Foldable Augmented Maps

Sandy MARTEDI\(^{(a)}\), Hideaki UCHIYAMA\(^{(b)}\), Guillermo ENRIQUEZ\(^{(c)}\), Nonmembers, Hideo SAITO\(^{(d)}\), Member, Tsutomu MIYASHITA\(^{(e)}\), and Takenori HARA\(^{(f)}\), Nonmembers

SUMMARY This paper presents a folded surface detection and tracking method for augmented maps. First, we model a folded surface as two connected planes. Therefore, in order to detect a folded surface, the plane detection method is iteratively applied to the 2D correspondences between an input image and a reference plane. In order to compute the exact folding line from the detected planes for visualization purpose, the intersection line of the planes is computed from their positional relationship. After the detection is done, each plane is individually tracked by the frame-by-frame descriptor update method. We overlay virtual geographic data on each detected plane. 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.

key words: augmented reality, paper interaction, folding model, tracking method

1. Introduction

In general, traditional paper maps can provide large scale and detailed information such as name of places and map symbols. However, the data on the traditional maps are usually static and tend to become out-of-date soon. In contrast, digital geo-referenced data such as 3D buildings is regularly updated. Recently, the integration of 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 [1]–[6].

Early augmented maps were based on the overlay of 3D geographic models using ARToolKit [1], [2], where 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 printed intersection dots in order to perform semantic registration between the maps and 3D geographic contents [3].

A projector-camera based table-top system focusing on 2D visualization has been developed by Reitmayr et al. [4] where the geographic animations are projected onto the table and manipulated using a personal digital assistant (PDA). In outdoor use, a global positioning system (GPS) is used as a trigger to find the user’s position on a map [5], [6]. The location information also enables the user to access the location-based media.

Previous works assumed that a surface is composed of one plane. However, a typical use of a paper map includes the folding, which was never discussed before. In that case, the assumption above cannot be applied because a surface is composed of multiple planes. Because folding can be regarded as an important action in paper map manipulation, we develop folding based visualization and interaction for augmented maps.

One of the main problems in foldable augmented maps is how to recognize whether a map is folded or not. Furthermore, we employ a typical augmented reality set-up using a video see-through HMD (monocular-based system) which make it more difficult. 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 [7], [8]. These solutions approximated the surface by a collection of triangles. Compared to a non-rigid surface, a folded surface can be simply modeled as a few multiple rigid planes.

We have presented our preliminary work on designing and implementing foldable augmented maps [9]. In this paper, we revisit the basic theory of the folded surface model and add more elaborated experiments on evaluating the accuracy of folding angle and the optimal threshold value for deciding the right moment to start the folded surface tracking.

This work is based on our previous work of single map image retrieval using 2D standard maps with intersection dots [3]. We first match keypoints between an input image and a reference map. From these correspondences, multiple planes are detected by iterative homography computation because the surface is composed of non-parallel multiple rigid planes. The exact folding line is then 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. When the map is folded, each plane is individually tracked by multiple planes tracking as the improvement of our last frame-by-frame descriptor update method [10]. The angle between the planes is compared with a threshold value to switch the state between the folded surface detection and
tracking.

In this paper, we also propose an augmented reality application based on the folded surface detection and tracking. Normally, the users want to match the 2D information on maps with real 3D scene in their heads when they lose their way outside. It is difficult to match them from user viewpoint in the ground. In this case, the users can watch 3D information of buildings and their texture on the folded maps as seen in Fig. 1. Thus, it helps the users know where they are.

To the best of our 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 contribution 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 Sect. 2. Section 3 describes our folded surface detection. In Sect. 4, the detailed implementation for foldable augmented map is discussed. User scenario of our foldable augmented map is discussed in Sect. 5. The accuracy and performance of our method is evaluated in Sect. 6, and Sect. 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 [11] are utilized in many applications because the use of the toolkit is the easiest way to develop augmented reality applications [1], [2], [6]. 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 [12]. Compared to the square markers, the dot marker has better visibility because it occupies smaller space, but still it has no relationship with geographic data. Our previous work included a geographic meaning into scattered dots printed such as representing intersections for novel geo-visualization called augmented reality geographic information systems (AR GIS) [3]. 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) [13].

