Simple Flat Origami Exploration System with Random Folds

Naoya TSURUTA, Jun MITANI, Yoshihiro KANAMORI and Yukio FUKUI

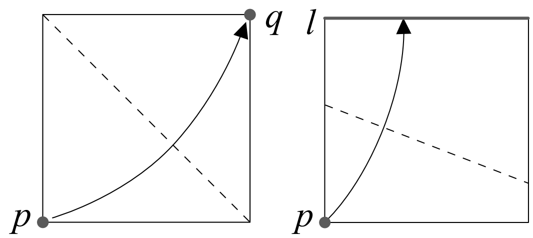
1. Introduction

Recent origami design techniques have made possible the design of realistic models. A modern realistic model with complex folds can represent the feature of an object in detail. The circle/river method [**Lan96, Meg94**] is a powerful tool for designing such complex origami models. This algorithm calculates the crease pattern to fold a tree structure that represents the skeleton of the target object. Another algorithm proposed by Tachi [**Tac09**] gives the crease pattern to fold an arbitrary 3D shape of a topological disk. However, there are no practical design tools suitable for simple origami models, which are made with a small number of folds. Existing design algorithms usually result in a complicated crease pattern. It seems that simple origami models are designed by a heuristic approach.

We proposed an enumeration based approach for exploring simple origami models [**TMKF12**]. By limiting the folding operations and the number of folds, $136,284$ different folded shapes were obtained with up to four times folds. In order to find, from a set of obtained pieces, the pieces that look like the shape of something, the pieces that are similar to an input polygon can be extracted by comparing the contour of folded pieces and the input polygon. Although this enumeration based approach can discover computer-generated origami models, the folding operations and the number of folds are strictly limited due to combinatorial explosion.

To address this problem, we have proposed an interactive system for exploring simple origami models by random generation of folded pieces [**TMKF13**]. The key idea is to assist "the action of labeling" ("*Mitate*" in Japanese). The action of labeling in origami is to recognize a folded shape as another object (such as animals, insects, and flowers) because of its color and physical appearance. The system generates origami pieces with several folds automatically, and displays them, so that the user can focus on the labeling action. Every time the user requests another set of pieces, different shapes appear, because the generation process includes randomness. We could discover several origami models. However, the generation algorithm in this prototype system cannot accommodate particular types of folds that have one degree of freedom (DOF) and there was no discussion about discovered models.

This paper proposes an improved generation method. The previous system used folds both with and without landmark points. A fold with landmark points refers



(A) (B)

FIGURE 1. Examples of the folds without and with DOF, respectively: (a) a fold places a corner to another corner; (b) a fold places a corner onto an edge of the paper

to a fold such as the corner-to-corner fold. A fold without landmark points refers to a fold placed on a meaningless location. We add the fold that has one DOF, such as a corner-to-edge fold. A corner-to-edge fold uses landmarks, but the location has one DOF. This new generation method gives various folded pieces that do not appear in the previous system. We also provide another generation method that considers symmetry. The resultant folded shapes differ greatly and look similar to human designed models. Finally, we discuss the models discovered by our system. We introduce three characteristic groups that are easily recongnized.

2. Random Generation of Simple Folded Pieces

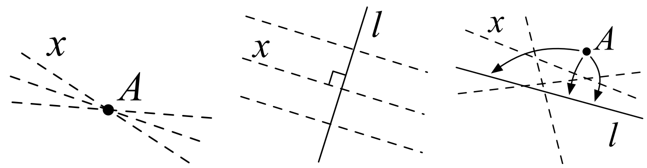
To generate a folded piece automatically, we randomly determine a fold, calculate the folded shape, and repeat this process a given number of times. A fold is composed of three parameters; the location of the fold, the folding direction (mountain or valley), and the number of layers to be folded. The problem is how to determine these parameters. Determining them completely at random will not generate commonly used folds, such as the corner-to-corner fold. Thus, we limit these parameters to generate meaningful folded pieces as if made by a human.

