Estimation of Folding Operation Using Silhouette of Origami

Yasuhiro Kinoshita † Toyohide Watanabe †

Abstract—**In this paper, we address a method for supporting origami folding process on origami. We focus on a folding operation with a view to understanding the state of origami. We estimate the folding operations by using one camera. The folding operation can be estimated by comparing origami shapes of before and after a folding operation. A silhouette model is introduced to grasp the correspondence of a state of an origami sheet to an applied folding operation. Comparing a silhouette with the shape of origami in a camera image, the folding operation can be estimated. Also, our method can distinguish the folding operation at high speed.**

Keywords: origami, camera image, feature points, silhouette model, estimation of folding operation

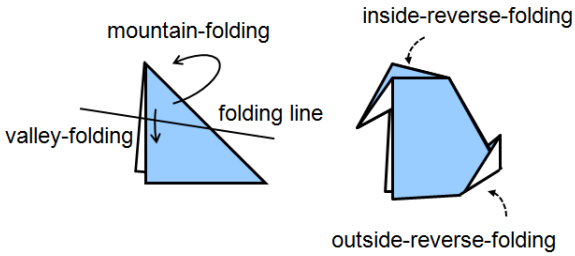

Figure 1: Folding operations

1 Introduction

In an origami work, generally the book called origami drill book has been often used. The drill book indicates the folding operation step by step with corresponding illustration. However, since the correlated folding operations are not completely expressed, it is not always easy for beginners to perform an origami work successfully on the way. To solve this problem, Kato, et al. proposed a method for understanding the folding process of an origami work by making 3D animations from origami drill books [1]. Although the animation of the folding process is easy to understand, there is a problem that the system cannot display a scene suitable to origami state of a user. Therefore, we propose an interactive folding operation support system which explains the folding operations clearly according to the procedural states of origami.

In order to support folding operations, it is necessary to recognize and maintain the state of origami. Moreover, it is necessary to recognize at high speed in order to construct an interactive system applicable to the operation of a user. There are several researches for recognizing the state of an origami sheet. Mitani, et al. proposed a method which uses a sheet printed with 2D bar-codes [2]. However, the recognition of the 2D bar-codes is time-consuming. Ju, et al. suggested a method which embeds radio tags into papers [3]. However, since this method requests a special device, it is not adoptable to develop a support system of origami. These methods are unsuitable to support user's folding operations. On the other hand, there are some researches about a data structure to express an origami [4, 5]. Additionally, a data structure for representing the state of origami was proposed by Miyazaki, et al. [6]. Many researches about origami use this data structure [7, 8]. The data structure can be updated by the folding operation. Therefore, if a folding operation which a user applied to origami is known, it is possible to judge what the state of origami is. We have studied for making our system understand an applied folding operation [9]. In this paper, we focus on the feature of origami, and propose a method for estimating folding operations rapidly. Our system uses only one camera to construct an experimental environment.

In the following sections, we first discuss our approach, and the outline of our system. Next, we describe the method of estimating folding operations from a series of camera images, and show experimental results. Finally, we affer a conclusion and our future work.

2 Approach

2.1 Characteristics of Origami

Origami deals with a two-dimensional object, and is bound by geometric constraint. An origami state is changed by each folding operation. In this paper, the origami of before applying a folding operation is called pre-origami and the origami of after applying the fold-

†Department of Systems and Social Informatics, Graduate School of Information Science, Nagoya University, Furocho, Chikusa-ku, Nagoya, 464-8603, Japan Phone: +81-52-789-2735 Fax: +81-52-789-3808 Email: {kinoshita, watanabe}@watanabe.ss.is.nagoya-u.ac.jp

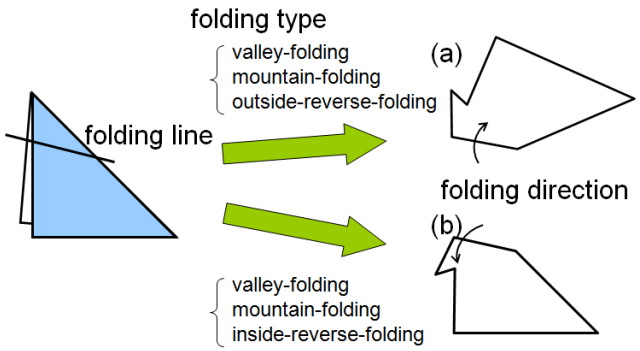

Figure 2: Different folding operations

ing operation is called post-origami. The change of an origami state depends on the pre-origami state and the folding operation. Therefore, we can understand the post-origami state from the pre-origami state and the folding operation.