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 [14]. 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 that is invariant to lighting and rotation change [4]. In their projector-camera system, they can ignore the scale invariance because the distance between a camera and a screen is fixed. Morrison et al. developed a mobile augmented reality map based on Phony SIFT [15] that works in a mobile phone [5]. Additionally, they evaluated the effectiveness of augmented maps compared to 2D standard digital maps.

Wireless communication technologies are also applied to combine 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 [16]. 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 planes 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 [17]. 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 [18]. Kanazawa and Kawakami improved the sampling way in this method. They used the distribution of keypoints defined by the distance between a keypoint and other keypoints [19]. Because RANSAC tries to detect a single model at a time, it sometimes requires huge computational cost to find multiple models [20]. Zuliani et al. developed a method for finding multiple models in parallel called multiRANSAC [21]. As a method without homography computation, Zucchelli et al. used optical flow [22]. They modeled 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 [23].

The methods mentioned above dealt with only a rigid surface. The shape of a non-rigid surface is necessarily to
be recovered. The non-rigid surface is usually modeled by a collection of triangles [7], [24]. The method attempted to estimate the position of triangles for each frame in order to estimate the deformation of the paper. Method proposed by Bellile et al [8] is also efficient for handling the deformation of the paper even in difficult conditions or against occlusions. Bo and Wang [25] use geodesic property for modeling paper that is suitable to satisfy the deformation of the paper. All methods above consider the deformation of the paper which very local planarity is recovered. Thus, they require a complex modeling and computation. Compared to a non-rigid surface as stated above, a folded surface can be simply modeled as multiple rigid planes.

Kergosien et al. [26] modeled a paper based on the boundary points of the surface. This method is effective for a paper since the boundary information is easy to obtain. The paper area can be easily distinguished from the environment. However, applying this model for our system requires an accurate boundary points detection. This is difficult when the boundary is occluded.

Lee et al. [27] used LEDs and sensors setup to track a folded surface. They also used a projector to display virtual contents. The tracking is relatively fast. The projector system allows the users see the content directly on the paper. However, the setup is not practical because every map should be equipped with sensors. In this case, detecting the content or appearance approach is more practical. Moreover, for outdoor use, the emitted light from projector is weak that makes it difficult to see.

3. Folded Surface Detection

To overlay a virtual object onto a surface, we need to estimate the relative pose of the surface to a camera. When a surface is folded, it consists of two planes. In this case, it is necessary to estimate the 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 Fig. 2 (a). Based on the folded shape, the folding ways can be classified into two types: mountain folding in Fig. 2 (c) and (e), and valley folding in Fig. 2 (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.

When the user folds a paper, we can assume the paper is folded in the middle. Using this fixed assumption, the folding model could be simpler. However, in the real environment, the user tends to fold a paper in conditions such as the folding line is not located in the middle of the paper. The user may also fold in another direction such as vertical folding. This will make the fixed folding assumption fail. To tackle this problem, we do not pre-define the position of folding lines in order to allow the user fold the surface horizontally and vertically at arbitrary positions. Instead, our model depends on the automatic model recognition that allows vertical, horizontal folding and even folding line in the arbitrary positions.

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 in each reference plane, 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. Instead, we automatically estimate the segmented planes in an online process.

Figure 3 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.

3.3 Multiple Plane Detection

We use a method similar to a sequential RANSAC-based
approach [18]. We first detect two planes of a folded surface. From an input image, we extract keypoints. For each
keypoint, we rely on descriptor based keypoint matching to establish the correspondence with reference planes. From all the correspondences, we compute the first homography for detecting the first plane from the reference to the image with RANSAC.

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.

3.4 Folding Direction and Folding Line Estimation

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. The user can dynamically change the folding position and the directions during online process.

In a folded surface, two planes are segmented by a folding line. In order to achieve a natural augmentation on the surface, it is necessary to estimate the exact folding line. In this case, the folding line is used as a separator between two planes. It is also used to divide the virtual contents according to the area of each plane.