2.1. Location of Fold. The location of a fold is described by a combination of points and lines. Initially, an unfolded origami paper has four corner points and four edges. We randomly choose a required number of landmarks. For example, we pick two points at random in the case of the fold that places a point onto another point.

Our previous system [**TMKF13**] used the 6th Huzita-Justin axiom [**Huz91, Jus91**] and the random sliding of a fold. All the creases were obtained by the 6th axiom that places point p_1 onto line l_1 and point p_2 onto line l_2. The obtained fold was slid in a random direction in the last n-steps. This random sliding simulates $judgementfold$ ($Gurai - ori$ in Japanese), which is a folding operation that does not use any references. This folding often appears in later folding steps in the artists' model. Although we could find new origami models (shown in Section 5), this method cannot deal with a fold that has one DOF, such as the corner-to-edge fold. Figure 1 shows examples of a fold with no DOF and a fold with one DOF.

Instead of randomly sliding the folds, we add the following three operations that have one DOF as shown in Figure 2:

S_1**:** Fold a line x which passes through a given point A.



(A) (B) (C)

FIGURE 2. Folds with one DOF: (a) a fold that passes through a point; (b) a fold that is perpendicular to a line; (c) a fold that places a point onto a line

S_2**:** Fold a line x which is perpendicular to a given line l.

S_3**:** Fold a line x which places a given point A onto a given line l.

These operations are three of the five one-fold *alignments* of points and lines described by Alperin and Lang [**AL09**]. Although any line in the plane has three degrees of freedom to be specified, these use only one DOF. Hence, we randomly determine the remaining parameter when we use these operations. The remaining parameter is an angle in the case of S_1 and a location on an edge in the cases of S_2 and S_3. Note that several Huzita-Justin axioms may have two or more solutions, but we assume that the folds have no DOF because the solutions are discrete.

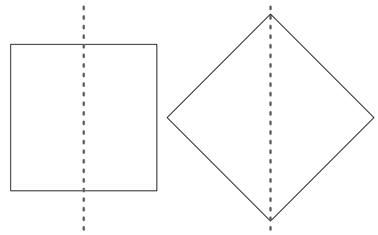
2.2. Folding Direction and the Number of Layers to be Folded. The folding techniques used in our system are simple valley folds and mountain folds. We randomly choose one of these.

After a location of the fold and a folding direction have been determined, we determine the number of layers to be folded. First, we choose a number from 1 to m at random. m is the total number of layers at the current state. Then, we try to fold m layers from the top in the case of valley folds. There are cases where we cannot fold m layers together when the fold intersects with existing creases, because faces adjacent to the existing creases must be folded at the same time. In such cases, we fold them together with the next layer and repeat this until we obtain a valid shape. Whether the fold is valid or not can be checked by testing the flat foldability for the inner vertices. The resultant folded shape is globally flat-foldable as long as we fold the paper from the top-most layer. We perform the same process from the bottom-most layer in the case of a mountain fold.

3. Random Generation of Symmetric Pieces

Here we describe another generation method that considers symmetry. This generation method is an extension of the simple folded piece generation we described above. We add another parameter, that is, the type of the symmetry axis. The symmetry axis has two types: book and diagonal (Figure 3). We randomly choose one of these at the very beginning of the generation process.

We use multiple folds or the squash fold to make symmetric shapes in one folding step. Both folding operations are calculated based on a single fold. The determination of which folding operation to use is depends on a positional relationship between a fold and symmetry axis. This relation can be categorized according to whether they intersect. Figure 4 shows a case where they do not intersect. The figure shows the right half of the square paper. If the flaps folded by an obtained



(A) Book (B) Diagonal

FIGURE 3. Two types of symmetry

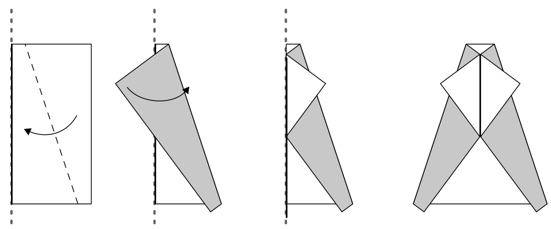


FIGURE 4. An entire folding operation that has two simple folds. The folds do not intersect the symmetry axis.