A folding operation is represented by a location of a folding line, a folding direction and a folding type such as "valley-folding" and "mountain-folding". An example of a folding type is shown in Fig. 1. For instance, according to the folding line as illustrated in Fig. 2, there are two folding directions and four folding types: "valley-folding", "mountain-folding", "inside-reverse-folding" and "outside-reverse-folding". The origami shapes 2(a) and 2(b) are two kinds of shapes decided by folding direction.

There is a relationship between a folding operation and the change of the origami shape. Therefore, the folding operation can be estimated by comparing the shape of pre-origami with that of post-origami. Since the information about the origami shapes can be obtained from camera images, we estimate the folding operation from camera images. To this end, we introduce a silhouette model. By this model, we estimate the folding operation at high-speed.

2.2 Silhouette Model

Several methods for recognizing objects in a camera image by maintaining the shape of an object have been proposed [10]. Kameda, et al. proposed a method for recognizing human posture using 3D models which express parts of a human body [11]. In this method, one camera image is matched with a 3D model. However, in order to recognize a 3D object, there is a problem that computational cost is high. Therefore, in order to reduce computational cost, we handle an origami as a two-dimensional object.

A silhouette maintained in our silhouette model represents the shape of an origami as an undirected graph. In this graph, all edges constitute a loop, and the edges do not cross. A vertex has a vertex-ID and coordinates. An edge has an edge-ID and two vertex-IDs of both ends. When a folding operation is applied, the shape of an origami is changed, and a new silhouette is needed. Therefore, our system obtains the silhouette which has the same shape as the post-origami from the previous silhouette (called pre-silhouette). The shape of a silhouette in the initial state is maintained in advance because the original shape of an origami sheet is square.

Our system can estimate the location of origami in a camera image (called camera-origami) by matching the origami with the silhouette. For corresponding to them, we use the vertexes of the silhouette and feature points extracted from the camera image. Since there are only small number of vertexes and feature points, the number of comparison is decreased. Therefore, our method requires little computational cost for matching the silhouette and the camera-origami. From the result of estimation, feature points can be corresponded with the vertexes.

In order to estimate the folding operation, the correspondence between the pre-silhouette and the camera-origami after the folding operation is necessary. As a result of correspondence, our system can estimate the folding operation from the difference between the shapes of pre-silhouette and the camera-origami.

When a folding operation is applied and the shape of the origami changes, a new silhouette which is similar to the new shape of origami is needed. The silhouette is obtained by applying the folding operation to the pre-silhouette. Therefore, it is possible to estimate the arbitrary folding operation of a user.

3 System Outline

3.1 Precondition

Our system uses a fixed camera which looks down an origami and takes a series of camera images. The camera is set up to just above an origami. To make the image processing simple, we assume that a background is black and two sides of the origami are colored white and light blue.

We restrict the folding operations which a user can apply. The folding operation is represented by the location of a folding line, a folding direction, and a folding type. The folding line is included in the region of pre-origami, and becomes the edge that constitutes the shape of post-origami. This is because the folding line has to be the edge of the post-origami shape so that it can be acquired from a camera image after the folding operation. An example is shown in Fig. 3. Fig. 3(a) is an available folding operation. The folding line of the pre-origami becomes the edge of the post-origami shape. On the other

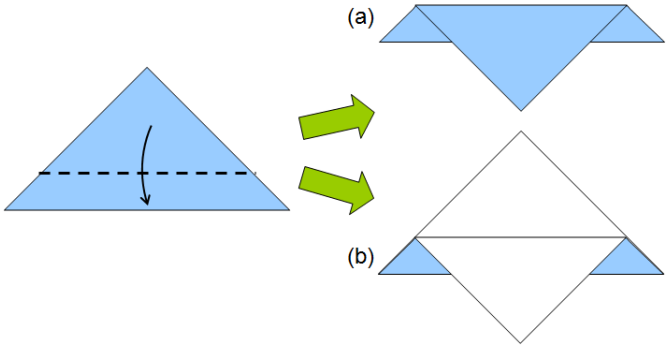

Figure 3: Two kinds of "valley-folding"