As a result of multiple planes 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 Fig. 4 (a). In order to have the exact folding line, the intersection line of these two planes needs to be computed as illustrated in Fig. 4 (b).

3.4.1 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 Fig. 5 (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 (Xc), where the origin is the camera center. XcYc plane is parallel to the image plane and Zc axis corresponds to the depth as illustrated in Fig. 5 (b). We compute the intersection line in the camera coordinate system.

For each plane, we first compute a 3×3 rotation matrix (R) and a 3×1 translation matrix (T) from the intrinsic camera parameters obtained by the camera calibration [28] and the homography (H) computed in multiple planes detection [29]. Next, we project the world coordinate system of each plane onto the camera coordinate system by using R and T of each plane.

3.4.2 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 from the two planes, we compute two edge points of the folding line. An edge point is obtained by computing an intersection of two border (boundary) lines.

In folding direction estimation, we categorized the direction into two cases: left-right folding case and top-bottom folding case as shown in Fig 6. Suppose we have two planes composed of four corners ABCD and EFGH for left-right folding in Fig. 6 (a), one edge point is obtained by computing the intersection from two lines AB and EF. The other point is obtained from two lines DC and HG. The intersection from two lines AB and EF is described as

\[ X_A + a_{AB} (X_A - X_B) = X_E + a_{EF} (X_E - X_F) \]

where each side represents a line equation. When we compute an intersection of two 3D lines in the camera coordi-
nate 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 Fig. 6 (b).

4. Implementation

The main procedure in our foldable augmented map is similar to prior approaches [30], [31]. Folded surface detection is equivalent to multiple planes detection. For the tracking process, we extend a single plane tracking method [10] in order to support multiple planes tracking.

4.1 Database Preparation

In our previous work, we developed 2D standard maps with intersection dots [3]. Instead of intersection dots, in this work we use a real geographic symbols printed on a map as illustrated in Fig. 7. We prepared seven maps for demonstrating our system.

The number of symbols in our maps is considerably sufficient for the tracking purpose (approximately more than 100 dots in a single map). The symbols are geographic-related points of the map. They represent map features such as the position of buildings, houses, stations and important places. Their existence is not obstructive because they are meaningful data for the map. Therefore, the visualization of the keypoints on the map becomes important. At this stage, we visualize the symbols as black dots. It is possible to visualize the symbols as user-friendly map icons to show its importance for the map. In this case, the icon detection should also be considered. Moreover, for outdoor use, the color of the symbols is crucial. Strong sunlight may cause high saturation on the captured image so that the extraction becomes difficult.

As the offline procedure, we compute the descriptors of each keypoint (symbol) in the reference maps. We use the locally likely arrangement hashing descriptor (LLAH) [13] for keypoint matching.

LLAH is a keypoint-based descriptor. It employs the local relationship between a keypoint and its neighboring keypoints. Even when some keypoints are occluded, the global planarity of the map can still be obtained using the descriptors from the visible keypoints. As a result, LLAH is robust against occlusion. This advantage is suitable for our system when occlusions occur in case the user hands are moving over the surface map.

In addition, a few number of LLAH descriptors for each keypoint is sufficient for the tracking purpose. Therefore, the size of database does not require large memory space. Moreover, the LLAH database can be implemented as a hash table that speeds up the whole computation.

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 (an inverted file). The ID is also related with the world coordinate $X_w$ ($Z_w = 0$) of the symbol.

4.2 Folding Initialization

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 the two planes. The angle between the two planes is computed using the dot product of two border lines such as $AB$ and $EF$ in Fig. 6 (a). The angle is used for determining the folding states (folded and unfolded) and the folding conditions (mountain and valley). The details about folding angle is explained later in Sect. 4.4.

4.3 Multiple Planes Tracking

We model a folded surface as two connected planes. To detect and track a folded surface we need to extend our single plane tracking method [10] to multiple planes tracking. In contrast to single plane tracking method, we also insert the plane ID into the database to distinguish each plane from another.

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 $X_w$ ($Z_w = 0$). In every frame, the descriptors of keypoints are collected and inserted at the indices of the keypoint in the database.