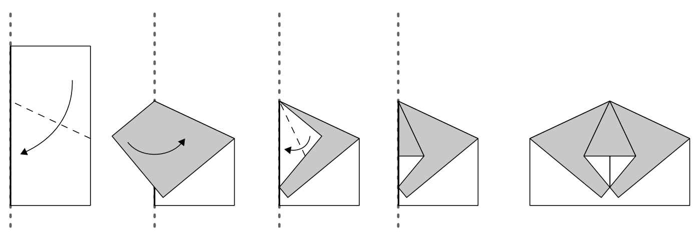


FIGURE 5. An entire folding operation becomes squash fold when the fold intersects with the symmetry axis.

fold intersect the symmetry axis, the flaps are folded again by the axis. Thus the entire folding operation has at most two simple folds.

Figure 5 shows a case of intersection. The flaps folded by an obtained fold intersect the symmetry axis, except for the case of the fold that is perpendicular to the axis. The folded part is folded back by the axis, and then, folded back again by the bisector of the angle between the axis and its edge so that the edge lies on the axis. The entire folding operation becomes a squash fold. If the fold is perpendicular to the axis, the entire folding becomes a simple horizontal fold.

The folded shape keeps symmetry as long as we use the above two folding operations. Note that these are not the only way to maintain the folded shapes' symmetry.

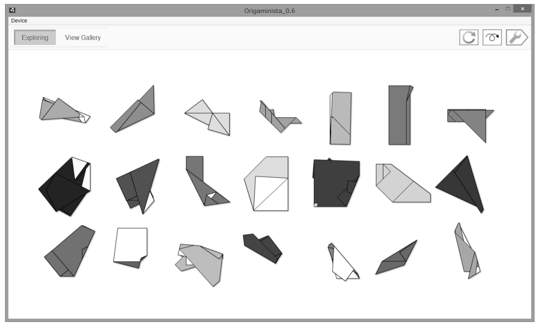


Figure 6. Interface of our system.

4. Our Proposed System

Our system generates dozens of folded pieces and displays them simultaneously. Figure 6 shows the interface of the system. The user chooses one of the displayed pieces if it looks like a desirable shape such as an animal. The system generates different shapes every time the user requests another set of pieces. The user can perform the following operations:

- Change the number of folds
- Request another set of pieces
- Rotate, turn over, or change the color of paper
- Label the selected piece and see its diagram
- Register the labeled piece to an online database server
- Set the initial state of the generation process

The last user operation changes the initial state of the generation process. The user can choose the initial state from the diagram. In other words, the first n steps can be fixed by the user.

5. Results

We implemented our system as a Flash application with Action Script. The online origami model database is implemented by MySQL and PHP. Our experiment revealed that a folded piece to be labeled appears after between 10 and 20 retrials on average. The computation time increases exponentially with the number of folds because the number of layers is approximately doubled by a single mountain fold or valley fold. Our system runs in real time on a PC with Intel Core i7 2.6 GHz CPU with the settings of six folds and twenty pieces. Figure 7 is an example model that we found.

We have also tried to find pieces that resemble existing origami models designed by origami artists. Figure 8 shows a well-known simple origami model "2 fold Santa" designed by Paula Versnick [**Ver**], an origami piece generated by our new generation method that includes folds with one DOF, and an origami piece generated by previous method using random sliding. The first fold of Paula's Santa is made by placing a corner onto nearly the middle point of the edge. This fold has one DOF. The folds with one DOF makes the folded shape neatly and easier to fold in comparison with those that have two DOF. Our new generation method generates more attractive origami pieces.