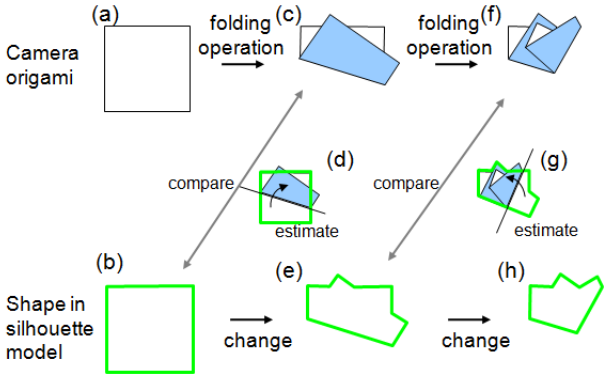

Figure 4: Processing flow

hand, since the folding line is contained inside the origami shape, as shown in Fig. 3(b), it is difficult to detect the folding line from the origami shape. Moreover, we deal with folding types, as "valley-folding", "inside-reverse-folding" and "outside-reverse-folding". Since "mountain-folding" is the same folding operation as "valley-folding" except the folding directions, only "valley-folding" is used.

The end of folding operation is pointed out by a user, because it is difficult to find out the end of the folding operation from camera images. In addition, a user notifies our system of the operations of folding back and inversion. This is because these operations cannot be estimated from a silhouette model. However, our system permits those folding operations because they do not need information from the camera image.

3.2 Processing Flow

Our system estimates the location of a camera-origami fundamentally. When a user notifies our system that a folding operation is applied, the folding operation is estimated. When making an origami work, it is necessary to estimate two or more folding operations. Therefore, the shape of the silhouette is changed at each folding operation as shown in Fig. 4. The processing flow illustrated in Fig. 4 is summarized as follows.

1. After changing the size of (b) according to (a), both of them are corresponded.
2. After user notifies the system of the end of a folding operation, (c) is compared with (b). Then a folding operation (d) is estimated.
3. A new silhouette (e) is obtained by changing (b) according to the estimated folding operation.
4. After the end of second folding operation, (f) is compared with (e). Then a folding operation (g) is estimated.
5. A new silhouette (h) is obtained.

4 Estimation of Folding Operation

4.1 Estimation of Origami Location

Our system estimates the location of a camera-origami by using a silhouette. As a result of the location estimation, the silhouette position moves to the location where the camera-origami is. However, when a folding operation is applied, the origami shape is different from the silhouette shape. In this case, the silhouette position moves to the location where the origami might have existed when the folding operation was not applied (as shown in (d) and (g) in Fig. 4). Next, we describe the procedure for estimating the location of a camera-origami.

First, our system extracts feature points from a camera image [12, 13] because feature points are used to correspond to a silhouette with a camera-origami. Vertexes of a camera-origami can be extracted mostly, although unnecessary points are also extracted from the boundary of a hand and the camera-origami. Therefore, feature points around the area of a hand are removed by skin detection. An extracted feature point also has a vertex-ID. If it does not correspond to a vertex of the silhouette, the vertex-ID becomes -1.

Second, our system tracks feature points. When a user folds an origami, the origami is rarely moved quickly. For this reason, feature points which represent the same vertex are very near in a series of camera images. Then, as a result of comparing the previous frame, the nearest point within a predefined threshold is assumed as the same point: our system succeeds to vertex-ID. In this method, it is possible to reduce the number of comparison between the silhouette and the camera-origami according to succeeded feature points.

