

Paper

A Study on Effect of Automatic Perspective Correction on Exemplar-based Image Inpainting

Hiroto Sasao (member)†, Norihiko Kawai†, Tomokazu Sato (member)†, Naokazu Yokoya (member)†

Abstract Image inpainting has been widely investigated to remove undesired parts of images. One of the effective approaches is exemplar-based inpainting, which uses texture patterns in an image as exemplars for filling in missing regions. As one of exemplar-based methods, an inpainting method based on automatic perspective correction using vanishing points was recently proposed. However, the target scene is limited to artificial one in which vanishing points are easily detectable. Although some other methods for automatic perspective correction have also been proposed, the effect on image inpainting has not been evaluated in detail so far. This paper analyzes the effect of multiple methods for automatic perspective correction on image inpainting by developing a method that combines a variety of automatic perspective correction methods and image inpainting. Specifically, we examine the influence of the amount of perspective distortion and characteristics of textures on image inpainting results by using images distorted by simulation. We also examine the effect using real images. In addition, we demonstrate the advantage of employing multiple criteria for perspective correction over the conventional method from the analyzed results.

Key words: Image Inpainting, Correcting Perspective Distortion, Texel Size, Matrix Rank, Vanishing Point.

1. Introduction

Image inpainting, which removes undesired objects in an image and automatically fills in the missing regions in a plausible manner, has been widely investigated to increase the utility value of images. Although many kinds of methods for image inpainting have been proposed so far[1–8], exemplar-based methods have been enthusiastically developed as one of the most effective approaches in the field of image inpainting because they can generate natural and complex textures even if missing regions are large.

The exemplar-based approach assumes that the textures that are similar to hidden ones behind missing regions exist in some part of a given image. Based on this assumption, a number of exemplar-based methods search the image except missing regions for appropriate textures and generate textures in the missing regions by synthesizing them[9–15]. Among them, a global optimization-based approach, which minimizes an energy function to generate plausible textures in any part of missing regions, has been intensively developed because of its effectiveness[13–15]. However, such exemplar-based methods often suffer from the shortage of appropriate exemplars in the image for missing regions.

To overcome the problem, some methods have attempted to increase available exemplars by considering rotation, scaling and reflection of texture patterns in the image[16,17]. These methods prepare patterns transformed with discrete parameter values before starting inpainting and use them as exemplars. However, usually projective transformation has not been considered in this framework because it contains a larger number of unknown parameters than other transformations mentioned above and it is difficult to generate the appropriate appearance of projectively transformed texture by just changing parameter values discretely.

For this problem, the correction of perspective distortion has been considered to determine parameter values rather than preparing discrete multiple values in projective transformation. As image inpainting methods using the correction of perspective distortion, those with manual interaction were proposed[18,19]. However, these methods require users to have special skills to correct perspective distortion. To relax the problem, an image inpainting method based on automatic estimation of projective transformation using vanishing points was recently proposed[20]. Unfortunately, the target scene must have parallel lines so that vanishing points can be detected in this method.

Received June 16, 2015; Revised September 30, 2015; Accepted November 16, 2015

† Nara Institute of Science and Technology (Nara, Japan)

On the other hand, in the field of computer vision, some methods for automatic perspective correction from a single image have been proposed[21,22]. Although these papers claim that image inpainting is one of the applications to which their methods are applicable, actual effect on image inpainting has not been validated.

In this paper, to obtain good inpainting results for a variety of images, we analyze the effect of the state-of-the-art methods for perspective correction on exemplar-based inpainting. Specifically, by using images distorted by simulation, we examine the influence of the amount of perspective distortion and characteristics of images on image inpainting results with a user study. We also examine the effect using real photographs. For the analysis, we have developed an image inpainting pipeline to combine automatic perspective distortion and example-based image inpainting with automatic adjustment of the resolution.

In addition, through the analysis of the effect of perspective correction methods on image inpainting, we demonstrate the advantage of employing multiple perspective correction criteria over a conventional method, which employs a single correction criterion for image inpainting.

The rest of this paper is organized as follows. We describe the pipeline of image inpainting with automatic perspective correction in Section 2. Section 3 introduces the overview of three state-of-the-art perspective correction methods[20–22] used for analysis and discusses the issues when they are applied to image inpainting applications. Section 4 describes a method for resolution adjustment required when combining image inpainting and perspective correction. In Section 5, we show inpainting results for simulated and real images and analyze them using a user study. We finally conclude this paper in Section 6.

2. Pipeline of Image Inpainting with Automatic Perspective Correction

Figure 1 shows the flow diagram of image inpainting combined with perspective correction. After inputting an image with missing regions (which are illustrated as red regions in images in this paper) (a), we apply one of the automatic perspective correction methods to the input image to obtain projective parameters in a homography matrix (b). We also calculate scaling parameters to adjust the resolution of the transformed image (c). Using the homography matrix H that consists of the projective and scaling parameters, we transform the input image (d), and apply an exemplar-based inpainting method to the transformed image (e). Finally, we obtain the result by transforming the inpainting result with the inverse matrix H^{-1} and copying the parts corresponding to the missing regions to the original input image (f).

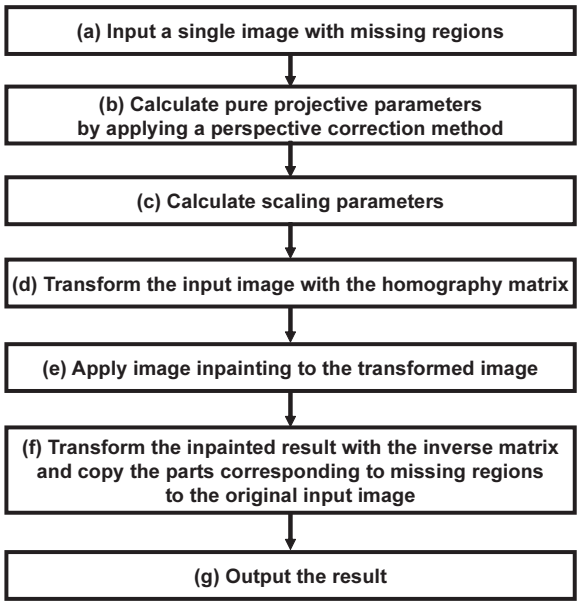
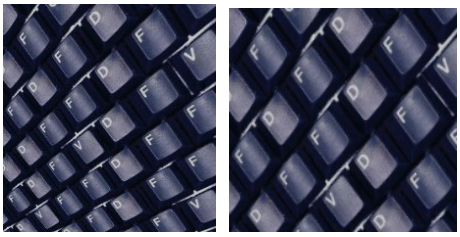


Fig. 1 Flow diagram of image inpainting with perspective correction.

(a) Original image (b) Correction of perspective distortion (c) Correction of affine distortion

Fig. 2 Distortion correction.