In addition, for folding purpose, we prepare two descriptor databases for tracking: reference and folding database. Two databases are used in order to avoid false correspondences on keypoint matching. This is because the descriptors of a plane in a folded surface are the subset of the descriptors the reference plane. Therefore, when the surface
Fig. 8  The state transition for the valley folding. The folded and unfolded state are determined by comparing the angle between two detected planes with a threshold value. In the unfolded state, one plane is tracked and in the folded state, multiple planes are tracked.

is not folded, the tracking accesses and updates the default database. Accordingly, when the surface is folded, the tracking accesses and updates the folding database.

After the folding initialization has finished, the reference plane (map) is segmented into two planes. Keypoints and the descriptors that belong to each plane are copied from the reference database into the folding database. Then the planes are tracked individually using the folding database. The descriptors of each plane are updated in the folding database.

In folded state, a keypoint is matched by searching its descriptors inside the folding database. From an input image, keypoints are first extracted by using 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.

When the planes are tracked, the descriptors of keypoints in each plane are updated into the folding database as in [10]. 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 folding database.

4.4 State Transition

We discuss the valley folding case as illustrated in Fig. 8. In the initial state, two planes are detected. We use a threshold angle to determine the folding state. When the angle between two planes is smaller than the threshold angle, the state changes to folded. When the angle is bigger than the threshold, the state does not change.

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

In contrast with the valley folding, the state transition for the mountain folding uses the opposite comparison. In the unfolded state, when the angle between two planes is bigger than the threshold, the state changes to folded. In the folded state, when the angle between two planes is smaller than the threshold, the state changes back to unfolded.

In each state transition, the folding 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. However, we keep tracking the successfully detected plane and augment the 3D models.

4.5 Augmentation

We use a set of 3D models of buildings as the virtual contents provided by CAD CENTER CORPORATION, Japan. Each plane in the folded surface is augmented independently. Therefore, we divide the virtual contents into different parts according to the size of each plane.

When the map is not folded in folding initialization, we render the virtual content entirely. When the map is 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 Fig. 9.

5. Scenario of Use

We seek novel interactions with augmented paper maps because maps are widely used and their functionality can be extended. In this section, we introduce our scenario of the novel uses of the folded surface.

Normally, when the users hold a map and search destination around a city, they often refer to the map in order to find their destination. By using augmented maps, the user can see and compare the 3D models and the real buildings. The virtual contents on augmented maps can help the user recognize the surroundings.
Useful augmented information could be shown depending on the inclination and distance of the paper to the camera. For instance, we can display subway maps when the map is far from the camera. In contrast, it shows 3D buildings when the map is close to the camera. The user can watch them as they pop up from the map. Depending on the viewer or camera position, the contents can be changed. This mechanism can be achieved because our tracking method can produce the relative pose between surface and camera.

Our method segments a paper into two planes based on folding interaction. Thus, the information shown onto each plane may vary 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 we can zoom and scroll a certain area, or show the specific information such as transit and area details.

Our method can also 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 space and the folding action.

We can also apply our folded surface model to other printed media besides map because the keypoint matching (LLAH) works as long as a certain number of keypoints are extracted. Instead of map symbols, printed characters inside books or newspaper can be extracted as blobs and the center of blob can be used as keypoints. We can use color segmentation to extract the blobs. The implementation using a document is initialized in the original LLAH [13]. We have introduced our applications using the same keypoint matching method for documents and books in AR annotation [3] and clickable augmented document [9].

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

For experiments, we use a desktop PC with specifications: Intel (R) Core (TM) 2 Quad CPU Q9550 2.8 GHz, 4 GB RAM and 640×480 pixel camera. The camera calibration is based on Camera Calibration Tools [32]. We implement our algorithm in C++ with OpenCV [33] and augment the 3D models using OpenVRML [34]. In order to get quantitative evaluations, instead of running the program with a plugged camera, we first captured a scene when the user folds a paper and save it into a video sequence. For whole experiments, we use the video sequence in order to get a valid comparison and ideal environment regardless of the camera frame rate.

