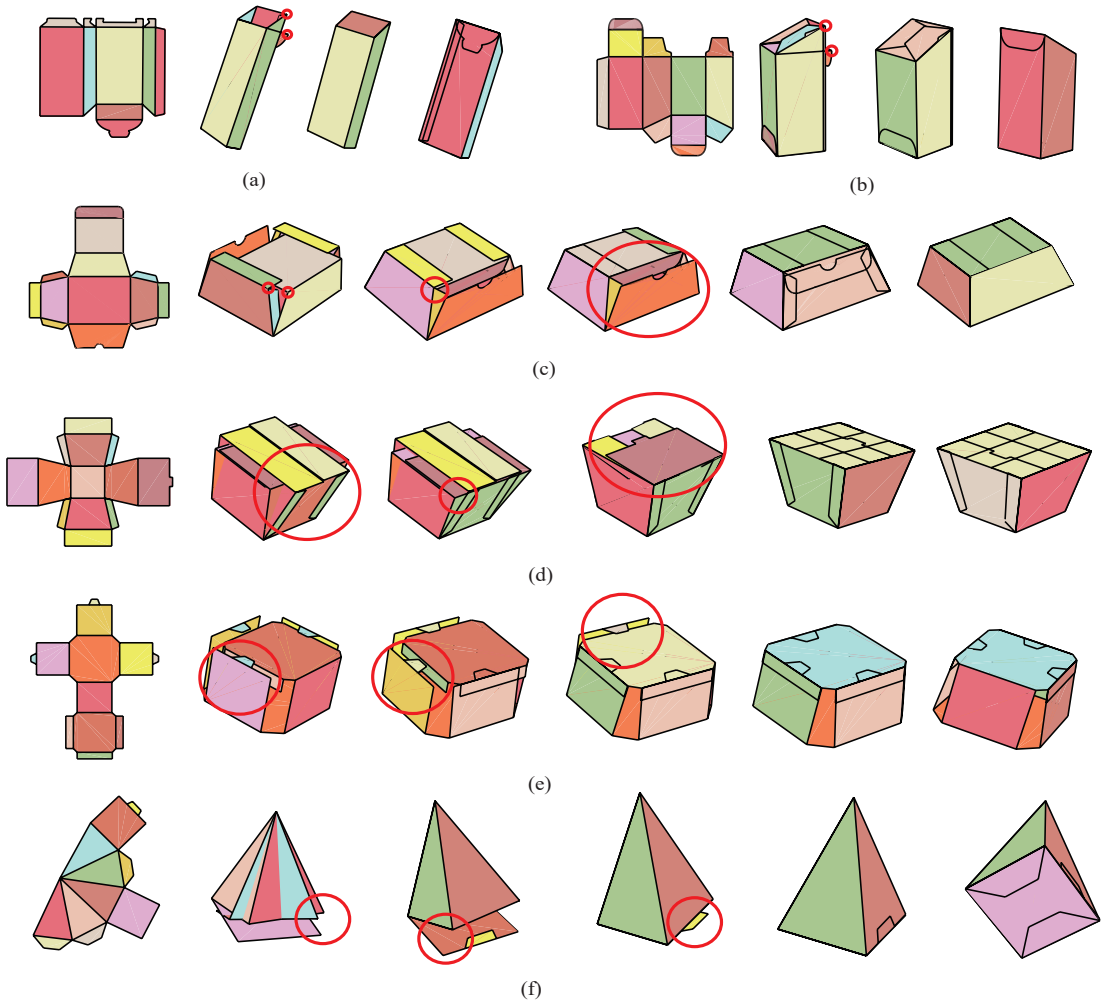
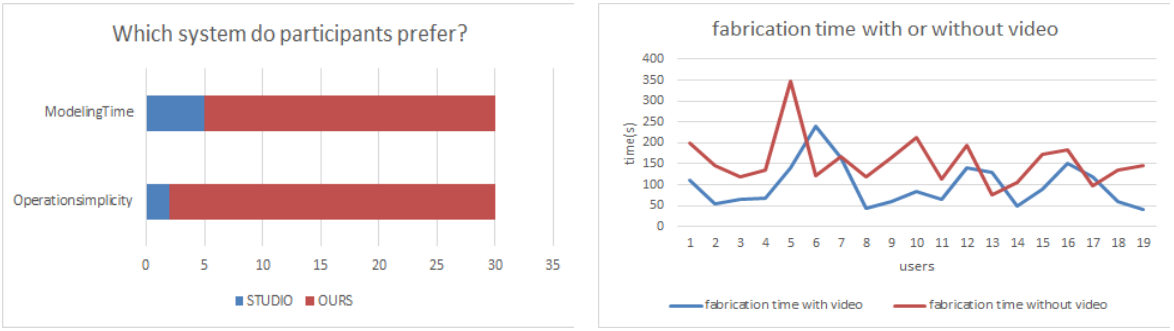
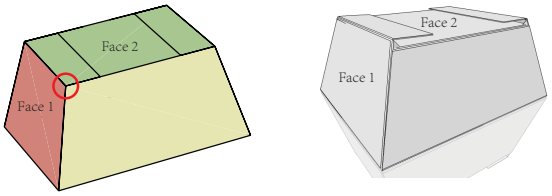