Third, our system matches the silhouette with the camera-origami in order to estimate the location of the camera-origami. After selecting two vertexes and two feature points whose distances are the same as shown

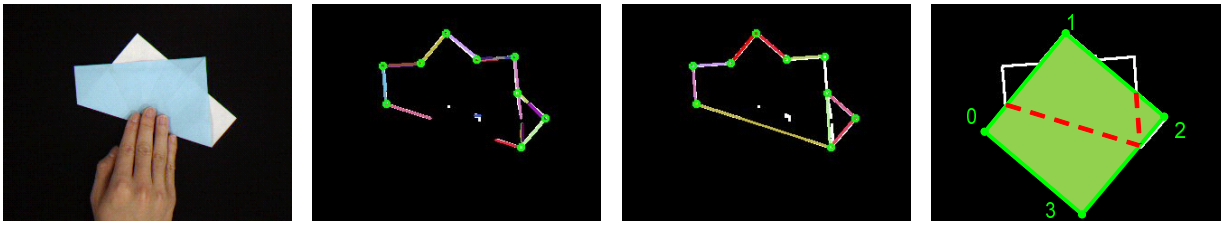

Figure 7: Estimation of a folding line

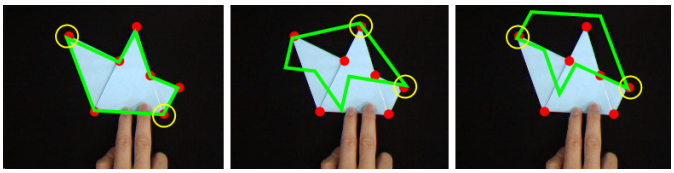

Figure 5: Example of superimposing the silhouette on the origami in a camera image

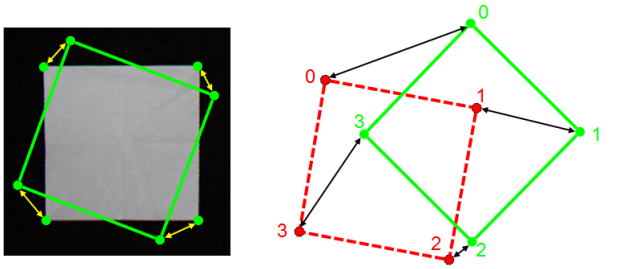

Figure 6: Matching degree

in Fig. 5, the silhouette position is moved so that they may overlap each other. Moreover, our system calculates the matching degree which describes the similarity between the silhouette and the camera-origami, and estimates that the origami exists in the location where the matching degree is the lowest. In matching, the number of comparison decreases by raising the priority of the pattern in which a vertex and a feature point that have the same vertex-ID overlap each other.

Finally, after the silhouette position is moved to the estimated location, labeling is carried out. A feature point near each vertex in the silhouette is assumed to be a vertex of the camera-origami and a vertex-ID is succeeded. In addition, as a result of labeling, feature points can be tracked at the next frame.

When the silhouette corresponds to the camera-origami correctly, the matching degree becomes lowest. To obtain such a degree, we focus on the gap between the silhouette and the camera-origami, and the moved distance of the silhouette position from the position in the previous frame. These are calculated from vertexes of the silhouette and feature points extracted from a camera image. The gap d_1 in Fig. 6(a) between the silhouette and the camera-origami is an average from each vertex of the silhouette to a feature point in the neighborhood, and is calculated like equation (1). C is a set of feature points which are extracted from a camera image, and c is a feature point included in C. The number of vertexes in the silhouette is represented as n, and each vertex is represented as v_i. Euclidean distance between v_1 and v_2 is expressed by $D(v_1, v_2)$. The gap d_2 in Fig. 6(b) between the silhouette of the current frame and the silhouette in the previous frame is an average of the moving distance of each vertex, and is calculated by equation (2). A moved vertex in the current silhouette is expressed by v, and a vertex of the silhouette in the previous frame is expressed by v'. Since an origami does not move quickly, it is considered that the origami exists in the near location from the origami in the previous frame. In addition, when the silhouette is moved to the right location, the gap often becomes smaller. Therefore, the matching degree M is defined in (3) as the production of d_1 and d_2.

$$d_1 = \frac{1}{n}\sum_{i=1}^{n} min(D(v_i, c)) \qquad c \in C \tag{1}$$

$$d_2 = \frac{1}{n}\sum_{i=1}^{n} D(v_i, v_i') \tag{2}$$

$$M = d_1 d_2 \tag{3}$$

4.2 Estimation of Folding Line

We explain the method for estimating the location of a folding line in Fig. 7. After the shape of an origami is extracted from a camera image as illustrated in Fig. 7(a) [14], straight lines are detected using Hough transform [15]. The result is shown in Fig. 7(b). Some of the detected lines are not the edges of a camera-origami because they are hidden by hands of a user. Therefore, if there is a segment of the straight line between two feature points, an edge can be detected as shown in Fig. 7(c), assuming