Here, a homography matrix typically has nine parameters and can get a distortion-corrected image, as shown in Fig. 2(c), from the original image in Fig. 2(a). However, the correction of only perspective distortion is usually sufficient for image inpainting because it makes the sizes and shapes of objects on an image plane uniform if they are the same in three-dimensional space as shown in Fig. 2(b). Therefore, only two parameters, which are referred to as pure projective parameters, are required to correct perspective distortion. However, since the pure projective parameters cannot adjust the scale, which is important not to generate blurry results when an inpainting result is transformed back to the original image with H^{-1}, we need two additional parameters for scaling. Therefore, we define a homography matrix to correct perspective distortion for image inpainting as



(a) Input image (b) Corrected image

Fig. 3 Example of perspective correction by equalizing texel sizes[21].

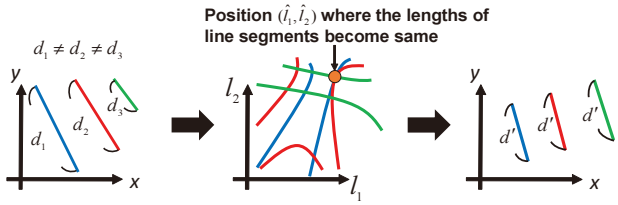


Fig. 4 Flow for equalization of lengths of line segments[21].

follows:

$$H = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ l_1 & l_2 & 1 \end{bmatrix}, \tag{1}$$

where l_1 and l_2 are pure projective parameters, and s_x and s_y are scaling parameters for horizontal and vertical directions, respectively.

In this study, we employ three methods for automatic perspective correction: (1) texel size-based method[21], (2) matrix rank-based method[22], and (3) vanishing point-based method[20]. For image inpainting, we mainly employ the existing exemplar-based method[16], which is one of the state-of-the-art methods[13–17] that use patches to search for similar patterns and update pixel values using the similar patterns in common. However, we also confirm the correlation between the method[16] and another inpainting method in experiments. In the following two sections, we first describe the overview of these perspective correction methods in Section 3. We then describe a method for adjusting the resolution of the transformed image for image inpainting in Section 4.

3. Methods for Automatic Perspective Correction

3.1 Texel Size-Based Method

Cyclic objects such as tiles on the ground and wall with the same size are usually transformed to ones with different sizes on an image plane by perspective projection. The texel size-based method[21] corrects the perspective distortion by calculating a homography matrix that equalizes the texel sizes using Hough transform as shown in Fig. 3. In the following, we briefly describe the method to calculate l_1 and l_2 in Eq. (1) in which $s_x = 1$ and $s_y = 1$.

This method first divides an image into segments and fits an ellipse to each segment. The length d_i of long axis of the ellipse fitted to the i-th segment is regarded as its texel size. Here, given that both endpoints of i-th long axis are (x_{i1}, y_{i1}), (x_{i2}, y_{i2}), the length is represented as follows.

$$d_i = \sqrt{(x_{i1} - x_{i2})^2 + (y_{i1} - y_{i2})^2}. \tag{2}$$

Given coordinate (x'_{i1}, y'_{i1}) and (x'_{i2}, y'_{i2}) that are transformed from (x_{i1}, y_{i1}) and (x_{i2}, y_{i2}) by a homography matrix, the relationship between the coordinates is represented as follows:

$$\begin{aligned} x'_{i1} &= \frac{x_{i1}}{l_1 x_{i1} + l_2 y_{i1} + 1}, \; y'_{i1} = \frac{y_{i1}}{l_1 x_{i1} + l_2 y_{i1} + 1} \\ x'_{i2} &= \frac{x_{i2}}{l_1 x_{i2} + l_2 y_{i2} + 1}, \; y'_{i2} = \frac{y_{i2}}{l_1 x_{i2} + l_2 y_{i2} + 1} \end{aligned} \tag{3}$$

Here, suppose that d' is the texel size after equalization, the following equation is satisfied.

$$\begin{aligned} d' &= \sqrt{(x'_{i1} - x'_{i2})^2 + (y'_{i1} - y'_{i2})^2} \\ &= \sqrt{(\phi x_{i1} - \varphi x_{i2})^2 + (\phi y_{i1} - \varphi y_{i2})^2} \end{aligned} \tag{4}$$

where $\phi = \frac{1}{l_1 x_{i1} + l_2 y_{i1} + 1}$, $\varphi = \frac{1}{l_1 x_{i2} + l_2 y_{i2} + 1}$. Given a certain d', Eq. (4) can be represented in l_1 - l_2 space by the similar way to Hough transformation as illustrated in Fig. 4. If the curves projected from different line segments cross at point $(\hat{l}_1, \hat{l}_2)$, the line segments are transformed to those with the same length d' by a homography with parameters $\hat{l}_1$ and $\hat{l}_2$. Crossing points are voted while changing d', and the most voted point is finally determined as pure projective parameters.

This method is effective for scenes including multiple objects whose sizes are almost the same in the 3D environment such as windows, tiles and iterative patterns on walls in artificial environments. However, the perspective correction does not work appropriately when each texel is not accurately extracted even if the same objects are arranged. When this method is applied to image inpainting, missing regions have little affect on the voting, resulting in little influence on perspective correction. However, it becomes more difficult to detect useful texels as missing regions become larger.

3. 2 Matrix Rank-Based Method

The matrix rank-based method[22] treats an image as a matrix whose elements are pixel intensities and calculates a homography matrix that transforms the original matrix to row-rank one. Specifically, given that input image I is distorted from the corrected image I^0 by homography matrix H^{-1}, this problem is formulated as follows:

$$\arg\min_{I^0,E,H} ||I^0||_* + \lambda||E||_1 \ \text{ subject to } \ I \circ H = I^0 + E, \tag{5}$$

where $||\ ||_*$ represents the matrix nuclear norm, which approximates the matrix's rank, $||\ ||_1$ represents the matrix ℓ_1-norm, E is a noise matrix, and λ is a positive weighting factor. The optimization problem is solved by iterative linearization of the constraint by augmented Lagrange multiplier method.

This method can work for some non-artificial scenes more effectively than texel-based or vanishing point-based methods because it does not require objects with the same size or parallel lines. However, since all pixel values in the image are used for optimization, it may not work for the scene in which various objects are randomly placed. When this method is applied to image inpainting, the perspective correction is largely affected by the shape and size of missing regions because pixel values in the missing regions are also used for optimization. Since this problem is essentially difficult to avoid, in experiments described in Section 5, we give a gray value to the missing regions before applying this correction method.

3. 3 Vanishing Point-Based Method

The vanishing point-based method[20] corrects perspective distortion by calculating three vanishing points to detect three planes. Specifically, first, edges are detected and line segments are fitted to the edges in an input image. Three vanishing points are then detected from the line segments by a RANSAC-based voting approach, and simultaneously each line segment is grouped according to which vanishing point the line segment is related with. It should be noted that line segments from two kinds of groups usually exist on a plane because a pair of two vanishing points makes a plane. Therefore, the group to which line segments around the missing region belong is checked, and the line passing through the two vanishing points related to the most dominant two kinds of line segments is calculated as follows:

$$ax + by + c = 0, \tag{6}$$

where $c \neq 0$. From a, b, and c, pure projective parameters l_1 and l_2 in Eq. (1) are calculated as follows.

$$l_1 = \frac{a}{c}, \quad l_2 = \frac{b}{c}. \tag{7}$$

For this method to work well, target scenes must contain some pairs of parallel lines such as window frames, tile seams and lines on streets. In the application of image inpainting, missing regions have little affect on the detection of line segments. However, it becomes more difficult to detect useful line segments as missing regions become larger.