Figure 12: More results need refinement. Users can manipulate more complicated cartons. The first and second column in each sub-figure are the flat mesh and initialization result of cartons, and the last two column shows the model after interaction in different views. Circles marked in red means the part that need to be merged.



(a) Preference System (b) Fabrication Time

Figure 13: (a) shows the collection of the former two answers related to the preference based on modeling time and operation simplicity. (b) shows the fabrication time with or without video.



(a) (b)

Figure 14: Different model constructed by our system (a) and STUDIO (b).

system. Another group was asked to make a carton without the guide video. Each participant was assigned (xuejin:how many examples?). The fabrication time was recorded for two groups. Our goal was to test the following hypotheses:

- **Hypo1:** Our system needs less time and effort to construct a digital 3D mockup than the traditional software.
- **Hypo2:** Our system provides more novel and practical functions to generate diverse layouts.
- **Hypo3:** Our system is effective for guiding non-expert users to fabricate complex cartons from 2D layouts.

We consider our three hypotheses in turn. With respect to **Hypo1**, we collect the answers of former two questions from 30 participants, all over eighteen years of age. Both male and female participants were included, and none had any computer graphics background.??? (xuejin:Describe the background of participants.) The result is shown in Figure 13(a). The chart shows that 96% of participants prefer our system considering the operation simplicity, and 96% of participants prefer our system considering the modeling time. We also performed a pared-samples (xuejin:what does this mean?) t-test at level $\alpha = 0.05$ to compare the preference significance. The test shows our system is significantly preferred by participants. (xuejin:(?How to show t-test result?)) Most participants vote for our system mainly because it takes much less time to make a 3D model using our system. Take Figure 12(e) for example, participants usually spent much time on adjusting the folding angles of the creases, while only three clicks are required in our system. On the other hand, one comment from the participants who prefer STUDIO is that the operation needed to be learned in STUDIO is just selecting a folding line and assigning angles, while our system requires them to learn more operations to construct the digital 3D model.

With respect to **Hypo2**, twenty-four participants gave the highest score to our layout optimization function. In addition to explore the diversity of 2D layouts, it also can adjust the imprecise faces on the 2D layout to reach an ideal model by construction. Figure 14 shows the final model constructed by our system and STUDIO. As we can see, (xuejin:Panel 1) is higher than $Panel_2$ in the digital model constructed by STUDIO, because of the 2D layout is not that precise. In comparison, our system is able to correct these design error by merging the 3D vertexes in different panels circled in red and optimizing the 2D layout simultaneously.