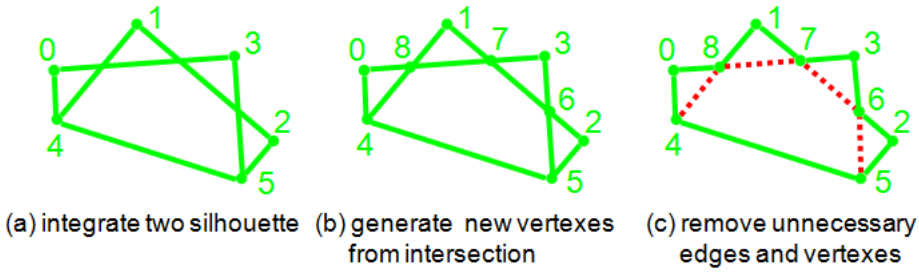

Figure 10: Generation of a new silhouette

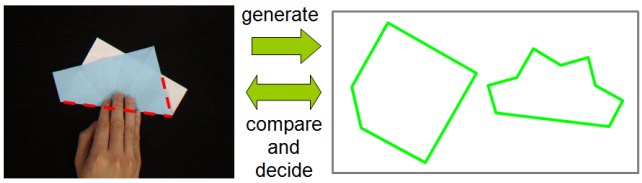

Figure 8: Decision of folding lines

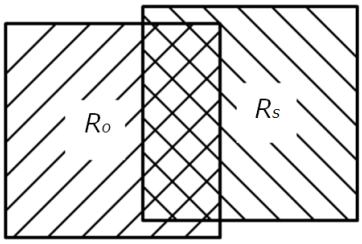

Figure 11: Calculation of overlapping degree

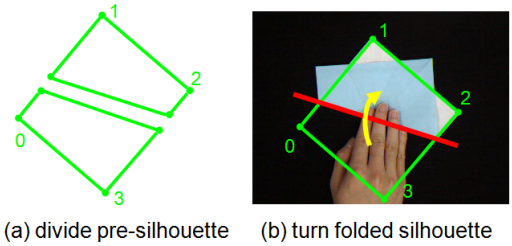

Figure 9: Decision of folding direction

there is an edge between the feature points. At this time, as shown in Fig. 7(c), not only the edge of the origami silhouette but also the border with the color of two sides of an origami might be extracted. As shown in Fig. 7(d), our system distinguishes whether the extracted straight line exists inside the pre-silhouette. In Fig. 7(d), the inside of the pre-silhouette is painted over with green, and edges which satisfy the condition of a folding line are displayed in dashed lines. In this method, there is the possibility that two or more folding line candidates are detected as shown in Fig. 7(d), but there is no problem.

4.3 Decision of Folding Operation

It is necessary to find the most suitable one of the detected folding lines, and decide the folding direction and the folding type. Therefore, our system compares the camera-origami with the silhouette shape that is obtained from each folding line. An example is shown in Fig. 8. The most suitable shape silhouette becomes a new one, and a folding line is determined.

A new silhouette is obtained from the pre-silhouette and one folding line. First, our system divides the pre-silhouette shape with a folding line as shown in Fig. 9(a). Second, the silhouette that represents the face which a user moved is turned around the folding line as shown in Fig. 9(b). The moved face can be judged from a camera image, and the folding direction is determined. Finally, two silhouettes are integrated as shown in Fig. 10, and a new silhouette is generated.

For determining a silhouette corresponding to the post-origami shape, our system uses overlapping degree which is the ratio that the origami region and the silhouette region have overlapped. The silhouette is moved as well as the way of estimating origami location. In Fig. 11, R_o is the origami region, and R_s is the moved silhouette region. Overlapping degree is defined in (4).

$$L = \frac{S_{R_o \cap R_s}}{S_{R_o \cup R_s}} \tag{4}$$

$S_{R_o \cap R_s}$ is the area of the region $R_o \cap R_s$ and $S_{R_o \cup R_s}$ is the area of the region $R_o \cup R_s$. Since the computational cost of calculating overlapping degree is large, our system uses d_1. The computational cost can greatly be reduced by calculating overlapping degree only when d_1 is smaller than a certain threshold value.