4. Resolution Adjustment

This section describes a method to determine scaling parameters s_x and s_y in Eq. (1). When image inpainting is applied to a transformed image, if the resolution of the image is too low, the final result becomes blurry after transforming the result on the transformed image back to the original image. If the resolution of the image is too high, image inpainting works inefficiently. To avoid the problem, we set the resolution of the corrected image so that the resolution of the missing region in the corrected image becomes consistent with the highest resolution of missing region Ω in the input image. In particular, we calculate vertical and horizontal magnifications s_x and s_y as follows:

$$\begin{cases} s_x = \min\left(\max_{(x,y)\in\Omega} \frac{1}{||H'(x+1,y)-H'(x,y)||_2},\ S\right) \\ s_y = \min\left(\max_{(x,y)\in\Omega} \frac{1}{||H'(x,y+1)-H'(x,y)||_2},\ S\right) \end{cases}, \tag{8}$$

where $||\ ||_2$ represents the ℓ_2-norm, $H'(x,y)$ represents the coordinate on the corrected image that is obtained by transforming (x,y) with the following homography matrix H'.

$$H' = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ l_1 & l_2 & 1 \end{bmatrix}, \tag{9}$$

where l_1 and l_2 are obtained by a perspective correction method. We use upper limit S for the magnifications because it is inefficient to inpaint the corrected image if the corrected image is overstretched as shown in Fig. 5.

5. Analysis through Experiments

In this section, we analyze the effect of perspective

(a) Input image with a missing region (red) (b) Corrected image

Fig. 5 Example of overstretched image.

correction on image inpainting through experiments using simulated and real images *. In experiments, we used the following five methods.

(A) Image inpainting[16] with texel size-based perspective correction[21].

(B) Image inpainting[16] with matrix rank-based perspective correction[22].

(C) Image inpainting[16] with vanishing point-based perspective correction[20].

(D) Image inpainting[16] without perspective correction

(E) Image inpainting[20] considering perspective distortion using vanishing points.

For experiments, we implemented method A, and we used the programs available from the websites*,** for methods B and E. For method C, we extracted the part for estimating vanishing points from the program for method E. We set upper limit S as 5 in methods A, B and C. As the difference between methods C and E both using vanishing points, method C transforms the entire image and applies image inpainting, while method E transforms each local texture patch when calculating similarity during image inpainting.

As for processing time, it took about 60, 10 and several seconds for texel size-based, matrix rank-based and vanishing point-based methods to correct the perspective distortion, respectively. It took several seconds for inpainting method[16] to finish inpainting with a PC (Core i7 3.4GHz of CPU and 8.0 GB of memory).

For evaluation, we requested subjects to give a score of 1 (bad) to 5 (good) to the inpainting results obtained from the methods, which were randomly placed so that the subjects cannot know the relationship between the methods and results. Here, it should be noted that when a perspective-corrected image whose width or height becomes more than ten times as long as that

* We show all results on http://yokoya.naist.jp/itemta15/
* http://perception.csl.illinois.edu/matrix-rank/tilt.html
** https://sites.google.com/site/jbhuang0604/publications/struct_completion

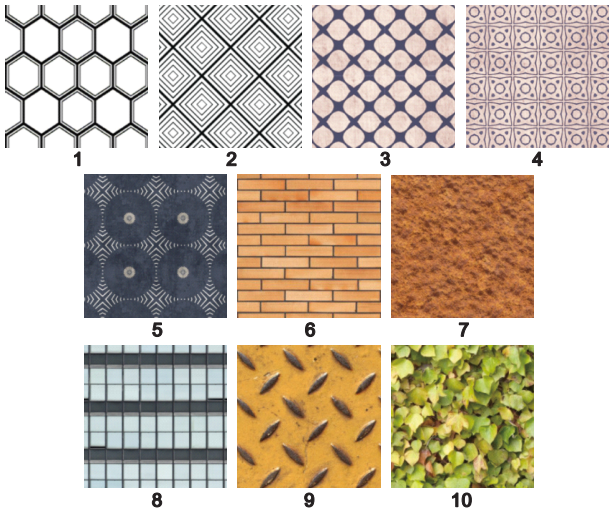


Fig. 6 Textures used for simulation.

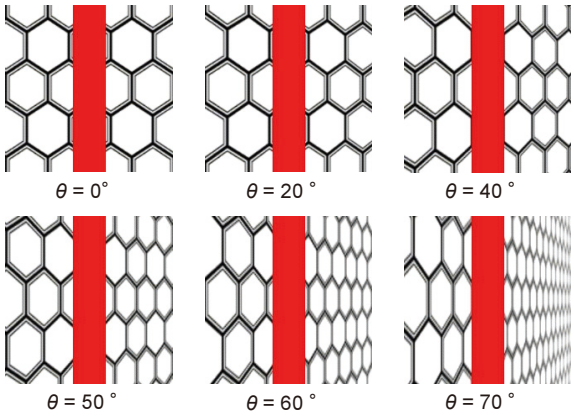


Fig. 7 Images distorted in simulation.

of the original input image in methods A, B and C, we regarded the perspective correction as failure because such an image is extremely distorted by transformation, and we did not apply image inpainting. As for method E, when the program stopped before outputting an inpainting result, we regarded the inpainting as failure.

5.1 Experiment Using Simulated Images

This section describes an experiment using simulated images to clarify the influence of the amount of perspective distortion and characteristics of texture patterns on inpainting results. In this experiment, we used 16 subjects, who were graduate students in Nara Institute of Science and Technology, and ten kinds of textures shown in Fig. 6. We generated distorted images with 200×200 pixels, like Fig. 7, by setting a virtual environment in which we kept a camera pose and a missing region on the image plane fixed, and a textured plane was rotated by θ degrees ($\theta = 0, 20, 40, 50, 60, 70$) as illustrated in Fig. 8.

Figure 9 shows the graph of average scores for ten

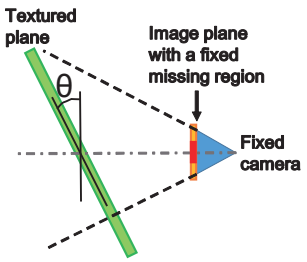


Fig. 8 Simulation environment.

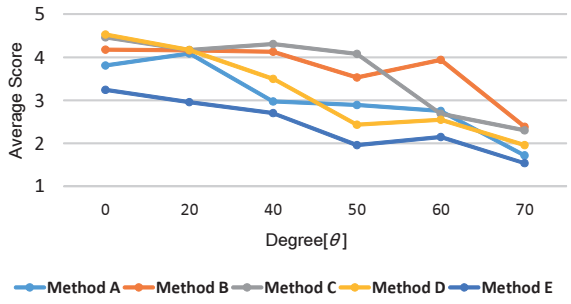
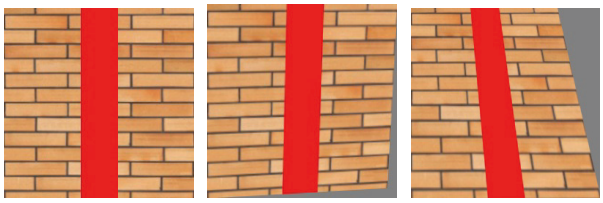


Fig. 9 Evaluation scores for ten resultant images with each rotation degree obtained by five methods.