For each experiment, we use a database that contains one map. The number of points inside the map varies to the type of experiments.

6.1 Accuracy of Estimated Folding Line

This experiment evaluates the accuracy of the estimated folding line in folded surface detection. We study the relation between the folding accuracy and the number of tracking points included in two planes. We assume that the number of the tracking points appear on each plane also indicates the robustness of our method against occlusions.

First, we prepared paper sheets and fold them on the middle so that it forms 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 put several number of random dots from 20 to 70. We refer those random dots as tracking points or keypoints. For each sheet, we prepare one database. Therefore, the number of points inside the database is the same as the points printed in the sheet.

For each sheet, we estimated two edge points on the folding line and projected the them onto the image coordinate system by using the computed camera pose. We manually clicked the two actual edge points on the folded paper captured in the image as the ground truth. For each edge point, we computed its Euclidean distance as illustrated in Fig. 10, and averaged the results as the error of folding. We assume that a good folding accuracy is acquired when the error is close to zero. We performed experiments on both with and without tracking cases in order to study the impact of tracking to the folding accuracy.

As described in Table 1, when the tracking method is

### Table 1

<table>
<thead>
<tr>
<th>Number of tracking points (first plane)</th>
<th>Number of tracking points (second plane)</th>
<th>Error with tracking (pixels)</th>
<th>Error without tracking (pixels)</th>
</tr>
</thead>
<tbody>
<tr>
<td>70</td>
<td>20</td>
<td>23.04</td>
<td>35.98</td>
</tr>
<tr>
<td>70</td>
<td>30</td>
<td>8.76</td>
<td>29.51</td>
</tr>
<tr>
<td>70</td>
<td>40</td>
<td>6.63</td>
<td>31.89</td>
</tr>
<tr>
<td>70</td>
<td>50</td>
<td>6.69</td>
<td>15.93</td>
</tr>
<tr>
<td>70</td>
<td>60</td>
<td>5.95</td>
<td>23.42</td>
</tr>
<tr>
<td>70</td>
<td>70</td>
<td>2.81</td>
<td>19.49</td>
</tr>
</tbody>
</table>

![Fig. 10](image) Accuracy of the estimated folding line. The distances between the two edge points of the projected line and the ground truth folding line are calculated.
used, the accuracy increases with the number of tracked points. This means that the accuracy is mainly affected by the estimated homographies because the accuracy of the homography is improved by using more tracking points to disperse the error.

We also calculated the accuracy of folding without tracking (the last column in Table 1). We assume that tracking will stabilize the detection so that the folding line position doesn’t change in each frame.

By comparing the error from the results, we know that the tracking yielded a better accuracy for the folded surface detection. It stabilizes the keypoints detection because the information from previous frames are kept. Tracking the planes also keeps the folding line remain in the same position. On the other hand, the error of folding line doesn’t decrease significantly even the number of tracking points increases when tracking is not used. Thus, this experiment proves that tracking can increase the folding accuracy.

There is also error discrepancy with and without tracking. The experiment with tracking stores the descriptors of previous frame into database. The database contains descriptors of the tracked frames from first to current frame. It contains sufficient and accurate information about the keypoints distribution in previous frames. This technique minimizes the computation error in current frame. Thus, the homography computation error is small. On the other hand, removing the tracking makes the database only contains descriptors for top view image. For tilted conditions, the detection fails or the homography computation gives inaccurate result because there is not enough information about tilted conditions in the database. It yields higher error. This make the error of computation is different with and without tracking.

6.2 Accuracy of Estimated Folding Angle

This experiment evaluates the accuracy of estimated folding angle. We used a paper map which contains 140 keypoints. We placed it in a fixed position in front of the camera. We then prepared a folding line on the paper such that it divides the paper into two planes and each plane has 70 keypoints. First, we detected and tracked the map. We then folded the map on fixed angles (we manually measure the fixed angle using a protractor). We used these manually measured angles as the ground truth data.

We then compared the folding angle values produced by the folded surface detection with the ground truth data. On each angle, we estimated the folding angle from several consecutive frames and averaged them. The result is shown in Fig. 11.

