Foldable Augmented Maps

Sandy Martedi*[1], Hideaki Uchiyama †[1], Guillermo Enriquez‡[1], Hideo Saito §[1],
Tsutomu Miyashita ¶[2], and Takenori Hara ‖[2]

[1]Keio University, Japan
[2]Dai Nippon Printing Co., Ltd.

ABSTRACT

This paper presents folded surface detection and tracking for augmented maps. For the detection, plane detection is iteratively applied to 2D correspondences between an input image and a reference plane because the folded surface is composed of multiple planes. In order to compute the exact folding line from the detected planes, the intersection line of the planes is computed from their positional relationship. After the detection is done, each plane is individually tracked by frame-by-frame descriptor update. For a natural augmentation on the folded surface, we overlay virtual geographic data on each detected plane. The user can interact with the geographic data by finger pointing because the finger tip of the user is also detected during the tracking. As scenario of use, some interactions on the folded surface are introduced. Experimental results show the accuracy and performance of folded surface detection for evaluating the effectiveness of our approach.

Index Terms: H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—Artificial, augmented, and virtual realities;

1 INTRODUCTION

Augmented paper maps have been getting a lot of attention since the past decade. In general, traditional paper maps can provide large scale and detailed information such as names of places and map symbols. However, the data on the traditional maps is static and tends to become out-of-date soon. In contrast, digital georeferenced data such as 3D buildings is unlimited and dynamically growing up. Recently, the integration between traditional maps and up-to-date digital geographic data has been discussed to enhance the functionality of the paper maps toward further novel uses as augmented paper maps.

Early augmented maps were based on the overlay of 3D geographic models using ARToolKit [12, 6]. In these systems, the user can interact with the models and watch them from arbitrary views with a live video see-through head-mounted display (HMD). In our previous work, we developed 2D standard maps with intersection dots printed in order to perform semantic registration between the maps and 3D geographic contents [32]. A projector-camera based table-top system focusing on 2D visualization has been developed by Reitamyr et al. [27]. In this system, geographic animations are projected onto the table and manipulated using a personal digital assistant (PDA). In outdoor use, positioning by global positioning system (GPS) is utilized as a trigger to find the user's position on a map [19, 21]. The location information also enables the user to access location based media.

The main technical issues in previous works are discussed on the assumption that a surface is composed of one plane. However, a typical use of a map includes folding, which was never discussed before. Because a surface is composed of multiple planes in this case, the assumption above cannot be applicable. The user would frequently fold and unfold the map to watch an appropriate region. In other words, folding is the action that changes the amount of the visible information according to the user's purposes. Because folding can be regarded as an important action in paper map manipulation, we develop folding based visualization and interaction for augmented maps.

In the development of foldable augmented maps, one of the main problems is how to recognize whether a map is folded or not. We need to use a single view image because we employ a typical augmented reality set-up using a video see-through HMD. In this case, the problem becomes the recovery of the surface shape of a reference plane from a single view image. Recently, the problem for a non-rigid surface was tackled [24, 10]. The solutions took the approximation of the surface by a collection of triangles. Compared to a non-rigid surface, a folded surface can be regarded as a description of a simple model such that the surface is composed of multiple rigid planes.

In this paper, we present folded surface detection and tracking for augmenting paper maps. This work is based on our previous work of single map image retrieval using 2D standard maps with intersection dots [32]. In folded surface detection, keypoint correspondences between an input image and a reference map are first established. From these correspondences, multiple planes are detected by iterative homography computation because the surface is composed of non-parallel multiple rigid planes. For a natural augmentation on the surface, the exact folding line is obtained by computing the intersection line of the planes. Based on the angle between the planes, we judge whether the map is folded or not. After the map is judged as folded, each plane is individually tracked by frame-by-frame descriptor update [31]. We extend our previous tracking for a plane to multiple planes. The angle between the planes is utilized as a trigger to switch the states between the detection and the tracking.

In our best knowledge, no other works have discussed folding visualization and interaction for papers that depend on the content of the paper in augmented reality. Because folding is a natural, usual and frequently-performed human behavior for papers, the development of the technique for detecting and tracking a folding surface is meaningful and important as a contribution to other researches in augmented reality.

The rest of the paper is organized as follows: the details of the previous augmented maps and augmentation on several surfaces are explained in Section 2. Section 3 described our folded surface de-

*e-mail: sandy@hvrl.ics.keio.ac.jp
†e-mail: uchiyama@hvrl.ics.keio.ac.jp
‡e-mail: nacho4d@hvrl.ics.keio.ac.jp
§e-mail: saito@hvrl.ics.keio.ac.jp
¶e-mail: Miyashita-T2@mail.dnp.co.jp
‖e-mail: Hara-T6@mail.dnp.co.jp

IEEE International Symposium on Mixed and Augmented Reality 2010
Science and Technolgy Proceedings
13 -16 October, Seoul, Korea
978-1-4244-9345-6/10/$26.00 ©2010 IEEE

tection. In Section 4, the detailed implementation for foldable augmented map is presented. User scenario of our foldable augmented map is discussed in Section 5. The accuracy and performance of our method is evaluated in Section 6, and Section 7 concludes this paper.

2 Related Works

The main issue in the studies of augmented maps is how to compute a relative pose between a camera and a paper map to overlay digital geographic contents. The approaches of the pose estimation can be divided into three categories: using fiducial markers, natural features and sensors.

Square marker based approaches using ARToolKit [15] are utilized in many applications because the use of the toolkit is the easiest way to develop augmented reality applications [12, 6, 21]. Because the visual appearance of the square markers is not appropriate due to the lack of the relationship with geographic data, another type of markers is sought in further researches. Rohs et al. printed a grid of points uniformly distributed over the map [28]. Compared to the square markers, the dot marker has better visibility because it occupies smaller space, but still has no relationship with geographic data. In our previous work, we have put a geographic meaning into scattered dots printed such as representing intersections for novel geo-visualization called augmented reality geographic information systems (AR GIS) [32]. These dots could be embedded into a map in terms of visibility because they are geographic data. In order to use the scattered dots as a marker, the local arrangements of dots are selected as descriptors called locally likely arrangement hashing (LLAH) [20].