(a) Texture 6 ($\theta = 0°$) (b) Result by method A (c) Result by method C

Fig. 10 Results obtained by applying perspective correction to an undistorted image.

resultant images with each rotation degree obtained by methods A to E. When distortion does not occur ($\theta = 0°$), method D, which does not correct perspective distortion, obtained the best score. Methods A, B, C and E tend to transform the image even if an input image has no distortion as shown in Fig. 10, resulting in worse inpainting results than method D. Although evaluation scores for all the methods tend to decrease as the amount of degree increases, the scores for method D decreases more rapidly than other methods. From this fact, we confirmed that automatic perspective correction is effective on image inpainting. Especially, methods B and C got better scores than others. Although method E also considers the perspective distortion, the method often gave blurry images, which resulted in the worst scores. We consider this is because image inpainting methods are different and the framework of embedding the perspective correction in an energy function in method E may tend to generate blurry textures.

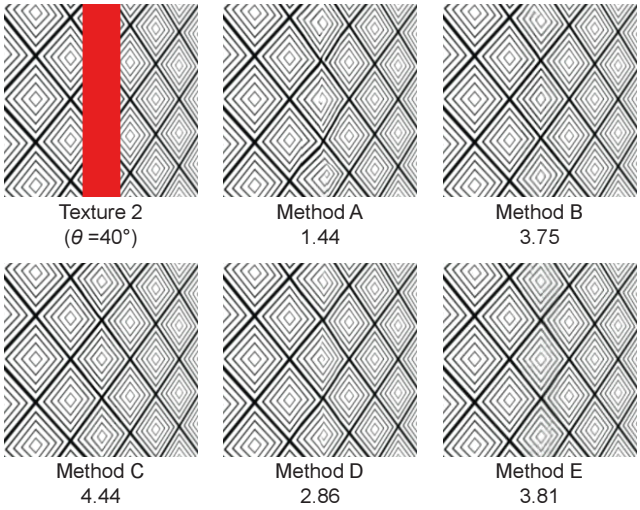


Fig. 11 Results for texture 2 with $\theta = 40°$.

Next, we analyze the results in detail by giving some examples which contain different characteristics and for which different methods obtained good scores. Figures 11, 12, and 13 show the resultant images and scores for texture 2 with $\theta = 40°$, texture 3 with $\theta = 60°$, and texture 10 with $\theta = 40°$ obtained by methods A to E, respectively. We analyze the results in detail in the following.

[Texture 2 with $\theta = 40°$ (Fig. 11)]

Texture 2 has a regular pattern with many parallel line segments. Since it is easy to detect line segments in the image, methods C and E, which use vanishing points, obtained good inpainting results. On the other hand, method A could not properly correct perspective distortion using texel sizes because of the slightly complex design, resulting in some unnatural patterns in the inpainting result.

[Texture 3 with $\theta = 60°$ (Fig. 12)]

In texture 3, a simple unique pattern is lined. Method A obtained a good result because it is easy to extract the texel size. Method B also could produce a good inpainting result. However, methods C and E gave some implausible patterns because the texel pattern has a curve and they could not find vanishing points properly.

[Texture 10 with $\theta = 40°$ (Fig. 13)]

Texture 10 has random texture patterns. Since each method failed to correct perspective distortion properly, images transformed by methods A and C were regarded as failure and methods B and E produced some blurs in the inpainting results. Although method D did not consider the perspective correction, it gave a better result. From the fact, we confirmed that existing methods for

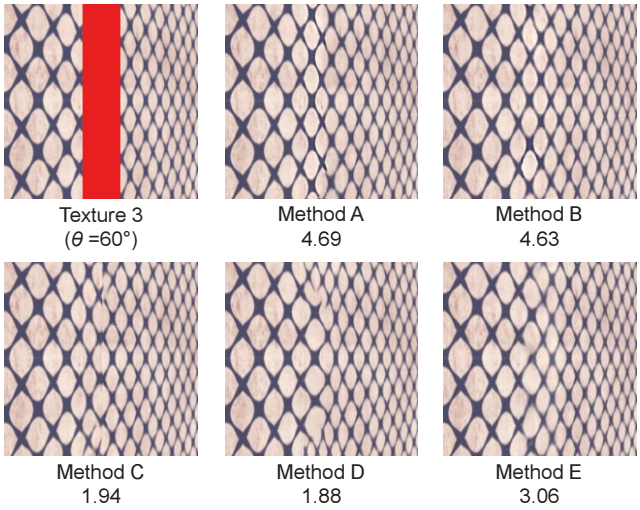


Fig. 12 Results for texture 3 with $\theta = 60°$.

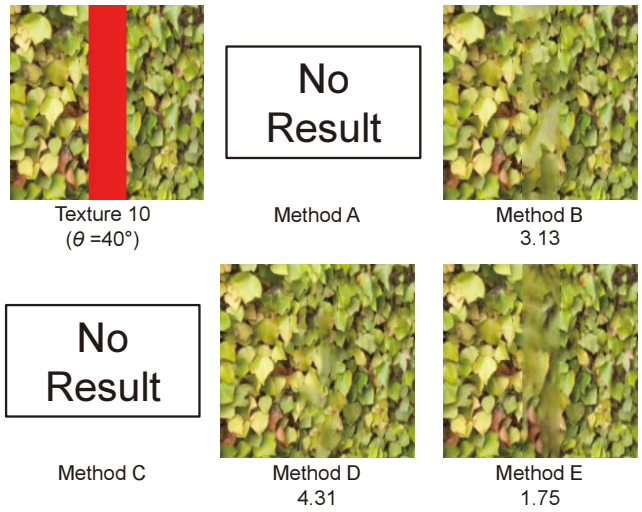


Fig. 13 Results for texture 10 with $\theta = 40°$.

perspective correction are not suitable to images with random patterns.

From the results, we can confirm that the accuracy of perspective correction by each method depends on characteristics of texture patterns and the amount of perspective distortion, and the methods that obtain better inpainting results than others also differ according to the characteristics and the amount of perspective distortion. As we can imagine from the principles of the methods, methods A and C tends to obtain good results for images with relatively simple texture patterns such that texel sizes can be easily extracted, and those with many parallel line segments such that vanishing points can be accurately detected, respectively. For random textures, in which it is difficult to extract texel sizes of objects with originally the same sizes and parallel line segments, inpainting without perspective collection often gives better results. We could not find the tendency of results obtained by method B from the experiments using simlulated images.

5.2 Experiment Using Real Images

This section describes an experiment using real photographs. In this experiment, we used 30 images shown in Fig. 14 and the inpainting results obtained by the five methods were evaluated by asking 14 subjects, who were also graduate students in Nara Institute of Science and Technology, to give scores of 1 (bad) to 5 (good). Note that we selected the images for which automatic correction methods can possibly achieve perspective correction by considering the principles of methods.

First, we discuss the results by picking four examples which contain different characteristics of textures as shown in Figs. 15 to 18. These figures show input image in (a), perspective-corrected images by methods A to C in (b) to (d), and resultant images by methods A to E and their scores in (e) to (i).