Although we previously described that overlapping degree is defined as (4), in effect, it is necessary to improve (4). Since a camera-origami is often occluded by hand,

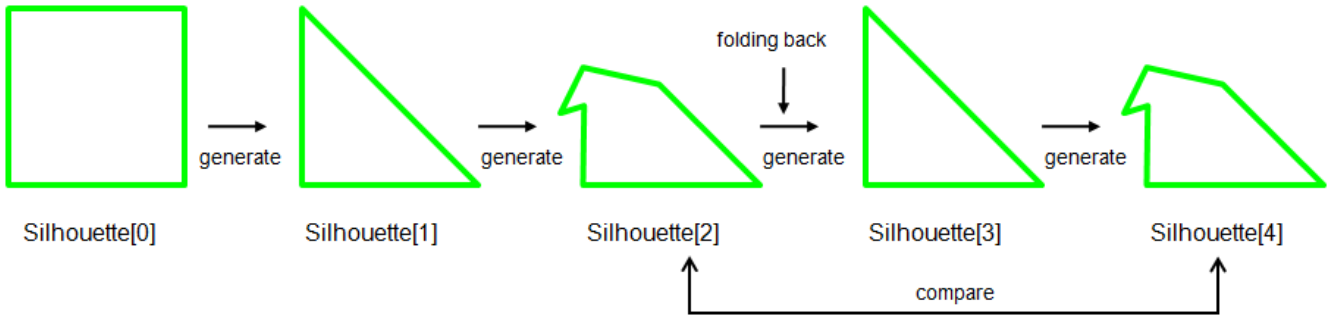

Figure 13: Shape transition in silhouette model

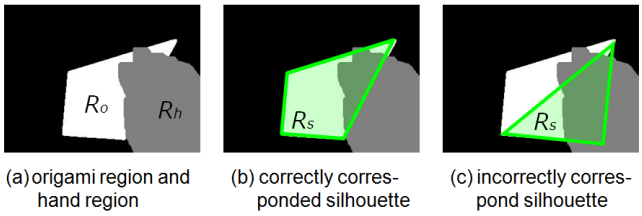

Figure 12: Correspondence between origami occluded by a hand and silhouettes

our system cannot calculate the correct overlapping degree. According to how to handle the hand region, the overlapping degree with the incorrect silhouette may become more than the degree with the correct silhouette. Therefore, we have to take occlusion into consideration. Occlusion is an important problem in the field of image recognition, and there are various researches about it [16]. Some of them use a stereo camera for solving this problem [17, 18]. In the researches of tracking a moving object in a camera image, an occluded object is recognized by using a trajectory [19, 20]. However, our system uses only one camera, and the camera-origami shape is changed: our system calculates overlapping degree by weighting the hand region. Therefore, overlapping degree is defined in (5).

$$L = \frac{S_{R_o \cap R_s} + S_{R_h \cap R_s} \times \alpha}{S_{R_o \cup R_s}} \quad (5)$$

R_h is the hand region detected by skin detection as shown in Fig. 12(a). Fig. 12(b) is a correct silhouette corresponded with the camera-origami, and Fig. 12(c) is an incorrect silhouette. Part of the hand region is included in the silhouette region in each figure. Since it is ambiguous whether the camera-origami exists in the region, the region is given with the weight α ($0 \leq \alpha \leq 1$). On the other hand, it is assumed that the camera-origami does not exist in the other part of hand region. In evaluation, α is set to 0.5.

After the folding line is decided, our system estimates the folding type. In order to estimate the folding type, our system uses the limitation that "inside-reverse-folding" and "outside-reverse-folding" are applied after folding mark has been left. The operation of leaving folding mark is input by the user. As a result, the folding type can be judged from the history of the silhouette shapes. Fig. 13 illustrates an example. The silhouette shape obtained by n-th operation is expressed as Silhouette[n]. When "inside-reverse-folding" or "outside-reverse-folding" is applied, Silhouette[n] and Silhouette[n-2] become similar by the limitation.

5 Evaluation Experiments

In order to show the effectiveness of our proposed method, we evaluated our prototype system. To show the implementability of the folding operation support system, accuracy and computing time in estimating the folding operation were evaluated.