Natural feature based approaches treat a map as a normal texture. In these approaches, the design of a descriptor for keypoint matching is an issue in general. A well-known local descriptor is scale-invariant feature transform (SIFT) that usually needs huge computational cost [18]. Several attempts to reduce the cost are performed to achieve real-time keypoint matching. Reitmayr et al. developed a SIFT-like descriptor based on the histogram of oriented gradients to be invariant to lighting and rotation change [27]. In their projector-camera system, they can ignore the scale invariance because the distance between a camera and a screen is fixed. Morrision et al. developed a mobile augmented reality map based on Phony SIFT [34] that works in a mobile phone [19]. Additionally, they evaluated the effectiveness of augmented maps compared to 2D standard digital maps.

Wireless communication technologies are also applied to combining paper maps with digital data. Reilly et al. embedded radio frequency identification (RFID) tags with the data onto the back side of a paper map [26]. In this system, the related data of the map is retrieved and displayed on a hand-held device when a user holds the device over the map. Compared to computer vision based approaches, precise geometric registration of overlaid content could not be performed.

Multiple plane detection is also another important issue for our foldable augmented maps. In the studies of computer graphics, Sechrest and Greenberg developed an edge based approach such that planes were detected from line segments [30]. Several approaches to find planes from keypoint correspondences between two images have been sought in the studies of computer vision. Vincent and Laganiere have developed a sequential RANSAC based homography computation [33]. Kanazawa and Kawakami improved the sampling way in this method such that they used the distribution of keypoints defined by the distance between a keypoint and other keypoints [14]. Because RANSAC tries to detect a single model at a time, it sometimes needs huge computational cost to find multiple models [8]. Zuliani et al. developed a method for finding multiple models in parallel called multiRANSAC [38]. As a method without homography computation, Zucchelli et al. utilized optical flow [37]. They modelled the motion of keypoints on a plane to segment planes from the cloud of keypoints. Heracles et al. developed a method for detecting planes in the cloud of 3D keypoints for a calibrated stereo camera [13].

The methods mentioned above dealt with only rigid planes. Because a plane is not always rigid, the shape of a non-rigid surface and other deformed surfaces also needed to be recovered for many virtual reality and computer vision applications [7, 4, 22, 23]. The non-rigid surface is usually modelled by a collection of triangles [24, 29]. Bellile et al. developed a method for estimating the non-rigid shape even in adverse conditions such as extreme self-occlusion [9, 10]. Kergosien et al. developed a mathematical model to simulate the bending and creasing behaviour of a paper-like sheet by taking into account the boundary points of the surface [16]. Bo and Wang used the geodesic of the surface instead of the boundary points to estimate the shape more accurately in intuitive manner [5]. Lee et al. took a different approach such that they put LED markers on the surface and utilized a hardware for camera tracking to project virtual contents on the surface [17]. Compared to a non-rigid surface, a folded surface can be simply modelled such that the surface is composed of multiple rigid planes.

3 Folded Surface Detection

To overlay a virtual object onto a surface, we need to estimate its relative pose to a camera. When a surface is folded, it consists of two planes. In this case, it is necessary to estimate relative pose of those two planes to the camera.

3.1 Folding Model

We first define our folding model for describing the relationship between two planes in a folded surface as follows. A surface can be folded in two directions based on the folding line: left-right folding and top-bottom folding as illustrated in Figure 1(a). Because we do not pre-define the position of folding lines, the user can horizontally and vertically fold the surface at arbitrary positions. Based on the folded shape, the folding ways can be classified into two types: mountain folding in Figure 1(c) and (e), and valley folding in Figure 1(b) and (d). The former is the case that the angle between two planes is more than 180 degrees. The latter is the case of less than 180 degrees.

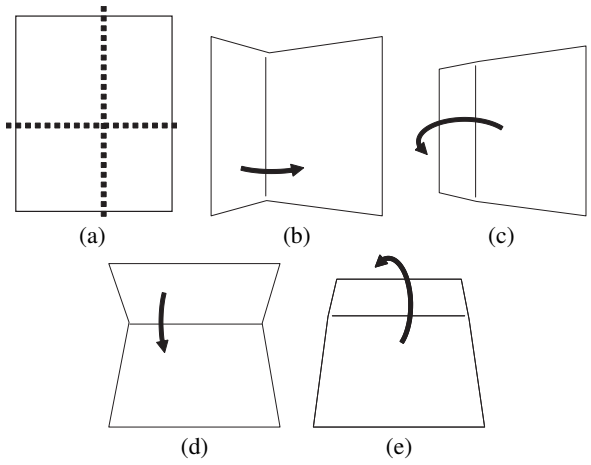


Figure 1: Folding model. (a) Defined folding directions. A surface is divided into two planes separated by either horizontal or vertical folding line at arbitrary positions. (b) Left-right valley folding. The angle between two planes is less than 180 degrees. (c) Left-right mountain folding. The angle is more than 180 degrees. (d) Top-bottom valley folding. (e) Top-bottom mountain folding.

3.2 Procedure Overview

In a pre-processing phase, we prepare a keypoint and descriptor database of reference planes. The keypoint database consists of a set of keypoints on each reference, which represents 2D distribution of keypoints. Each keypoint is related with its descriptors. Because we do not pre-define the position of folding lines, we do not store segmented planes beforehand.

Figure 2 represents the flowchart of our folded surface detection. First, we establish 2D correspondences between an input image and the reference plane by descriptor based keypoint matching. From the correspondences, multiple planes are detected by iterative geometric verifications. Next, we compute a folding direction and folding line from the positional relationship of the planes. The output of foldable surface detection is two edge points of the folding line on the reference.