Image 28 in Fig. 15 captures wire meshes, which consists of almost the same patterns. For this image, method A successfully corrected the perspective distortion by properly calculating texel sizes and provided a good resultant image as shown in Fig. 15(e). On the other hand, method B only slightly corrected the perspective distortion and method C failed. The meshes are apparently straight lines but actually a little waved. Therefore, it is difficult to find such parallel line segments that are used for detecting vanishing points accurately. As a result, the resultant image in Fig. 15(f) includes more unconnected wires than that in Fig. 15(e). However, compared to the result without no perspective correction shown in Fig. 15(h), we confirmed that slight correction is effective to some extent.

Image 15 in Fig. 16 captures windows on a building, which are placed at even intervals. For this image, methods B, C and E obtained good scores as shown in Figs. 16(f), (g) and (i). As for method B, since the missing region is thin, it did not influence the perspective correction. As for methods C and E using vanishing points, window frames lead to many parallel line segments, resulting in finding good vanishing points. On the other hand, method A could not correct the perspective distortion properly because each window has a three-dimensional structure and the occlusions disturb extracting each window size. As a result, the inpainting produced blurs in the missing region as shown in Fig. 16(e) because the image has no appropriate size of exemplars around the missing region. The failure of correction does not improve the inpainting quality.

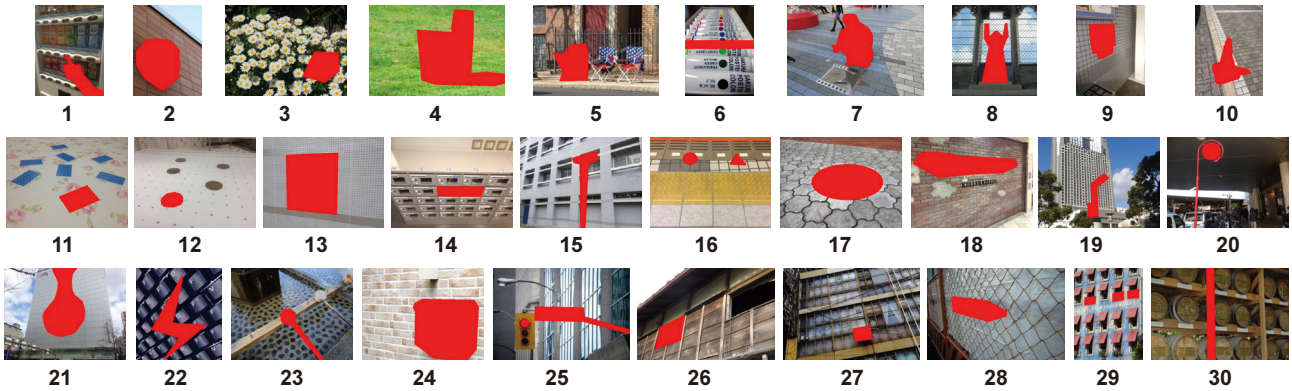


Fig. 14 Real images used for experiment.

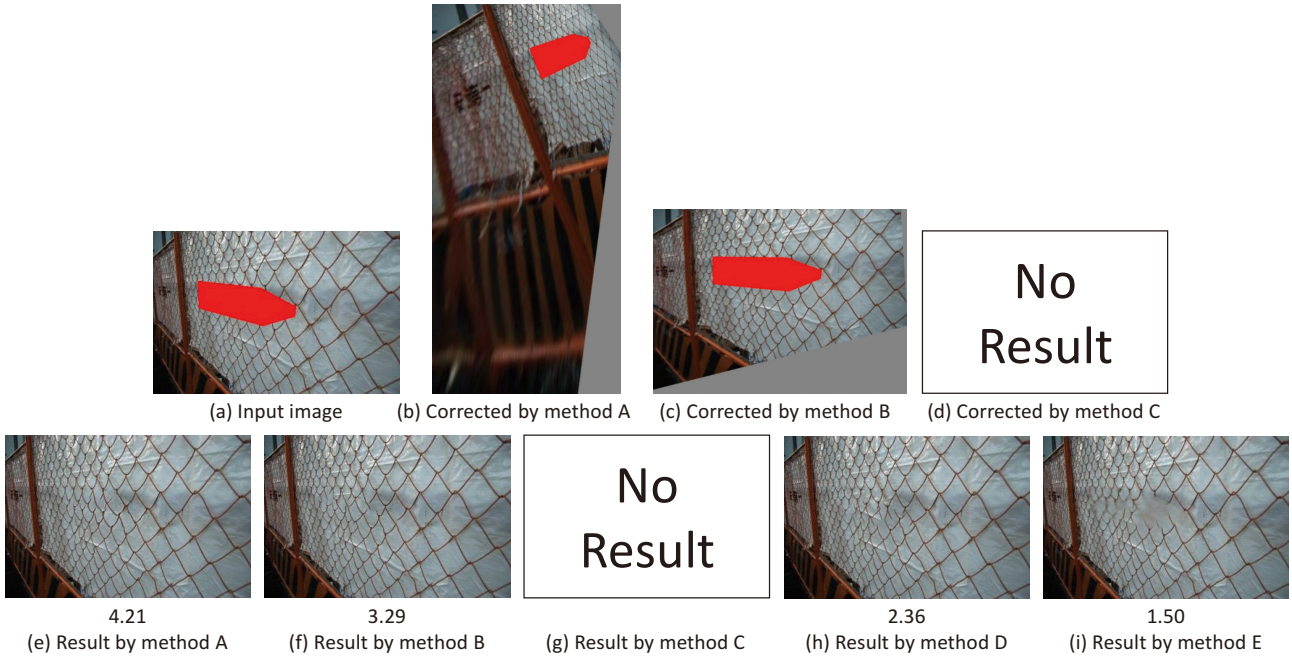


Fig. 15 Results of image 28 for which method A obtained a score of 4 or more.

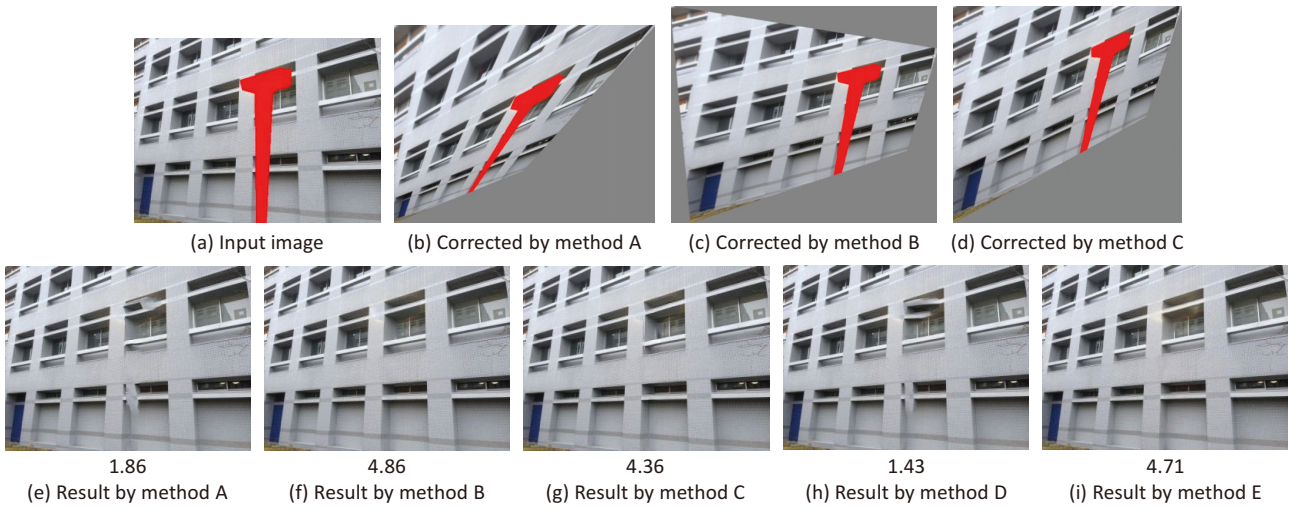


Fig. 16 Results of image 15 for which methods B, C and E obtained scores of 4 or more.