5.1 Description of Experiments

In the experiment, we used the origami work called "gonbei-garasu". The folding process of "gonbei-garasu" consists of five folding operations which can be estimated as illustrated in Fig. 14. Folding operations other than folding back can be estimated. The folding types of *operation*1, *operation*2, *operation*3 and *operation*4 are "valley-folding", and the type of *operation*5 is "inside-reverse-folding". We made "gonbei-garasu" 20 times, and estimated the operations 100 times in total.

Experiment environment is as follows.

- CPU: Intel(R) Core(TM)2 Duo CPU E7400 2.80GHz
- Main Memory: 2GB
- Development Environment: Microsoft VisualC++, OpenCV [21]
- WEB Camera: ELECOM UCAM-E1D30MBK

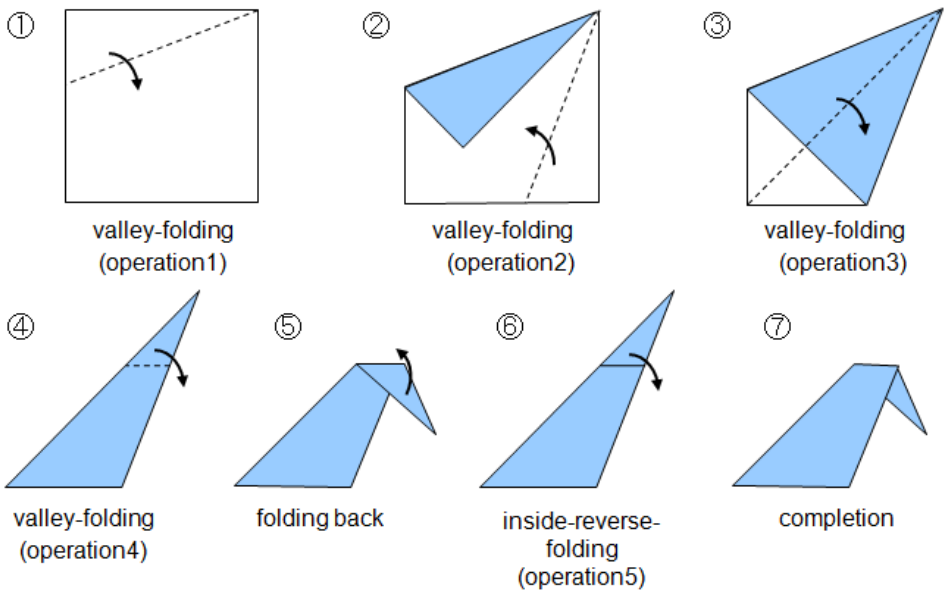

Figure 14: Folding process of "gonbei-garasu"

Table 1: Average of computation time and accuracy of estimation

	*operation*1	*operation*2	*operation*3	*operation*4	*operation*5
computation time (ms)	159	160	143	181	205
accuracy	1.00	1.00	1.00	1.00	0.95

We prepared a black board as a background, and operated an origami on the board. In our experiment, a camera is adjusted so that the camera view might not exceed the area of the board.

5.2 Experimental Results

Table 1 shows the average of computation time and accuracy for estimating each folding operation. Accuracy is calculated by dividing the number of times correctly estimated by all number of times.

From Table 1, it turns out that the average of computation time for estimating a folding operation is about 170ms. From this result, we can see that sufficient high-speed processing is realized for supporting folding operations. In addition, accuracy of estimating folding operation is 99.0% on the whole, and our system can start over even if estimation fails. Therefore, this shows our system is practical.

In some cases, our system failed in the estimation of *operation*5. Although the folding type of *operation*5 is "inside-reverse-folding", the estimated result was "valley-folding". This is caused by twe reasons. One is that the shape of origami is affected by the thickness of paper. The other is that the difference between a camera-origami and a silhouette becomes larger as more folding operations are applied. Since this difference cannot be ignored with a complex origami work, we have to deal with this problem.

6 Conclusion

In this paper, we proposed the method which can estimate a folding operation with a camera image. Experimental results showed that our method can estimate folding operations at high speed. Also, estimation accuracy was enough high. Our future work is to introduce the model which expresses the state of origami, and then construct the folding operation support system.