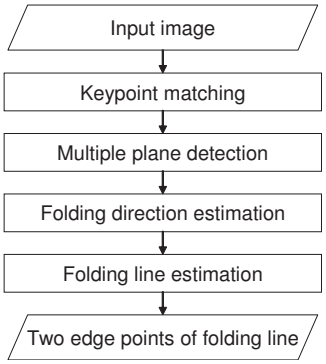


Figure 2: Procedure overview. The procedure of folded surface detection can be divided into four processes. From an input image, two edge points of the folding line on the reference surface are eventually output.

3.3 Multiple Plane Detection

Multiple planes inside a folded surface are detected from keypoint correspondences. We take a similar method with a sequential RANSAC based approach [33].

We first detect two planes of a folded surface. From an input image, we extract keypoints. For each keypoint, we utilize descriptor based keypoint matching to establish the correspondence with reference planes. From all the correspondences, we compute the first homography from the reference to the image with RANSAC. This means that the first plane is detected.

Then, in order to remove the keypoints in the first plane from the correspondences, we project the keypoints in the reference plane onto the image using the first homography. By thresholding the distance between each projected keypoint and its nearest keypoint in the image (we set 3 pixels), we exclude keypoints included in the first plane from the correspondences. For the rest of the correspondences, the second homography is computed with RANSAC to detect the second plane.

In fact, this multiple plane detection approach can be applied to a folded surface with many folds by setting the number of iterations for computing homographies.

3.4 Folding Direction Estimation

As described in Section 3.1, we define two folding directions. In order to estimate left-right or top-bottom folding of the paper, we simply use the relative position between two planes.

Suppose two planes are detected. We compute the center of each plane on the reference plane to make a vector connecting the centers. If the direction of the vector is horizontal, the folding direction is set to left-right. Otherwise, the folding direction is set to top-bottom. The estimated direction is utilized for folding line estimation in edge point estimation.

Even though the directions are defined, the user can dynamically change the folding position and the directions.

3.5 Folding Line Estimation

In a folded surface, two planes are segmented by a folding line. In order to achieve a natural augmentation on the surface, the exact folding line needs to be estimated.

From multiple plane detection, we have two homographies between an input image and a reference plane. This means that the reference plane is detected twice in the image as illustrated in Figure 3(a). In order to have the exact folding line, the intersection line of these two planes needs to be computed as illustrated in Figure 3(b).

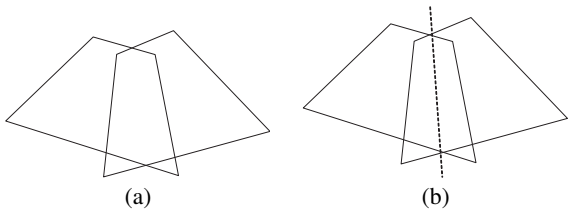


Figure 3: Folding line intersection. (a) A reference plane is detected twice in the image. (b) We estimate a folding line by computing the intersection line of two planes.

3.5.1 Coordinate Transformation

Two world coordinate systems independently exist in the image because two different correspondences between a reference plane and an image are established as illustrated in Figure 4(a). In order to estimate a folding line, these two planes should be described in the same coordinate system.

As a common coordinate system for two planes, we use a camera coordinate system ($\mathbf{X_c}$), where the origin is the camera center, $\mathrm{X_c Y_c}$ plane is parallel to the image plane and $\mathrm{Z_c}$ axis corresponds to a depth as illustrated in Figure 4(b). We compute the intersection line in the camera coordinate system.

For each plane, we first compute a 3×3 rotation matrix ($\mathbf{R}$) and 3×1 translation matrix ($\mathbf{T}$) from intrinsic camera parameters from camera calibration [36] and the homography ($\mathbf{H}$) computed in multiple plane detection [11]. Next, we project the world coordinate system of each plane onto the camera coordinate system by using $\mathbf{R}$ and $\mathbf{T}$ of each plane as

$$\mathbf{X_c} = [\mathbf{R_1}|\mathbf{T_1}]\,\mathbf{X_w} \quad (1)$$

$$\mathbf{X_c} = [\mathbf{R_2}|\mathbf{T_2}]\,\mathbf{X_w} \quad (2)$$

where $\mathbf{X_w}$ is the world coordinate system of the reference plane ($Z_w = 0$) and the suffixes (1 and 2) correspond to each detected reference plane. By using these equations, two planes are described in the same coordinate system.

3.5.2 Edge Point Estimation

A folding line has two edge points because the size of a reference plane is not infinite. Instead of directly computing an intersection line by solving equations for two planes, we compute two edge points of the folding line. An edge point is obtained by computing an intersection of two side lines.

In folding direction estimation, we categorized the direction into two types. The combination of corners to compose the side lines of the planes depends on the types.

Suppose we have two planes composed of four corners **ABCD** and **EFGH** for left-right folding in Figure 5(a). One edge point is

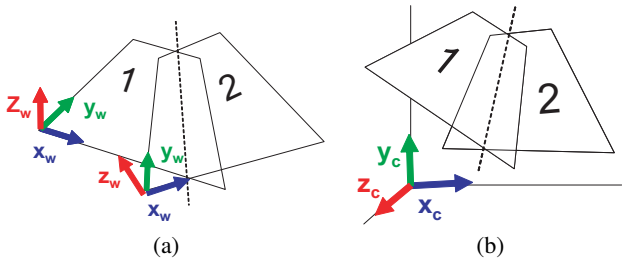


Figure 4: Coordinate transformation. (a) Each detected plane is related with a world coordinate system of the reference. (b) Two planes are transformed into a camera coordinate system to estimate a folding line.

obtained by computing the intersection from two lines **AB** and **EF**. The other point is obtained from two lines **DC** and **HG**. Each line is described as

$$\mathbf{X} = \mathbf{X_1} + a(\mathbf{X_2} - \mathbf{X_1}) \tag{3}$$

where $\mathbf{X}$ is a point on a line, $\mathbf{X_1}$ and $\mathbf{X_2}$ are two corners and a is a parameter. When we compute an intersection of two 3D lines in the camera coordinate system, we use a least square method because there are three equations with respect to two unknown parameters (each line of a). By computing two edge points from each set of corners, we obtain a folding line segment.