Our detection method can calculate the angle between plane from 40 degrees to 290 degrees. Outside that range, due to extreme tilt, our method successfully detected and tracked one plane. In this case we do not calculate the folding angle between two planes.

We can see that the detected angle is almost correctly estimated around the planar condition (the average error from angle 150 degrees to 210 degrees is 4.19 degrees). On valley folding (angle range is between 40 degrees and 180 degrees), the angles tend to be higher than the ground truth angle. On the other hand, the mountain folding (angle range is between 180 degrees and 290 degrees), the angles tend to be lower than the ground truth. The average error of the estimated folding angle from all of experiment set is 9.07 degree. This average error information can be taken into account in order to improve folding accuracy by optimizing the folding angle.

6.3 Optimal Threshold Angle for Folding

After we know the accuracy of folding angle in the section Sect. 6.2, we determine an angle as threshold value for changing from planar state into folded state and vice versa. To be precise, this threshold value should be a value near 180 degrees. However, due to the tolerance of planarity in RANSAC, two planes can not be detected distinctively in near planar condition. In addition, in near planar condition the orientation of two planes change from frame to frame that makes the folding line estimation inaccurate. Therefore, it is necessary to determined the threshold value to start the folding line estimation.

The threshold value can be determined arbitrarily. Suppose there is an optimal value that can be achieved from stable detection of two planes, this value can be determined based on the number of detected keypoints. We assume the maximum detected keypoints will determine the best condition for folding because the number of detected keypoints is equivalent with the accuracy of plane tracking. Therefore, an optimal threshold value depends on the number of detected keypoints.

In this experiment, we studied the relationship between the calculated folding angle and the number of detected keypoints. When largest number of detected keypoints in a certain angle is obtained, we assume that angle is the optimal threshold value. We used a paper map which contains 140 keypoints. We placed it in a fixed position in front of the camera. We then prepared a folding line on the paper such that it divides the paper into two planes and each plane has...
70 keypoints. We folded the paper based on this folding line on several angles. Then we observed the number of detected keypoints as shown in Fig. 12 for valley folding and Fig. 13 for mountain folding.

From the result we can see that the number of keypoints in the first plane tends to get larger close to the planar condition (angle 150 degrees to 210 degrees). This is because in those angles, the paper can be regarded as planar even though the paper is slightly folded. Accordingly, keypoints are mainly acquired in the first plane. On the other hand, the second plane tends to decrease. The keypoints in the second plane are regarded as outliers of the homography computation in the first plane. As a result, it is difficult to distinguish the first plane and the second plane in nearly planar condition.

There is a certain angle where the average of detected keypoints in the first plane and second plane is maximal. The detected keypoints are distributed equally on both planes. We use this condition as the best time to start the folded state and the angle is the optimal threshold for folding. From the experiment we achieved the optimal threshold of valley folding is 152 degrees with the average keypoints on each plane is 63 and the optimal threshold for mountain folding is 206 degrees with the average detected keypoints is 62.

### 6.4 Folding at Arbitrary Positions

The examples of folding at arbitrary positions are illustrated in Fig. 14. The figure shows that the user can vertically and horizontally fold the map.

We consider the folding in arbitrary position in our scenario to relax our constraint (half side folding). In fact, in practical use, the paper may be folded at arbitrary positions accidentally. If we ignore this condition and divide the map equally when the user folds at arbitrary positions, the number of keypoints in the database and the map will differ. Some keypoints that belong to one plane will be misplaced into another plane. As a result, this difference will lead to inaccurate homography computation. On top of that, accommodating multiple folding lines requires folding in arbitrary position. Toward a general folding interaction, it is necessary to consider the folding line at arbitrary positions.

The minimum area of each plane for successful detection depends on the number of keypoints included in each plane as discussed in Sect. 6.1. If the number of keypoints in one of the detected planes is equal or higher than 20, the folding detection can be performed, as proved by the experiments. However, the estimated folded line includes error in that case.