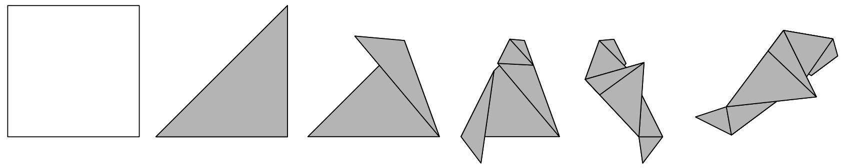
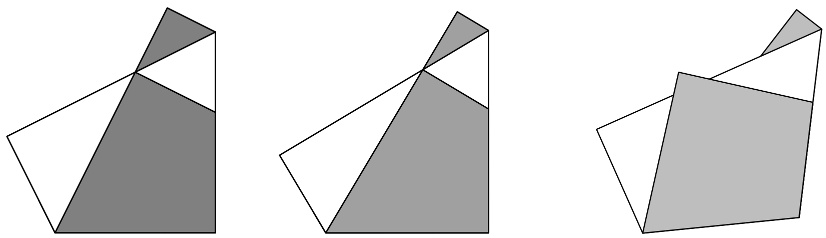


FIGURE 7. Auto-generated diagram of manatee.



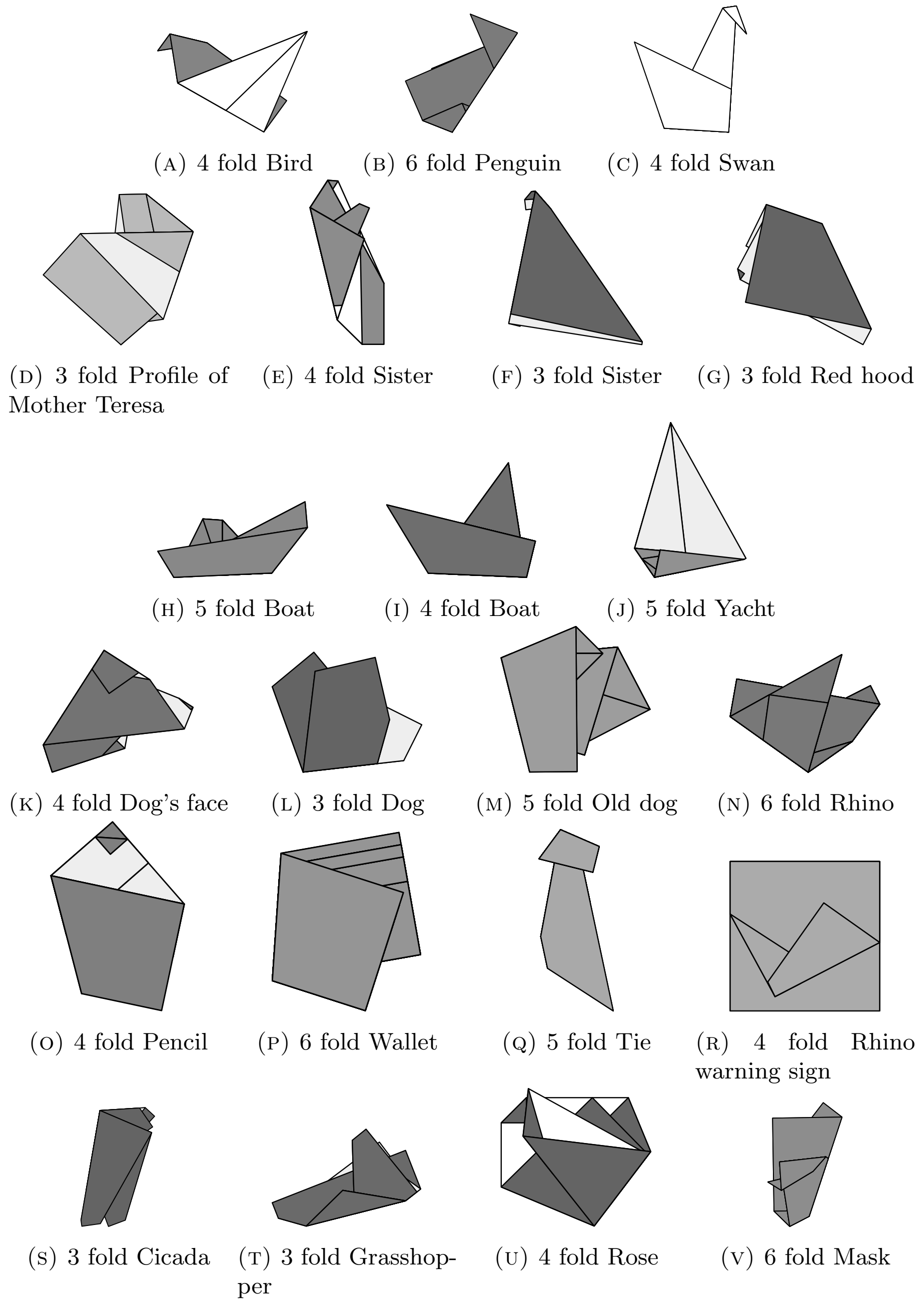
(A) 2 fold santa (B) With one DOF fold (C) Without one DOF fold