For top-bottom folding, the combination of corners is illustrated in Figure 5(b).

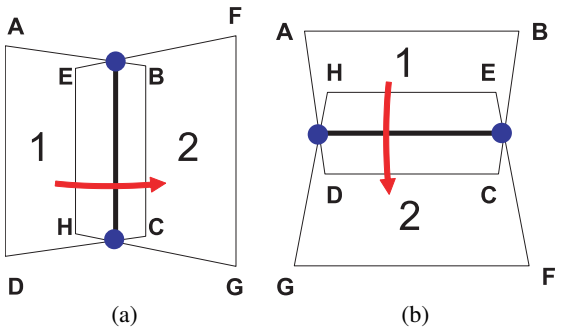


Figure 5: Edge point estimation. (a) Left-right folding case. We compute two edge points from two lines **AB** and **EF**, and two lines **DC** and **HG**. By connecting the points, a folding line is obtained. (b) Top-bottom folding case. We compute two intersections of two lines **AD** and **HG**, and two lines **BC** and **EF**.

4 Implementation

The main procedure in our foldable augmented map is similar with that for detecting and tracking multiple planes [35, 25]. Folded surface detection is equivalent to multiple plane detection. For the tracking process, we extend our previous tracking method for a plane [31] to multiple planes. Finger tip detection is also implemented for the interaction with overlaid virtual contents.

4.1 Database Preparation

In our previous work, we developed 2D standard maps with intersection dots [32]. Instead of intersection dots, we now utilize geographic symbols printed on a map as illustrated in Figure 6. This map is also generated from GIS. The symbols contain important spots such as streets, buildings, rivers, and railways. By using this map, the user can explicitly select a symbol to access its information in practical applications. As well as our previous work, we can use multiple maps.

As off-line procedure, we compute the descriptors of each keypoint (symbol) in the reference maps. The descriptors we utilize for keypoint matching is locally likely arrangement hashing (LLAH) [20]. In LLAH, a keypoint has multiple indices (1D descriptors) computed from the geometrical relationship of neighbor keypoints. Because each keypoint has unique symbol ID (= map ID + keypoint ID), each symbol ID is stored at the indices of the hash table as a descriptor database. The ID is also related with the world coordinate $\mathbf{X_w}$ ($Z_w = 0$) of the symbol.

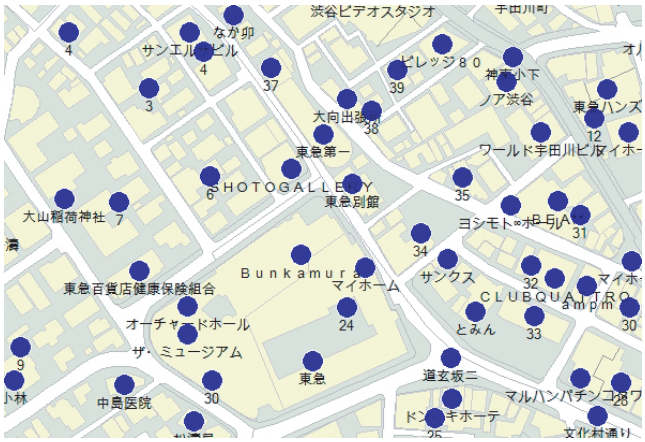


Figure 6: A part of the map with geographic symbols. Blue small circles are geographic symbols that we use. This map is generated from GIS. We extract the center of each circle by color segmentation as keypoint extraction.

4.2 Folding Initialization

As initialization, we compute the angle between two planes because we use the angle as a trigger to start tracking two planes. In other words, we judge whether a map is folded or not in the initialization based on folded surface detection.

In folded surface detection, the pixels of the symbol color are extracted from an input image. The center position of each symbol region is computed as a keypoint. Next, keypoint correspondences between the input and the reference maps are established by using LLAH. From the correspondences, the map ID is identified and two planes composing the folded surface are detected by iterative geometric verifications.

Because the planes are described in the same camera coordinate system, we can compute the geometrical relationship between two planes. The angle between two planes is computed using dot product of two side lines such as **AB** and **EF** in Figure 5(a). If the angle becomes more than a threshold (we set 120 degrees), the map is judged as folded and the multiple plane tracking starts.

The angle is also utilized for identifying the folding condition of mountain and valley. If the angle is larger than 180 degrees, the condition is mountain.

4.3 Multiple Plane Tracking

When the planes are not tracked, the map cannot be folded much because the range of keypoint matching with the descriptor database of the reference maps is narrow. In order to widen the range, it is necessary to track two planes. Because our previous tracking is performed to one plane [31], we extend it to multiple plane tracking. Additionally, to distinguish the processes of the detection and tracking, we prepare a new tracking descriptor database instead of updating the descriptor database of the reference planes.

4.3.1 Keeping Detected Planes as New Planes

After the folding initialization is finished, the reference plane (map) is segmented into two planes. In order to track the planes individually, we first keep each segmented map as a new one. This means that the keypoints and their descriptors in each new map are stored in the tracking database.

For keypoints (symbols) in each segmented map, we put a new symbol ID (= new map ID + keypoint ID). The symbol ID is also related with the world coordinate $\mathbf{X_w}$ ($Z_w = 0$). Because each keypoint has already had its descriptors (indices) in LLAH computed in folded surface detection, the symbol ID is inserted at the indices of the keypoint in the tracking database.

4.3.2 Tracking Detected Planes

In multiple planes tracking, keypoint matching is performed with the tracking database.