All experiments in our evaluation always use the ideal condition by folding the paper fairly into two sides where the folding line is located in the middle of the paper. We assume that the result of this ideal condition will also apply to folding lines in arbitrary positions. In fact, the folding at arbitrary positions is equivalent to the half side folding with different number of keypoints on each plane. Therefore, the accuracy of folding at arbitrary positions is proven.
in Sect. 6.1. The accuracy of folding line and the angle will decrease since dividing the plane unequally is similar to reducing the number of keypoints in one side of plane.

6.5 Performance

Processing time of the different tasks in the implementation is given in Table 2. We recorded and averaged each computational cost in consecutive 451 frames. In the folded surface detection, the RANSAC-based homography computation is the most costly task. However, this computation can be replaced with multiRANSAC [21] for faster computation in the future. Because tracking multiple planes takes at about 3 ms, the augmentation during the tracking was at over 30 frames per second.

The computation cost for augmentation depends on the complexity if the 3D models. Simplifying the 3D models can speed up the augmentation. In our implementation, we simply use a region clipping method for separating 3D models in each plane. Overlapping 3D model around the folding line requires an automatic visualization handling such as remeshing based on the clipping. This process demands more computational time which will be a challenge for our future research.

7. Conclusions and Future Works

We presented a method for developing augmented maps with a folded surface. Our method allows the augmentation of the virtual contents not only when the camera orientation changes but also when the map is folded. In our approach, we model folded surface and develop a method to detect and track it. We extend our single plane tracking by descriptor update to multiple-plane tracking.

We presented experiments on folding a map to show the accuracy and performance of our method. The augmentation when tracking is performed at over 30 fps, which is sufficient for real-time application.

Folding is a challenging interaction to explore. Our folded surface model considers only one vertical or horizontal folding line. We are planning to increase the number and orientation of folding lines so that the user can fold the map anywhere on the surface.

Acknowledgment

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.

References

2009.


[34] “OpenVRML.” http://openvrmrl.org/

Sandy Martedi received B.Eng. degree from Bandung Institute of Technology, Indonesia in 2005. He received M.Eng. degrees in engineering in 2010 and is currently in Ph.D. course of Department of Computer Science at Keio University, Japan. His research interests are augmented reality, 3D reconstruction and Geographical Information System (GIS).

Hideo Saito received the B.Eng., M.Eng., and Ph.D. degrees in electrical engineering from Keio University, Japan, in 1987, 1989, and 1992, respectively. He has been on the faculty of Department of Electrical Engineering, Keio University, Japan since 1992. From 1997 to 1999, he was a visiting researcher at the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, where he joined the virtualized reality project. From 2000 to 2003, he was also a researcher with PRESTO, JST. Since 2006, he is a professor in the Department of Information and Computer Science, Keio University. He is a member of IPSJ, VRST, ITE, IIEEJ, and SICE. His research interests include computer vision, image processing, augmented reality, and human-computer interaction.

Hideo Uchiyama received the BS., MS., and Ph.D. degrees in engineering from Keio University, Japan, in 2006, 2007, and 2010, respectively. In 2008, he was a visiting scholar on the project for augmented reality applications for GIS at Ecole Centrale de Nantes, France. He is currently a postdoctoral fellow at INRIA Rennes, France. His research interests include vision based augmented reality, feature matching and tracking.

Guillermo Enriquez received the B.Eng. in Information Science and Engineering from Shibaura Institute of Technology in 2009 and M.Eng., from Keio University in 2011, Japan. He has interest in image processing, augmented reality and human-computer interaction research as well as mobile and web applications development.

Tsutomu Miyashita is the general manager of interaction design department at DNP Digitalcom Co., Ltd. the subsidiary company of Dai Nippon Printing Co, Ltd. He has been in charge of application development for Smartphone and Tablet PC and evaluation for new business model.

Takenori Hara received the B.Eng., M.Eng. degrees in Computer Science from Kyushu Institute of Technology, Japan, in 1996 and 1998, respectively. He has been on the researcher of Media Technology Research Center, Dai Nippon Printing, Japan since 1998. From 2005 to 2008, he was a visiting researcher at the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, where he joined the computational photography project. His research interests include computer vision, image processing, augmented reality, and human-computer interaction.