FIGURE 8. Comparison of Paula Versnick's 2 fold santa [**Ver**] and generated pieces with and without one DOF fold.

Figure 9 shows models stored in the online database. These models were generated by the previous generation method we briefly discussed in Section 2.1. We found that there were similar models, and have extracted three interesting groups. The first row in Figure 9 shows there are three bird models. The paper is folded in half into a triangle in the first step, and the corner becomes the beak of a bird. The second group is hooded persons and four models are included, as seen in the second row. The inside-out representation is used in these models. Inside-out is a technique that uses both sides of the origami paper and is often used for penguins, pandas, and texture of zebras. This technique is useful for both simple and complex origami. In simple origami, the inside-out technique enables a wide range of expression with a smaller number of folds. The third group contains two boats and a yacht. Boats and yachts are very typical objects in simple origami. We often see these kinds of models in origami books for children. The inside-out representation is also used in the yacht model. We could not find any characteristics for the models in the fourth or later lines. In other words, these models are rare and hard to discover.

We held a workshop for four small children aged from 5 to 7 years. After 10 minutes of instruction, each child could find about 7 models in 40 minutes. Figure 10 shows four of the discovered models. Our exploring system was enjoyable for the small children. However, several steps in the auto-generated diagram are difficult to understand due to the lack of annotation and of distortions that indicate the relationships between the layers.

Figure 11 shows four examples of symmetric models generated by the method we mentioned in Section 3. These models were obtained by two times symmetric folding. Note that a single symmetric folding may contain multiple folds. These models seem as if they could have been designed by humans. Considering symmetry is effective for generating well-arranged folded pieces. Adding other rules will be



(A) 4 fold Bird (B) 6 fold Penguin (C) 4 fold Swan

(D) 3 fold Profile of Mother Teresa (E) 4 fold Sister (F) 3 fold Sister (G) 3 fold Red hood

(H) 5 fold Boat (I) 4 fold Boat (J) 5 fold Yacht

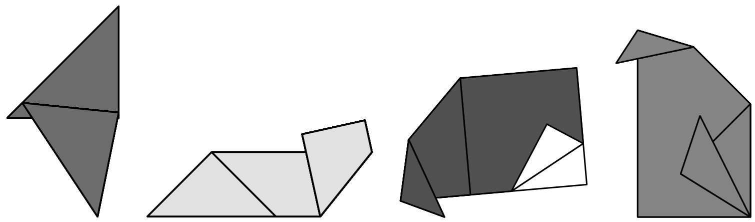
(K) 4 fold Dog's face (L) 3 fold Dog (M) 5 fold Old dog (N) 6 fold Rhino