Considering **Hypo3**, 38 participants (xuejin:But only 19 users in Fig 13) were asked to fabricate (xuejin:a randomly selected?) 2D layout into a physical mockup. The comparison of the fabrication times with and without our guide video is shown in Figure 13(b). As we can see, most participants who did not watch the guide video spent more time to folding a carton than the other group. (xuejin:Explain the other three special case.) We also performed an independent-samples t test at level $\alpha = 0.05$ to compare the fabrication time, and the result shows the guide video has significant effectiveness on the fabrication of complex cartons.

Limitations Shape optimization can be used in most cases through interaction, however, there are still some failure cases that we can not deal with. Figure 15 shows a case that failed to generate 3D model by optimization. The faces circled in red should be merged together to produce a strong enforced face, while our system expanded the face away. Moreover, we cannot deal with the case that if we want to layout down the back red face and keep the bottom lines of green side faces together. The carton layouts whose face graph has a loop will also be failed to construct corresponding model such as milk cartons.

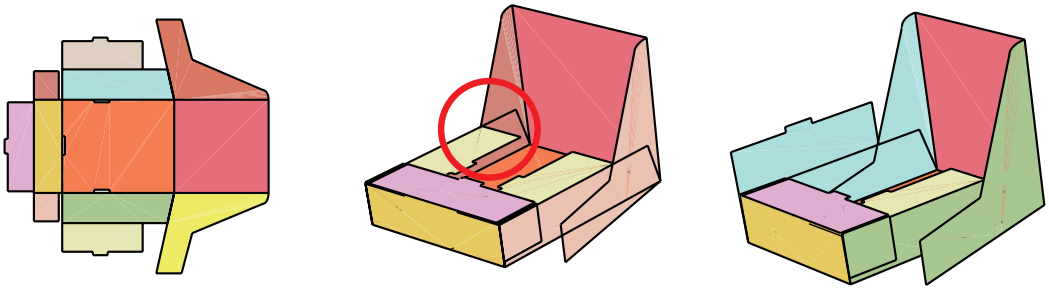


Figure 15: A failure case that shape optimization can not handle with.

7 Conclusion and Future Work

In this paper, we present an interactive modeling system to construct a 3D carton model from a 2D expanded layout. Based on the automatically folded rough model, a series of shape refinement suggestions are provided for smartly optimizing and editing the carton model, which assists users on producing the desired model efficiently and productively. Our current system could be further improved in many directions. More intelligent editing tools can be implemented. For example, when the user moves a vertex, the vertex translation can be propagated to other symmetric vertexes. We also would like to improve our automatic folding part with a more comprehensive optimization formulation considering both shape closure and stability. Moreover, it is desired by designers to support appearance editing in our system.

References

[1] T. Biedl, A. Lubiw, and J. Sun. When can a net fold to a polyhedron? *Comput. Geom. Theory Appl.*, 31(3):207–218, June 2005.

[2] T. C. Biedl and B. GenÃ§. When can a graph form an orthogonal polyhedron? pages 100–102, 2004.

[3] M. Botsch, M. Pauly, M. Gross, and L. Kobbelt. Primo: Coupled prisms for intuitive surface modeling. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, SGP '06, pages 11–20, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[4] S. Bouaziz, M. Deuss, Y. Schwartzburg, T. Weise, and M. Pauly. Shape-up: Shaping discrete geometry with projections. *Comput. Graph. Forum*, 31(5):1657–1667, Aug. 2012.

[5] L. Cao, J. Liu, and X. Tang. 3d object reconstruction from a single 2d line drawing without hidden lines. In *Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1 - Volume 01*, ICCV '05, pages 272–277, Washington, DC, USA, 2005. IEEE Computer Society.

[6] Y. Chen, R. Peng, and Z. You. Origami of thick panels. *Science*, 349(6246):396–400, 2015.

[7] B. Deng, S. Bouaziz, M. Deuss, A. Kaspar, Y. Schwartzburg, and M. Pauly. Interactive design exploration for constrained meshes. *Computer-Aided Design*, 61:13 – 23, 2015.

[8] ESKO. Studio. https://www.esko.com/en/products/studio.

