From Image Inpainting to Diminished Reality

Norihiko Kawai, Tomokazu Sato, and Naokazu Yokoya

Graduate School of Information Science, Nara Institute of Science and Technology
8916-5 Takayama, Ikoma, Nara 630-0192, Japan
{norihi-k,tomoka-s,yokoya}@is.naist.jp

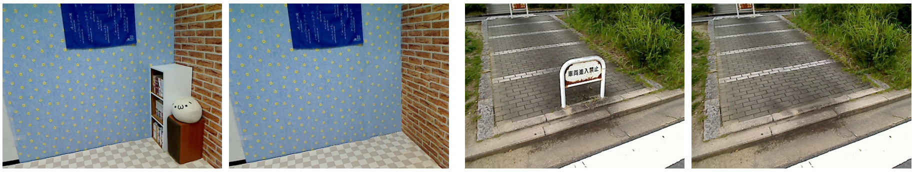
Abstract. Image inpainting, which removes undesired objects in a static image and fills in the missing regions with plausible textures, has been developed in the research fields of image processing. On the other hand, Diminished Reality (DR), which visually removes real objects from video images by filling in the missing regions with background textures in real time, is one of the growing topics in Virtual/Mixed Reality, and considered as the opposite of Augmented Reality. In this paper, we introduce the state-of-the-art of image inpainting methods and how to apply the image inpainting to diminished reality.

Keywords: image inpainting, diminished reality, augmented reality.

1 Introduction

Image inpainting, which removes undesired objects in a static image and fills in the missing regions with plausible textures, has been developed in the research fields of image processing. On the other hand, Diminished Reality (DR), which visually removes real objects from video images by filling in the missing regions with background textures in real time, is one of the growing topics in Virtual/Mixed Reality, and considered as the opposite of Augmented Reality. Diminished reality can be used for various applications. For example, some pieces of furniture may be removed to simulate different arrangements (Fig. 1(a)), signboards can be removed for landscape simulations (Fig. 1(b)), and augmented reality (AR) markers can be hidden to achieve seamless fusion between virtual objects and the real world [1–3]. Diminished reality methods can be classified into two categories: One uses actual background images by capturing them in advance or with multiple cameras, and the other generates a plausible background by applying an image inpainting technique. For scenes in which the actual background of a target object cannot be observed, or for cases where it is burdensome for users to capture the background, we can employ the latter method. In this paper, we focus on the image inpainting-based diminished reality approach, and introduce the state-of-the-art of image inpainting methods and how to apply the image inpainting to diminished reality. In addition, we briefly introduce our recent diminished reality method and its results [4].

R. Shumaker and S. Lackey (Eds.): VAMR 2014, Part I, LNCS 8525, pp. 363–374, 2014.



(a) Furniture removal

(b) Signboard removal

Fig. 1. Example applications of diminished reality. Images on the left are inputs, and those on the right are our results.

2 Image Inpainting for Removing Objects

In this section, we introduce the state-of-the-art image inpainting methods. We then introduce searching methods for speeding up image inpainting.

2.1 Image Inpainting Methods

Image inpainting (also referred to as image completion) methods can be largely classified into two categories: One uses information only around the target region, and the other uses the similarity of textures. The former approach fills in the missing regions by calculating pixel values considering the continuity of pixel intensity from the boundary of the missing region assuming thet neighbor pixels have similar pixel values. As the representative method, Bertalmio et al. [5] propagate colors along edges by using partial differential equations. This type of method is effective for small image gaps like scratches in a photograph. However, the resultant images easily become unclear when the missing regions are large because the methods cannot generate complex textures in principle. Therefore, the latter approach has been intensively developed these days.

This approach uses textures in an image as exemplars based on the assumption that textures appropriate for missing regions are similar to those in the remainder of the image. The methods in this approach can be classified into two categories. One is based on successive texture copy and the other on global optimization. In the former approach, the application of texture synthesis technique to image completion was originated by Efros et al. [6]. In this method, texture is successively copied to the boundary of the missing regions. Although this method can generate complex textures in the missing regions, the quality of synthesized texture largely depends on the order of copy. For this problem, in order to maker more plausible textures, the order of texture copy has been determined with some criteria (e.g., the number of fixed pixels in a patch and strength of an edge in [7]). Nevertheless, these methods still have the problem that a discontinuous texture tends to be generated by the greedy fill-in order.

In order to settle this, global optimization-based methods have been proposed. As the representative method in this approach, Wexler et al. [8] generate optimal textures in missing regions by minimizing an objective function based