(O) 4 fold Pencil (P) 6 fold Wallet (Q) 5 fold Tie (R) 4 fold Rhino warning sign

(S) 3 fold Cicada (T) 3 fold Grasshopper (U) 4 fold Rose (V) 6 fold Mask

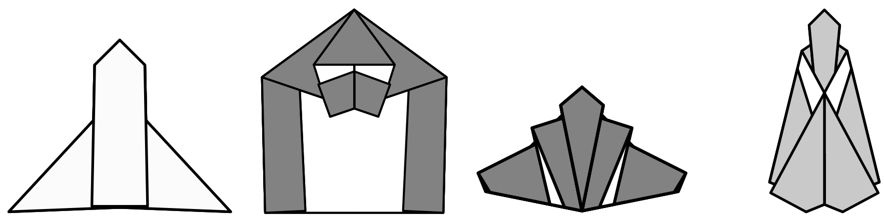
FIGURE 9. Example models stored in the online database

an interesting challenge; for example, considering rotation symmetry may generate a lot of flowers.



(A) Bird (B) Otter (C) Whale (D) Penguin

FIGURE 10. Example models discovered by small children



(A) Rocket (B) Circus tent (C) Japanese imperial palace's dolls (D) Japanese court lady wearing a ceremonial kimono

FIGURE 11. Examples of obtained by the generation of pieces with symmetry

6. Conclusion

We have proposed an interactive exploration system for simple flat origami. Our system generates folded pieces with a specified number of folds, and displays them. The user can discover new computer generated origami models by labeling them, without folding a physical sheet of paper. Folding techniques used in our system are limited to the simple valley fold and mountain fold, or the squash fold. Adding other folding techniques such as reverse folds would increase the variety of folded shapes.

Folds that have one DOF have now been added. These folds enable neat shapes than the previous system. Note that it is necessary to perform a quantitative evaluation to determine which method generates more meaningful and recognizable pieces.

The action of labeling in origami may be an interesting research topic. Our results showed that there are characteristic shapes which are easy to recognize. Avoiding the folds used to generate those shapes will lead to the discovery of different models. In our system, the labeling depends on a person. If we obtain a large number of labeled origami pieces, automatic labeling of simple origami may be possible by artificial intelligence.

References

[AL09] Roger C. Alperin and Robert J. Lang, *One-, two-, and multi-fold origami axioms*, Origami 4: Fourth International Meeting of Origami Science, Mathematics, and Education (Robert J. Lang, ed.), A K Peters Ltd, 2009, pp. 371–393.

[Huz91] Humiaki Huzita, *Axiomatic development of origami geometry*, the First International Meeting of Origami Science and Technology (H. Huzita, ed.), Italy: Dipartimento di Fisica dell'Universita di Padova, 1991, pp. 143–159.

[Jus91] Jacques Justin, *Resolution par le pliage de l'equation du troisieme degre et applications geometriques*, the First International Meeting of Origami Science and Technology (H. Huzita, ed.), Italy: Dipartimento di Fisica dell'Universita di Padova, 1991, pp. 251–261.

[Lan96] Robert J. Lang, *A computational algorithm for origami design*, the Twelfth Annual Symposium on Computational Geometry, New York: ACM Press, 1996, pp. 98–105.

[Meg94] Toshiyuki Meguro, *Tobu kuwagatamushi-to ryoikienbunshiho (flying stag beetle and the circular area molecule method)*, Oru, Sojusha, Tokyo, 1994, pp. 92–95.

[Tac09] Tomohiro Tachi, *3d origami design based on tucking molecules*, Origami 4: Fourth International Meeting of Origami Science, Mathematics, and Education (Robert J. Lang, ed.), MA: A K Peters, 2009, pp. 259–272.

[TMKF12] Naoya Tsuruta, Jun Mitani, Yoshihiro Kanamori, and Yukio Fukui, *A computer aided design system for simple origami pieces based on the enumeration of folded shapes (in japanese)*, Science of Origami (Jun Maekawa, ed.), vol. 2, Koryo Miura, 2012, pp. 33–44.

[TMKF13] ______, *An exploring tool for simple flat origami based on random foldings*, International Conference on Simulation Technology (JSST2013), 2013.

[Ver] Paula Versnick, *Orihouse*.