From an input image, keypoints are extracted by the same way as the folding initialization. Using LLAH, each keypoint in the image has a symbol ID retrieved from the tracking database. Next, all keypoints in the image are clustered by the map ID extracted from the symbol ID. For each keypoint cluster, we perform RANSAC based homography computation as geometric verification. Finally, we have two homographies corresponding to two planes.

4.3.3 Frame by Frame Descriptor Update

When the planes are tracked, the descriptors of keypoints in each plane are updated into the tracking database by the same way as our previous work [31].

For each plane, we project all keypoints in the reference map onto the image using the homographies. By this projection, the correspondences between keypoints in the reference and those in the image are established by thresholding their distances. If a keypoint in the image has a correspondence, the symbol ID of the projected keypoint is inserted at the indices of the keypoint in the tracking database.

4.4 State Transition

The state transition between unfolded and folded occurs in each image frame. The following explanation focuses on the valley folding as illustrated in in Figure 7.

From the initial state, two planes are detected. When the angle between two planes is less than the threshold angle, the state changes to folded. When the angle is more than the threshold, the state does not change.

In the folded state, the same process is performed. Two planes are detected and the angle between them is computed. When the angle is more than the threshold, the state comes back to the unfolded state. Otherwise, the state does not change.

In each state transition, the angle computation fails when only a plane is detected. As a result, we can not determine whether the state will change from the folded state into the unfolded state or vice versa. Thus, the state does not change and only single plane is processed for its augmentation.

We set the threshold angle as 120 degrees for the valley folding and 240 degrees for the mountain folding. In contrast with the valley folding, the state transition for the mountain folding uses the opposite comparison as follows. In the unfolded state, when the angle between two planes is more than the threshold, the state changes to folded. In the folded state, when the angle between two planes is less than the threshold, the state changes back to unfolded.

4.5 Augmentation

The virtual content of augmented maps we have is a set of 3D models of buildings provided by CAD CENTER CORPORATION in Japan. The coordinate system of each model is the same as that

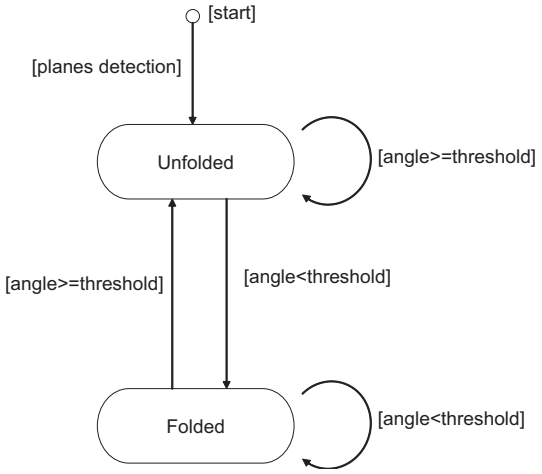


Figure 7: State transition for the valley folding. The folded and unfolded state is determined by calculating the angle between two detected planes. In the unfolded state, a plane is tracked. In the folded state, multiple planes are tracked.

of reference maps. Because the augmentation of each plane is individually performed, we divide the virtual content into different parts according to the size of the planes from the result of folded surface detection.

First, we load the whole part of the virtual content that corresponds to the map we use. While the map is not judged as folded in folding initialization, we render the virtual content entirely. When the map is judged as folded, the virtual content is divided into two parts at the estimated folding line. We then overlay the virtual content on each plane using each homography as illustrated in Figure 8.

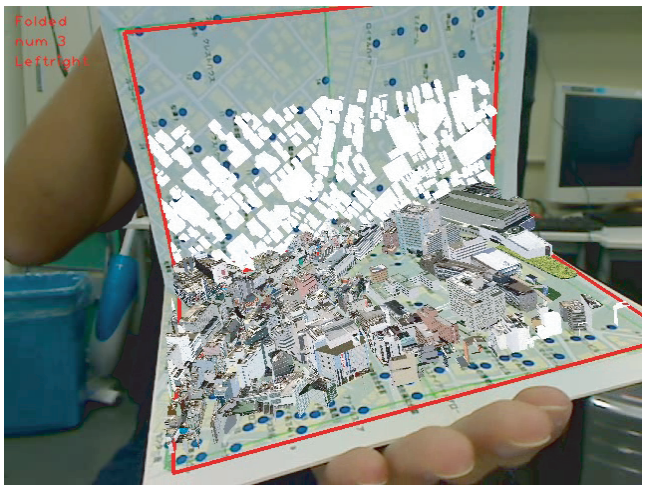


Figure 8: Augmentation on a folded surface. Because we have virtual contents on the reference plane, we can divide them into two parts according to the size of each plane, and overlay virtual contents on each plane independently.

4.6 Interaction

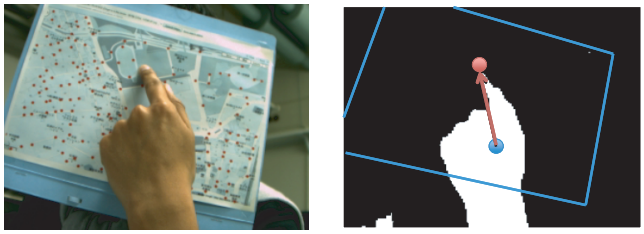
In the augmented paper maps, user interaction with the maps still need to be studied together toward a novel use. Hence, we implemented a method that enables users to select the symbols for accessing the related data. For finger pointing based interaction, finger tip detection is utilized.

4.6.1 Detecting a Finger Tip

After two planes are detected or tracked, we start to detect a finger tip from the image. In order to extract a hand region, we first use the simplest but accurate enough HSV color space classifier computed beforehand. By thresholding HSV for each pixel, we have a mask image of the hand region.

We assume that the user actually touches somewhere on the map as illustrated in Figure 9(a). In order to detect a finger tip, the user hand has to pose a pointing gesture. Also, the dorsal part of the hand should appear in the image entirely. We try to detect the upper end of the hand as a finger tip. It is obtained by computing the center of gravity of the hand region and finding the farthest point from the center as illustrated in Figure 9(b).