[9] T. Igarashi, T. Moscovich, and J. F. Hughes. As-rigid-as-possible shape manipulation. In *ACM SIGGRAPH 2005 Papers*, SIGGRAPH '05, pages 1134–1141, New York, NY, USA, 2005. ACM.

[10] T. Kanade. A theory of origami world. *Artificial Intelligence*, 13(3):279 – 311, 1980.

[11] L. Kavraki, P. Svestka, J. Latombe, and M. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. Technical report, Stanford, CA, USA, 1994.

[12] M. Kilian, S. Flöry, Z. Chen, N. J. Mitra, A. Sheffer, and H. Pottmann. Curved folding. *ACM Trans. Graph.*, 27(3):75:1–75:9, Aug. 2008.

[13] M. Kilian, A. Monszpart, and N. J. Mitra. String actuated curved folded surfaces. *ACM Trans. Graph.*, 36(3):25:1–25:13, May 2017.

[14] N. Kishi and Y. Fujii. Origami, folding paper over the web. In *Proceedings of the Third Asian Pacific Computer and Human Interaction*, APCHI '98, pages 337–, Washington, DC, USA, 1998. IEEE Computer Society.

[15] J. S. Ku and E. D. Demaine. Folding Flat Crease Patterns with Thick Materials. *ArXiv e-prints*, Jan. 2016.

[16] Y. G. Leclerc and M. A. Fischler. An optimization-based approach to the interpretation of single line drawings as 3d wire frames. *International Journal of Computer Vision*, 9(2):113–136, 1992.

[17] J. Liu, Y. Chen, and X. Tang. Decomposition of complex line drawings with hidden lines for 3d planar-faced manifold object reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33:3–15, 2010.

[18] A. Lubiw. When can a polygon fold to a polytope? *Dept.comput.sci.smith College*, 1996.

[19] T. Marill. Emulating the human interpretation of line-drawings as three-dimensional objects. *Int. J. Comput. Vision*, 6(2):147–161, June 1991.

[20] T. Marill. Why do we see three-dimensional objects? 1992.

[21] G. Mullineux and J. Matthews. Constraint-based simulation of carton folding operations. *Comput. Aided Des.*, 42(3):257–265, Mar. 2010.

[22] R. Nimnual and S. Suksakulchai. Virtual reality for packaging folding practice. In *International Conference on Control, Automation and Systems*, pages 1011–1014, 2007.

[23] J. O'Rourke. Folding and unfolding in computational geometry. In *Revised Papers from the Japanese Conference on Discrete and Computational Geometry*, JCDCG '98, pages 258–266, London, UK, UK, 2000. Springer-Verlag.

[24] R. Poranne, E. Ovreiu, and C. Gotsman. Interactive planarization and optimization of 3d meshes. *Computer Graphics Forum*, 32(1):152–163, 2013.

[25] K. Shoji, K. Kato, and F. Toyama. 3-d interpretation of single line drawings based on entropy minimization principle. 2, 2001.

[26] J. Solomon, E. Vouga, M. Wardetzky, and E. Grinspun. Flexible developable surfaces. *Comput. Graph. Forum*, 31(5):1567–1576, Aug. 2012.

[27] G. Song and N. M. Amato. A motion planning approach to folding: From paper craft to protein folding. Technical report, College Station, TX, USA, 2000.

[28] T. Tachi. Simulation of rigid origami. 2009.

[29] T. Tachi. Geometric considerations for the design of rigid origami structures. In *Proceedings of the International Association for Shell and Spatial Structures Symposium*, Shanghai, China, 2010.

[30] T. Tachi. Rigid-foldable thick origami. 2011.

[31] C. Tang, X. Sun, A. Gomes, J. Wallner, and H. Pottmann. Form-finding with polyhedral meshes made simple. *ACM Trans. Graph.*, 33(4):70:1–70:9, July 2014.

[32] J. M. Thiel. Interactive manipulation of virtual folded paper. Master's thesis, UBC, 1998.

[33] C. Zou, H. Yang, and J. Liu. Separation of line drawings based on split faces for 3d object reconstruction. *2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 00:692–699, 2014.