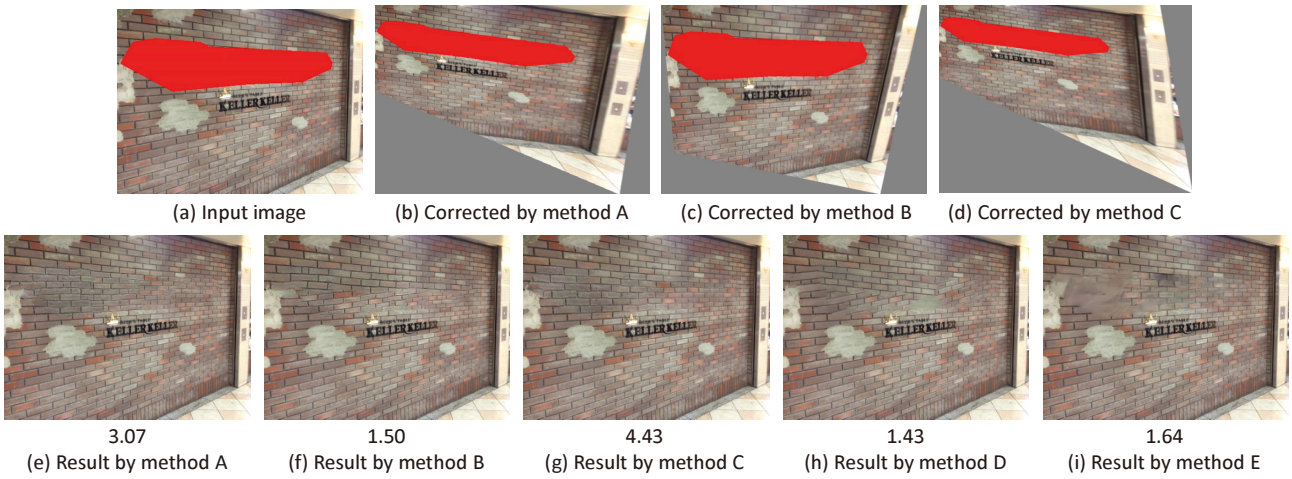


Fig. 17 Results of image 18 for which method C obtained a score of 4 or more.

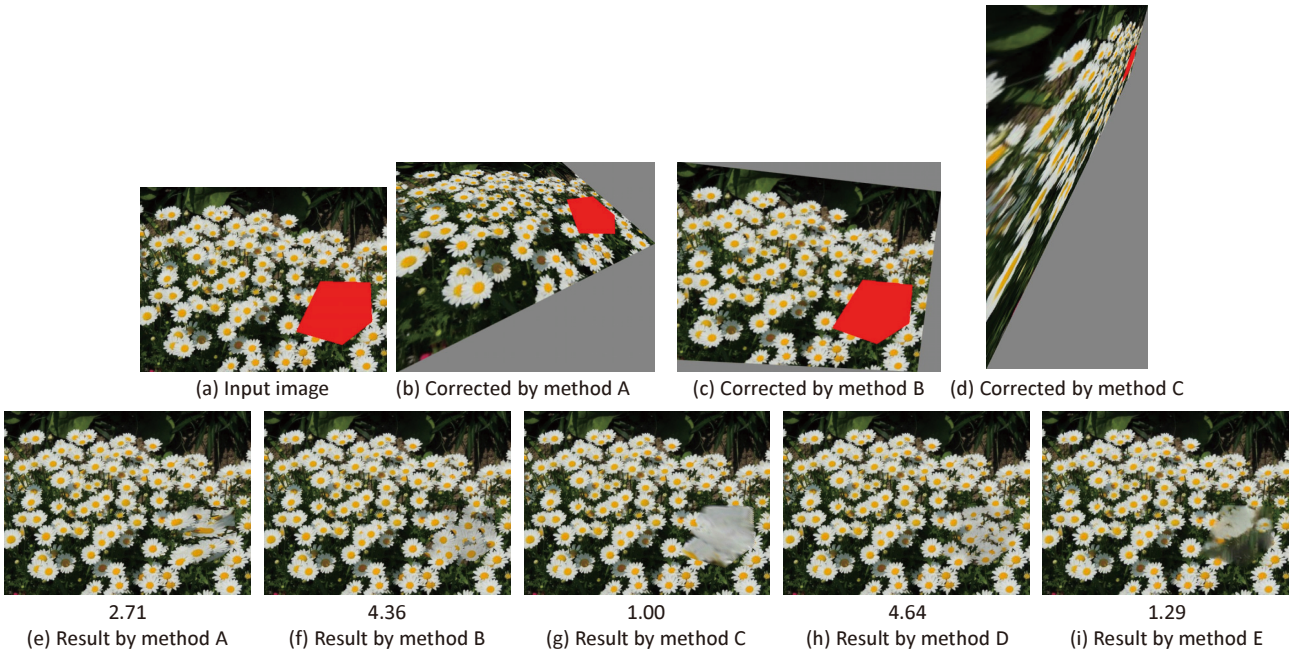


Fig. 18 Results of image 3 for which methods B and D obtained a score of 4 or more.

Image 18 in Fig. 17 captures a brick wall, which has many parallel line segments. For this image, method C obtained a good score as shown in Fig. 17(g) because it detected vanishing points from the line segments and corrected the perspective distortion most successfully. Method A also corrected the perspective distortion well but the remaining small distortion caused texture distortion in the resultant image in Fig. 17(e). Since the input image has a relatively large missing region, it was difficult for method B to achieve successful perspective correction. As a result, the inpainting produced brick textures in the missing region whose directions are inconsistent with those of the surrounding textures.

Image 3 in Fig. 18 captures flowers, which are apparently almost the same but the shapes and scales differ from one another. Since the shapes of flowers are not simple and the flowers are overlapping each other, method A could not find appropriate texel sizes and provided unnatural transformation. Method C also transformed the image badly because there are almost no parallel line segments. As a result, methods A and C produced unnaturally stretched textures and blurs as shown in Figs. 18(e) and (g), respectively. On the other hand, method B did not transform the image so that the rank can be minimized and outputted a good result. For such an image with random textures, method D, which does not consider the perspective correction, gave the best score.