Because we have computed the region of the folded maps in the image, we can restrict our finger tip search to the map region only in the image.



(a) (b)

Figure 9: Finger tip detection. (a) User's pointing. A user points a map symbol with touching the map. (b) Definition of finger tip position. The farthest point from the center of the hand region is defined as a finger tip.

4.6.2 Accessing Related Data

The user can access the related data of each map symbol by pointing the symbol. Because we assume that the user actually touches the map while pointing, we search the nearest map symbol at the finger tip in the image.

We define pointing interaction by observing the position of finger tip. If a user's finger tip constantly stays close to one of map symbols in several consecutive frames, the map symbol is recognized as pointed. Then, we overlay the data related to the symbol after the pointing occurs. In our implementation, we overlay a picture as related data when a map symbol is pointed as illustrated in Figure 10.

5 Scenario of Use

We are seeking novel interaction with augmented paper maps because maps are really widespread and their functionality can be extended. As is well-known, folding is a common action when handling geographical, tourism or any kind of map. In this section, we introduce our scenario of the novel uses of the folded surface.

Useful augmented information could be shown depending on the inclination and distance to the camera. For instance, the system shows subway maps when the map is far from the camera. In contrast, it shows 3D buildings when the map is close from the inclined camera. The user can watch them as they pop from the map. Depending on user's position, the contents can be changed. This will be enabled by our tracking method.

The information shown onto each plane can be different because our method recognizes each segmented plane. Thus, various kinds of information can independently be shown on each plane at the same time. For example, we overlay 3D data on one plane and 2D data on the other plane. In addition, by recognizing hand gestures

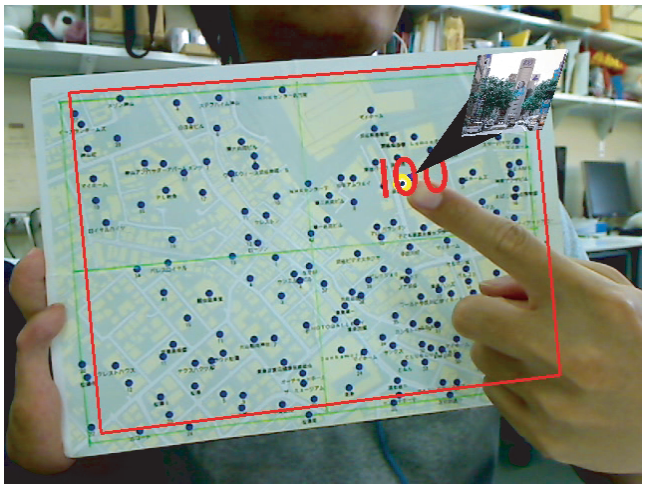


Figure 10: Accessing the data of each symbol. We overlay a picture and ID of the symbol as example contents when a map symbol is pointed.

we can zoom and scroll a certain area, or show the transit information.

Because folding action is not limited to maps, our method can be applied to any papers such as newspapers and books. For newspapers, handling its wide space is an interesting issue. The readers often fold and unfold a newspaper to read articles because of its wide space. We can apply our method to newspapers so that some virtual contents can be displayed according to the size of the article and the folding action.

By applying our method, we can add an innovative functionality to paper-based media. It will become interactive and offers a new experience to its readers. Hence, our method is useful for both AR researches and newspaper or books publishers.

6 Evaluation

Compared to our previous works [32, 31], the original parts of this work are mainly folded surface detection and multiple plane tracking. In this section, we evaluate the accuracy and performance of these two parts as new experiments. Additionally, we show the result of the recovery of a surface folded at arbitrary positions for folded surface augmentation.

We have tested these experiments using a desktop PC with specifications: Intel Quad core 2.8GHz, 4GB RAM and 640×480 pixel camera. The camera calibration is based on Camera Calibration Tools [1]. We implement our algorithm in C++ with OpenCV [2]. The 3D models are rendered by OpenVRML [3].

6.1 Accuracy of Estimated Folding Line

This experiment evaluates the accuracy of the estimated folding line in folded surface detection. We investigate the relationship between the accuracy and the number of keypoints included in two planes. The number of the keypoints appeared on each plane also indicates the robustness of our method against occlusions.

First, we prepare white papers and fold them at the center of each paper to make two planes. For each plane, we randomly put blue dots as described in Table 1. In one plane (first), we randomly put 70 blue dots. In the other plane (second), we change the number of random dots from 10 to 70. Therefore, we prepare 7 different papers.

For each paper, we estimate two edge points of the folding line by our method. We then project the two edge points onto the image coordinate system by using the computed camera pose. As a ground

truth, we manually click two actual edge points on the folded paper captured in the image. Because the projected edge points of the folding line and the ground truth are now in the same image coordinate system, they can be compared by a metric. For each edge point, we compute Euclidean distance as illustrated in Figure 11, and average the results. By comparing the projected edge points estimated by our method and the ground truth, we check the accuracy of the estimated folding line.

As described in Table 1, the accuracy increases when the number of keypoints in a plane increases. This means that the accuracy is mainly affected by the accuracy of the estimated homographies because the accuracy of the homography is improved by using more keypoints to disperse the error.

Table 1: The average error of estimated folding line. For one plane, we prepare the different number of keypoints. The error is almost proportional to the number of keypoints in each plane.