References

[1] J. Kato, T. Watanabe, H. Hase, and T. Nakayama, "Understanding illustrations of origami drill books", Transactions of Information Processing Society of Japan, **41**(6), 2000, pp. 1857–1873.

[2] J. Mitani, "Recognition and modeling of paper folding configuration using 2D bar code", Transactions of Information Processing Society of Japan, **48**(8), 2007, pp. 2859–2867 (in Japanese).

[3] W. Ju, L. Bonanni, R. Fletcher, R. Hurwitz, T. Judd, R. Post, M. Reynolds, and J. Yoon,

"Origami desk: Integrating technological innovationand human-centric design", Proceedings of the 4th Conference on Designing Interactive Systems , 2002, pp. 399–405.

[4] T. Uchida, and H. Itoh, "Knowledge representaion of origami and its implementation", Transactions of Information Processing Society of Japan, **32**(12), 1991, pp. 1857–1873 (in Japanese).

[5] Y. Furuta, J. Mitani, and Y. Fukui, "Modeling and animation of 3D origami using spring mass simulation", In Proceedings of NICOGRAPH International 2008, (12), 2008, pp. 550–554.

[6] S. Miyazaki, T. Yasuda, S. Yokoi, and J. Toriwaki, "An origami playing simulator in the virtual space", The Journal of Visualization and Computer Animation, **7**(1) , 1996, pp. 25–42.

[7] H. Shimanuki, J. Kato, and T. Watanabe, "Recognition of forlding process from origami drill book", Proceedings of the Seventh International Conference on Document Analysis and Recognition, **32**(12), 2003, pp. 550–554.

[8] T. Terashima, H. Shimanuki, J. Kato, and T. Watanabe, "Understanding folding process of origami drill books based on graphic matching", Proceedings of IAPR Conference on Machine Vision Application, 2005, pp. 261–264.

[9] Y. Kinoshita, and T. Watanabe, "Estimation of folding operations using silhouette model", Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering and Computer Science 2009, WCECS 2009, 20-22 October, 2009, San Francisco, USA pp. 1270–1275

[10] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts", IEEE Transactions on Pattern Analysis and Machine Intelligence , **24**(4), 2002, pp. 509–522.

[11] Y. Kameda, M. Minoh, and K. Ikeda, "Three dimensional motion estimation of a human body using a difference image sequence", Proceedings of the 2nd Asian Conference on Computer Vision '95, 1995, pp. 181–185.

[12] J. Shi, and C. Tomasi, "Good features to track", Proceedeings of the 94th Computer Vision and Pattern Recognition , 1994, pp. 593–600.

[13] C. Harris, and M. Stephens, "A combined corner and edge detector", Proceedings of the 4th Alvey Vision Conference, 1988, pp. 147–151.

[14] J. Canny, "A computational approach to edge detection", IEEE Transaction on Pattern Analysis and Machine Intelligence , **8**(6), 1986, pp. 679–698.

[15] D.H. Ballard, "Generalizing the hough transform to detect arbitrary shapes", Pattern Recognition, **13**(2), 1981, pp. 111–122.

[16] D. Koller, J. Weber, and J. Malik, "Robust multiple car tracking with occlusion reasoning", Proceedings of the Third European Conference on Computer Vision, **1**, 1994, pp. 189–196.

[17] M.Z. Brown, D. Burschka, and G.D. Hager, "Advances in computational stereo", IEEE Transactions on Pattern Analysis and Machine Intelligence, **25**(8), 2003, pp. 993–1008.

[18] D. Scharstein, and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms", International Journal of Computer Vision, **47**(1-3), 2002, pp. 7–42.

[19] S. Kamijo, T. Nishida, and M. Sakauchi, "Occlusion robust and illumination invariant vehicle tracking for acquiring detailed statistics from traffic images", IEICE TRANSACTIONS on Information and Systems, **85-D**(11) , 2002, pp. 1753–1766.

[20] A. Senior, A. Hampapur, Y.L. Tian, L. Brown, S. Pankanti, and R. Bolle, "Appearance models for occlusion handling", Image and Vision Computing, **24**(11), 2006, pp. 1233–1243.

[21] Willow Garage, OpenCV (Open source Computer Vision), Library of programming functions for real time computer vision, http://opencv.willowgarage.com/.