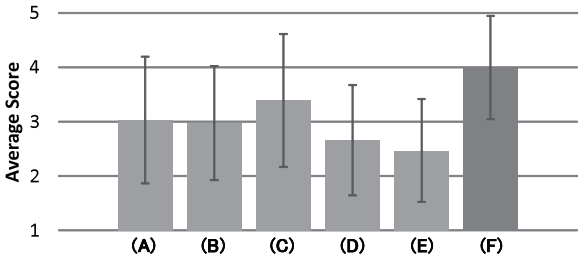


Fig. 19 Average scores and standard deviations for resultant images.

Table 1 Images for which each method obtained scores of 4 or more.

Method	Image No.	No. of images
A	4,12,16,19,21,22,24,28	8
B	3,12,14,15,22,29	6
C	4,8,10,14,15,16,18,21,22,29	10
D	1,3,4,13	4
E	15	1

Next, we discuss the overall results. From the graphs of methods A to E shown in Fig. 19, which shows the average scores and standard deviations for 30 resultant images, and Table 1, which shows images for which each method obtained scores of 4 or more, we can know that method C obtained higher scores and more good images than others. However, the difference of scores is not large. On the other hand, since the standard deviation is large and the image numbers largely differs according to method, we can know that the quality of resultant images by each method largely varies according to characteristics of images.

In summary, the quality of resultant images largely depends on the compatibility between methods and images, and the characteristic of each method is as follows.

Method A is effective for images with simple regular textures in which texel sizes are accurately calculated.

Method B is effective when missing regions are relatively small.

Method C is effective for images with parallel and straight line segments.

Method D is often more effective than the methods that try to correct perspective distortion, especially for random texture patterns.

Method E tends to output blurry textures although the characteristics is almost the same as with method C.

When the perspective correction succeeds regardless of correction method, the inpainting results also become good in many cases. On the other hand, when the perspective correction fails, the inpainting usually produces blurs and unnatural textures whose scales and directions are inconsistent with those of the textures around missing regions because textures with appropriate scales and directions may not exist in the badly transformed image. This often happens especially for images with random textures. In such a case, applying image inpainting to the non-corrected image is often effective.

5.3 Advantage Using Multiple Perspective Correction Criteria

This section demonstrates the advantage of using multiple methods for perspective correction for image inpainting from the evaluation results obtained using real images. Here, we assume method F which selects the image with the best score from the four resultant images obtained by methods A, B, C and D. The rightmost graph in Fig. 19 shows the average score and the standard deviation of 30 images for method F. From the results, the significant difference between method F and each of other methods was observed using the t-test with a 5% significant level. As a result, currently the use of multiple perspective correction methods is effective for obtaining more good inpainting results than using one of the state-of-the-art methods like method E[20]. Therefore, the inpainting system in which a user selects one of the results given with multiple criteria for perspective correction is a desirable solution to obtain good results for many kinds of images for now.

5.4 Discussion about Using Different Inpainting Methods

This section discusses results obtained by different exemplar-based inpainting methods for the three images in Figs. 16 to 18 in Section 5.2 to confirm the correlation between method[16] and another inpainting method. Here, we employed "Content-Aware Fill" function in Adobe Photoshop CS5, which was developed based on inpainting method[14] with fast searching method PatchMatch[23] and is the most popular commercial software for inpainting.

Fig. 20 shows results obtained by applying Photoshop CS5 to images corrected by methods A, B and C and non-corrected images. We compare the results in (a), (b) (c) and (d) in Fig. 20 with those in (e), (f), (g) and (h) in Figs. 16 to 18.

As for image 15, the resultant images are not good even if the correction succeeds as shown in the top row of Figs. 20(b) and (c). From the fact, we confirmed that successful perspective correction does not always lead to good inpainting results depending on inpainting methods. When the perspective correction failed, the

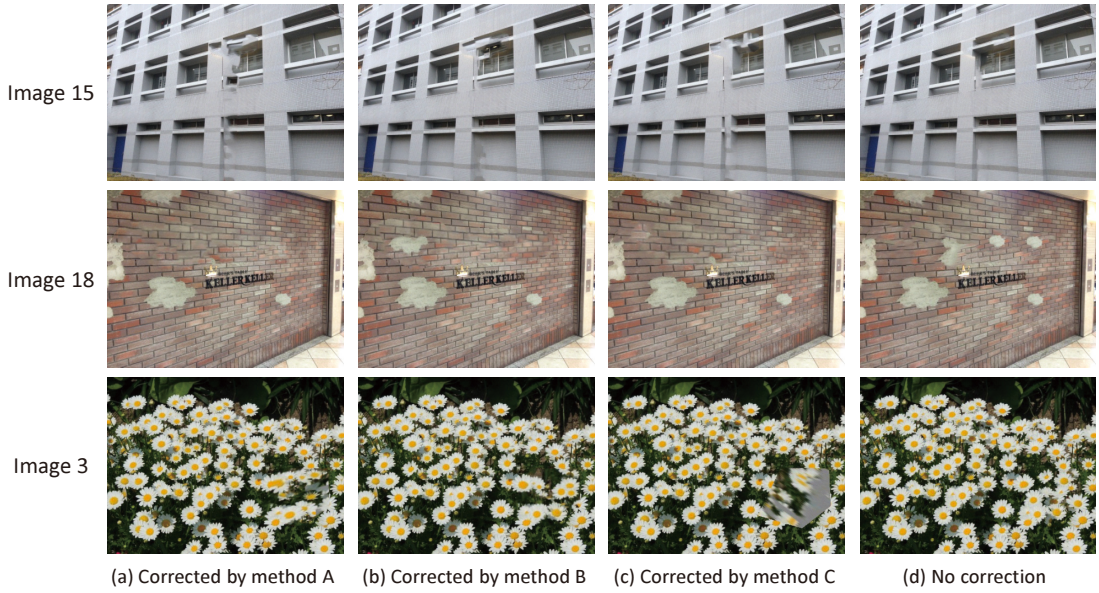


Fig. 20 Results obtained by photoshop CS5.

two inpainting methods did not produce good results in common.

As for image 18, when the perspective correction succeeded, the inpainting produced consistent textures with appropriate sizes and directions as shown in the middle row of Figs. 20(a) and (c) as similar to those in Figs. 17 (e) and (g). When the correction failed or the correction was not conducted, the resultant images contain textures that are inconsistent with the surrounding textures as shown in Figs. 20(b) and (d). The produced textures are quite similar to those in Figs. 17 (f) and (h).

As for image 3, the inpainting produced good results when the correction succeeded or the correction was not conducted as shown in the bottom row of Figs. 20(b) and (d). When the correction failed, unnatural stretched textures and blurs were produced as shown in Figs. 20(a) and (c). The tendency of results is almost the same with that of results by inpainting method[16] as shown in Fig. 18.

From the comparison of results, we confirmed the similar tendency between inpainting method[16] and Photoshop CS5.

6. Conclusion

In this paper, we have analyzed the effect of automatic perspective correction on exemplar-based image inpainting. In experiments, we employed the existing exemplar-based inpainting method with multiple automatic perspective correction methods for simulated and real images. The experiment using simulated images examined the influence of the amount of perspective distortion and characteristics of texture patterns on inpainting results. The experiment using real images examined the relationship between the methods and characteristics of images. From the experimental results, we confirmed that the influence varies depending on the compatibility between methods and images. In addition, we demonstrated the effectiveness of using multiple criteria for perspective correction for image inpainting.