Number of tracking points in the first plane	Number of tracking points in the second plane	Error (pixels)
70	10	14.93
70	20	23.30
70	30	12.89
70	40	7.18
70	50	5.49
70	60	7.91
70	70	1.99

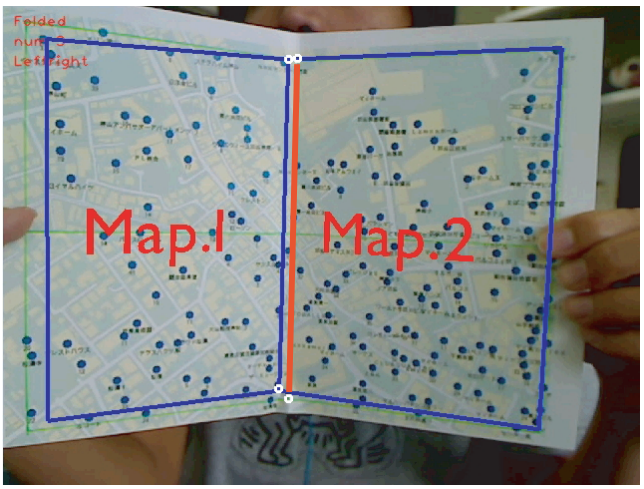


Figure 11: Accuracy of estimated folding line. The estimated folding line is projected onto an image. A red line is the actual folding line that is extracted manually by clicking. Each distance between two edge points of the projected line and the folding line is calculated.

6.2 Folding on Arbitrary Positions

The examples of folding on arbitrary positions are illustrated in Figure 12. As figures show, the user can vertically and horizontally fold the map.

The minimum area of each plane for successful detection depends on the number of keypoints included in each plane as discussed in Section 6.1. If the number of keypoints in one of the detected planes is more than 10, the folding detection can be performed, proved by the experiments. However, the estimated folded line includes the error in that case.

Because we use RANSAC based homography computation as geometric verification, two regions of a folded surface need to be rigidly planar. To keep two regions as planar surfaces, we used maps printed on thick papers. In the future, this constraint can be relaxed by using other deformable surface models described in related works.

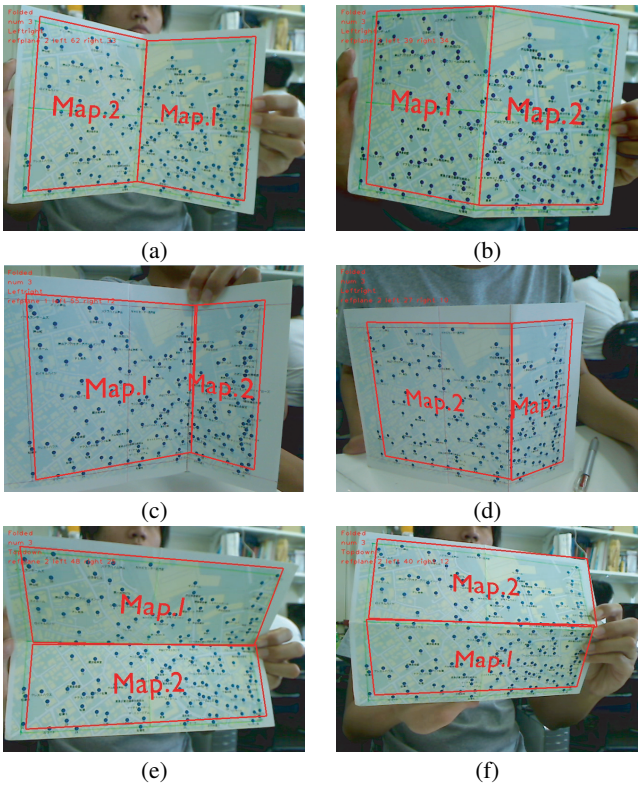


Figure 12: Folding results at arbitrary positions. (a) Left-right valley folding. (b) Left-right mountain folding. (c) Left-right valley folding at different position from (a). (d) Left-right mountain folding at different position from (b). (e) Top-bottom valley folding. (f) Top-bottom mountain folding.

6.3 Performance

Each processing time of our augmented maps implementation is described in Table 2. We recorded and averaged each computational cost of folded surface detection, multiple plane tracking and augmentation on 451 frames.

In folded surface detection, RANSAC based homography computation needed the most computational cost depending on the number of iterations. However, this computation can be replaced with multiRANSAC [38] for faster computation in the future. Because multiple plane tracking took about 3 ms, the augmentation during the tracking was over 30 frames per second.

The cost of augmentation depends on the number of 3D models included in the map. When the models are too much accurate and detailed, the number of polygons needs to be reduced.

7 Conclusions and Future Works

We presented a method for developing augmenting maps with a folded surface. Our method allows users to fold the paper to view the geographic data such as 3D building models. Hence, our augmented maps can display virtual contents during not only the camera orientation changes but also the folding interaction. In our method, we designed a model of a folded surface and procedure

Table 2: Computational time in the application. Folded surface detection needs the most computational cost because of RANSAC based homography computation. During the tracking, the augmentation is performed over 30 fps.

Tasks	Computational time (msec)
Folded surface detection	71.7
Multiple plane tracking	2.9
Augmentation	8.03

of tracking the folded surface. We extended our previous tracking by descriptor update for tracking multiple planes.

We presented experiments on folding a map to show the accuracy and performance of our method. The augmentation during the tracking is performed over 30 fps, which is almost the same as the time for capturing. The augmentation time depends on the number of polygons included in the map.

Folding is a challenging interaction to explore. In this paper we implement our folded surface model contains two folding lines: vertical and horizontal fold lines. We are planning to increase the number and orientation of folding lines so that the user can fold the surface arbitrarily anywhere on the surface.

ACKNOWLEDGEMENT

This work is supported in part by a Grant-in-Aid for the Global Center of Excellence for high-Level Global Cooperation for Leading-Edge Platform on Access Spaces from the Ministry of Education, Culture, Sport, Science, and Technology in Japan and Grant-in-Aid for JSPS Fellows. We thank CAD CENTER CORPORATION for providing the maps and 3D data in this work.

REFERENCES

[1] Camera Calibration Tools. http://www.doc.ic.ac.uk/ dvs/calib/main.html.
[2] OpenCV. http://sourceforge.net/projects/opencvlibrary/.
[3] OpenVRML. http://openvrml.org/.
[4] N. G. Ali, A. Z, R. Duraiswami, and L. S. Davis. Structure of applicable surfaces from single views. In *Proc. ECCV*, pages 482–496, 2004.
[5] P. Bo and W. Wang. Geodesic-controlled developable surfaces for modeling paper bending. In *Proc. Eurographics*, 2007.
[6] J. Bobrich and S. Otto. Augmented maps. In *ISPRS*, 2002.
[7] M. S. Brown and W. B. Seales. Image restoration of arbitrarily warped documents. *PAMI*, 26(10):1295–1306, 2004.
[8] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *C. of the ACM*, 24:381–395, 1981.
[9] V. Gay-Bellile, A. Bartoli, and P. Sayd. Direct estimation of non-rigid registrations with image-based self-occlusion. In *Proc. ICCV*, 2007.
[10] V. Gay-Bellile, A. Bartoli, and P. Sayd. Direct estimation of non-rigid registrations with image-based self-occlusion reasoning. *PAMI*, 32:87–104, 2010.
[11] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, second edition, 2004.
[12] N. R. Hedley, M. Billinghurst, L. Postner, R. May, and H. Kato. Explorations in the use of augmented reality for geographic visualization. *Presence*, 11:119–133, 2002.
[13] M. Heracles, B. Bolder, and C. Goerick. Fast detection of arbitrary planar surfaces from unreliable 3d data. In *Proc. IROS*, pages 5717–5724, 2009.
[14] Y. Kanazawa and H. Kawakami. Detection of planar regions with uncalibrated stereo using distribution of feature points. In *Proc. BMVC*, pages 247–256, 2004.
[15] H. Kato and M. Billinghurst. Marker tracking and HMD calibration for a video-based augmented reality conferencing system. In *Proc. IWAR*, pages 85–94, 1999.
[16] Y. Kergosien, H. Gotoda, and T. Kunii. Bending and creasing virtual paper. *IEEE CG&A*, 14(1):40–48, 1994.
[17] J. C. Lee, S. E. Hudson, and E. Tse. Foldable interactive displays. In *Proc. UIST*, pages 287–290, 2008.
[18] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60:91–110, 2004.
[19] A. Morrison, A. Oulasvirta, P. Peltonen, S. Lemmela, G. Jacucci, G. Reitmayr, J. Näsänen, and A. Juustila. Like bees around the hive: a comparative study of a mobile augmented reality map. In *Proc. CHI*, pages 1889–1898, 2009.
[20] T. Nakai, K. Kise, and M. Iwamura. Camera based document image retrieval with more time and memory efficient LLAH. In *Proc. CBDAR*, pages 21–28, 2007.
[21] V. Paelke and M. Sester. Augmented paper maps: Exploring the design space of a mixed reality system. *ISPRS*, 65:256–265, 2010.
[22] M. Perriollat and A. Bartoli. A quasi-minimal model for paper-like surfaces. In *Proc. CVPR*, volume 0, pages 1–7, 2007.
[23] M. Perriollat, R. Hartley, and A. Bartoli. Monocular template-based reconstruction of inextensible surfaces. In *Proc. BMVC*, 2008.
[24] J. Pilet, V. Lepetit, and P. Fua. Fast non-rigid surface detection, registration and realistic augmentation. *IJCV*, 76:109–122, 2008.
[25] J. Pilet and H. Saito. Virtually augmenting hundreds of real pictures: an approach based on learning, retrieval, and tracking. In *Proc. IEEE VR*, 2010.
[26] D. Reilly, M. Rodgers, R. Argue, M. Nunes, and K. Inkpen. Marked-up maps: combining paper maps and electronic information resources. *PUC*, 10(4):215–226, 2006.
[27] G. Reitmayr, E. Eade, and T. Drummond. Localisation and interaction for augmented maps. In *Proc. ISMAR*, pages 120–129, 2005.
[28] M. Rohs, J. Schoning, A. Kruger, and B. Hecht. Towards real-time markerless tracking of magic lenses on paper maps. In *adjunct Proc. Pervasive*, pages 69–72, 2007.
[29] M. Salzmann, F. Moreno-Noguer, V. Lepetit, and P. Fua. Closed-form solution to non-rigid 3d surface registration. In *Proc. ECCV*, pages 581–594, 2008.
[30] S. Sechrest and D. P. Greenberg. A visible polygon reconstruction algorithm. *ACM TOG*, 15:17–27, 1981.
[31] H. Uchiyama and H. Saito. Augmenting text document by on-line learning of local arrangement of keypoints. In *Proc. ISMAR*, pages 95–98, 2009.
[32] H. Uchiyama, H. Saito, M. Servieres, and G. Moreau. AR GIS on a physical map based on map image tetrieval using LLAH tracking. In *Proc. MVA*, pages 382–385, 2009.
[33] E. Vincent and R. Laganiere. Detecting planar homographies in an image pair. In *Proc. ISISPA*, pages 182–187, 2001.
[34] D. Wagner, G. Reitmayr, A. Mulloni, T. Drummond, and D. Schmalstieg. Pose tracking from natural features on mobile phones. In *Proc. ISMAR*, pages 125–134, 2008.
[35] D. Wagner, D. Schmalstieg, and H. Bischof. Multiple target detection and tracking with guaranteed framerates on mobile phones. In *Proc. ISMAR*, pages 57–64, 2009.
[36] Z. Zhang. Flexible camera calibration by viewing a plane from unknown orientations. In *Proc. ICCV*, pages 666–673, 1999.
[37] M. Zucchelli, J. Santos-Victor, and H. I. Christensen. Multiple plane segmentation using optical flow. In *Proc. BMVC*, pages 313–322, 2002.
[38] M. Zuliani, C. S. Kenney, and B. S. Manjunath. The multiransac algorithm and its application to detect planar homographies. In *Proc. ICIP*, pages 153–156, 2005.