Although we employed an exemplar-based inpainting approach in this study, we should consider the effect of perspective correction on other inpainting approaches in future work since a variety of effective inpainting methods that do not use patch similarity also have been proposed such as pixel-wise method[5], method using subspace[6] and method using neural network[7].

In addition, since this paper handled scenes in which a single homography transformation is effective, we should consider three-dimensional scenes as well as develop an inpainting method applicable to such a scene in future work. For example, image segmentation and clustering texel sizes could be useful as keys to handle a three-dimensional scene in a single image.

Acknowledgment This work was partially supported by Grants-in-Aid for Scientific Research Nos. 23240024, 25540086 and 15K16039 from the Japan Society for the Promotion of Science (JSPS).

References

1)M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. In *Proc. ACM SIGGRAPH*, pp. 417–424, 2000.
2)T. Chan and J. Shen. Non-texture inpainting by curvature-driven diffusions (CDD). *Journal of Visual Communication and Image Representation*, Vol. 12, No. 4, pp. 436–449, 2001.
3)A. Levin, A. Zomet, and Y. Weiss. Learning how to inpaint from global image statistics. In *Proc. IEEE Int. Conf. on Computer Vision*, pp. 305–312, 2003.
4)T. Amano. Correlation based image defect detection. In *Proc. IAPR Int. Conf. on Pattern Recognition*, pp. 163–166, 2006.
5)Y. Pritch, E. Kav-Venaki, and S. Peleg. Shift-map image editing. In *Proc. IEEE Int. Conf. on Computer Vision*, pp. 151–158, 2009.
6)T. Ogawa and M. Haseyama. Missing intensity interpolation using a kernel pca-based pocs algorithm and its applications. *IEEE Trans. on Image Processing*, Vol. 20, No. 2, pp. 417–432, 2011.
7)J. Xie, L. Xu, and E. Chen. Image denoising and inpainting with deep neural networks. In *Proc. Advances in Neural Information Processing Systems*, pp. 341–349, 2012.
8)T. Ogawa and M. Haseyama. Missing texture reconstruction method based on error reduction algorithm using fourier transform magnitude estimation scheme. *IEEE Trans. on Image Processing*, Vol. 22, No. 3, pp. 1252–1257, 2013.
9)I. Drori, D. Cohen-Or, and H. Yeshurun. Fragment-based image completion. In *Proc. ACM SIGGRAPH*, pp. 303–312, 2003.
10)A. Criminisi, P. Pérez, and K. Toyama. Region filling and object removal by exemplar-based image inpainting. *IEEE Trans. on Image Processing*, Vol. 13, No. 9, pp. 1200–1212, 2004.
11)W-H. Cheng, C-W. Hsieh, S-K. Lin, C-W. Wang, and J-L Wu. Robust algorithm for exemplar-based image inpainting. In *Proc. Int. Conf. on Computer Graphics, Imaging and Visualization*, pp. 64–69, 2005.
12)Z. Xu and J. Sun. Image inpainting by patch propagation using patch sparsity. *IEEE Trans. on Image Processing*, Vol. 19, No. 5, pp. 1153–1165, 2010.
13)N. Komodakis and G. Tziritas. Image completion using global optimization. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 442–452, 2006.
14)Y. Wexler, E. Shechtman, and M. Irani. Space-time completion of video. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 29, No. 3, pp. 463–476, 2007.
15)N. Kawai, T. Sato, and N. Yokoya. Image inpainting considering brightness change and spatial locality of textures and its evaluation. In *Proc. Pacific-Rim Symp. on Image and Video Technology*, pp. 271–282, 2009.
16)N. Kawai and N. Yokoya. Image inpainting considering symmetric patterns. In *Proc. IAPR Int. Conf. on Pattern Recognition*, pp. 2744–2747, 2012.
17)S. Darabi, E. Shechtman, C. Barnes, D. B. Goldman, and P. Sen. Image melding: Combining inconsistent images using patch-based synthesis. *ACM Trans. on Graphics*, Vol. 31, No. 4, pp. 82:1–82:10, 2012.
18)D. Pavić, V. Schőnefeld, and L. Kobbelt. Interactive image completion with perspective correction. *The Visual Computer*, Vol. 22, No. 4, pp. 671–681, 2006.
19)J. Huang, J. Kopf, N. Ahuja, and S. B. Kang. Transformation guided image completion. In *Proc. IEEE Int. Conf. on Computational Photography*, pp. 1–9, 2013.
20)J. Huang, S. B. Kang, N. Ahuja, and J. Kopf. Image completion using planar structure guidance. *ACM Trans. on Graphics*, Vol. 33, No. 4, 2014.
21)D. Aiger, D. Cohen-Or, and Niloy J. Mitra. Repetition maximization based texture rectification. *Computer Graphics Forum*, Vol. 31, No. 2, pp. 439–448, 2012.
22)Z. Zhang, A. Ganesh, X. Liang, and Y. Ma. TILT: Transform invariant low-rank textures. *Int. Journal of Computer Vision*, Vol. 99, No. 1, pp. 1–24, 2012.
23)C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Trans. on Graphics*, Vol. 28, No. 3, pp. 24:1–24:11, 2009.

Hiroto Sasao received his B.E. degree from University of Fukui in 2013. He received his M.E. degree in information science from Nara Institute of Science and Technology (NAIST) in 2015. He has been working for Hitachi Kokusai Electric Inc. from 2015.

Norihiko Kawai received his B.E. degree in informatics and mathematical science from Kyoto University in 2005. He received his M.E. and Ph.D. degrees in information science from Nara Institute of Science and Technology (NAIST) in 2007 and 2010, respectively. He was a research fellow of the Japan Society for the Promotion of Science and a postdoctoral fellow at the University of California at Berkeley in 2010-2011. He has been an assistant professor at NAIST since 2011. He is a member of IEEE, IEICE, IPSJ and VRSJ.

Tomokazu Sato received his B.E. degree in computer and system science from Osaka Prefecture University in 1999. He received his M.E. and Ph.D. degrees in information science from Nara Institute of Science and Technology (NAIST) in 2001 and 2003, respectively. He was an assistant professor at NAIST in 2003-2011. He was a visiting researcher at Czech Technical University in Prague in 2010-2011. He has been an associate professor at NAIST since 2011. He is a member of IEEE, IEICE, IPSJ, VRSJ and ITE.

Naokazu Yokoya received his B.E., M.E., and Ph.D. degrees in information and computer sciences from Osaka University in 1974, 1976, and 1979, respectively. He joined Electrotechnical Laboratory (ETL) of the Ministry of International Trade and Industry in 1979. He was a visiting professor at McGill University in Montreal in 1986-87 and has been a professor at Nara Institute of Science and Technology (NAIST) since 1992. He has also been a vice president at NAIST since April 2013. He is a fellow of IPSJ, IEICE and VRSJ and a member of IEEE, ACM SIGGRAPH, JSAI, JCSS and ITE.