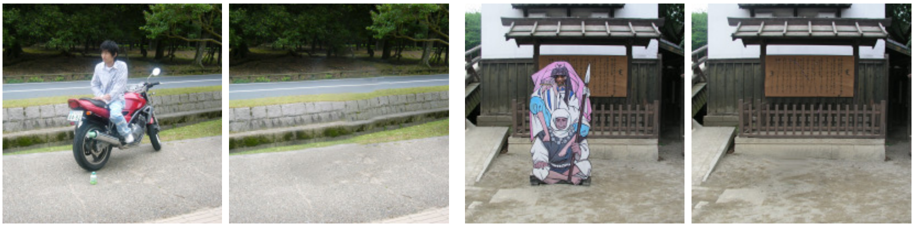


Fig. 2. Results of our image inpainting method [10]

on pattern similarity between the missing region and the reminder of the image. Specifically, the objective function is minimized by iterating two processes: searching for a similar pattern in the reminder of the image, and updating pixel values in the missing regions. Although this method can generate complex and good textures for many images, unnatural textures are still generated due to the paucity of available samples in the image. To increase available samples, there have already been some attempts in terms of photometric and geometric expansion of patterns. For example, our previous methods in [9, 10] allow brightness transformation of texture patterns to utilize patterns with the same geometry but different brightness, and use symmetric patterns. Fig. 2 shows example results of our method. Darabi et al. [11] use screened poisson to adjust color and symmetric, rotating, scaling patterns. However, it is difficult to automatically and appropriately estimate parameters of geometric transformation because various changes in texture patterns exist in ordinary photographs. Therefore, some methods are proposed for dealing with various changes in geometric patterns with manual interactions [12, 13].

2.2 Searching Method for Speeding Up Image Inpainting

In the exemplar-based method mentioned above, it takes much time to exhaustively search for similar patterns. For this problem, an approximate nearest neighbor search algorithm “PatchMatch” has been proposed [14]. This method propagates pixel positions of similar patterns when we make correspondences between pixels in the missing region and the reminder of the image pixel by pixel by raster scan. In addition, it also gives a good correspondence seed with random search. This method was improved for dealing with geometric changes in texture patterns as “Generalized PatchMatch” [15]. By using these search methods, the computational time of image inpainting is drastically improved from several tens of minutes to a few seconds.

3 Image Inpainting-Based Diminished Reality

In this section, we introduce six methods [1–4, 16, 17] in the field of image inpainting-based diminished reality. These methods basically assume that target objects are fixed in the 3D environment. A target object in research [1–3] is an AR marker, and the others target general objects. We review these methods in terms of four factors: (1) real-time processing, (2) the temporal coherence of textures, (3) the quality of image inpainting, and (4) the determination of mask regions in which foreground textures are to be replaced with background ones.

3.1 Real-Time Processing

As mentioned in Section 2, it still takes at least a few seconds for inpainting methods to fill in missing regions. Therefore, just applying an image inpainting method to each frame cannot remove objects in real time. To overcome the problem, three approaches have been proposed. One uses a very simple approach, and one alters a conventional image inpainting method to reduce the computational cost, and the other employs a semi-dynamic approach.

As regards a simple approach, Siltanen [1] mixed several specific pixel values around the target region. Although this method can rapidly generate textures, it is difficult to generate natural and complex textures using such a simple approach. As the second approach, Herling et al. [16] basically applied the the combination of methods in [18] and [14] with use of grayscale and reduction of resolution. Although the method achieved the real-time performance, the quality of inpainting decreased compared with the original inpainting method. Herling et al. [17] have also proposed a different diminished reality method by improving the energy function used in [16] using spatial cost to quicken the energy convergence. They also employed a parallel processing for searching process. By these, the quality is quite improved compared with their previous method. As the third approach, Korkalo et al. [2] and we [3, 4] proposed a semi-dynamic approach, which conducts two processes concurrently: image inpainting for a key frame, and the overlay of the inpainted texture with geometric and photometric adjustment for every frame. In this approach, though target objects are hidden with incomplete textures until the image inpainting finishes, advanced image inpainting methods can be applied. For example, in our paper [4], image inpainting method [10], which considers photometric and geometric pattern changes, was applied to diminished reality.

3.2 Temporal Coherence of Textures

For the temporal coherence of textures, the methods in [1, 16] basically generate textures for every frame. Therefore, they tend to cause unnatural changes in geometry between frames. Although Herling et al. [16] attempt to reduce texture flickering between frames by propagating patch correspondences in image inpainting from frame to frame, it is insufficient to achieve geometric consistency between frames taken with large camera motion. To overcome this problem, Herling et al. [17] improved their original method [16] by employing a homography, and thus determined the search areas in the next frame by assuming the background around the target object to be almost planar. Our previous method [3] also used a homography to synthesize an inpainted result when hiding an AR marker. These methods successfully preserve the temporal coherence of planar scenes. In addition, in our most recent paper [4], we approximated the background of the target objects by combining the local planes. For this, the scene around the target object is divided into multiple planes, whose number is automatically determined, and inpainted textures are successfully overlaid on the target object using multiple homographies by considering the estimated planes and camera-pose given by visual-SLAM (Simultaneous Localization and Mapping).

3.3 Quality of Image Inpainting

As mentioned above, Siltanen [1] mixed several specific pixel values for filling in missing regions. Therefore, the quality is insufficient if the textures in surrounding background are complex. To synthesize more natural textures for diminished reality, Herling et al. [16] applied an example-based image inpainting method [18], and they have also improved their energy function by considering spatial costs [17]. In their methods, the whole input image is searched for texture patterns that are similar to that around the target region, and pixel values in the target region are determined using similar patterns. Generally, although example-based inpainting methods yield good results, they produce unnatural results when an image's regular patterns have a distorted perspective.

To solve this problem, using the idea of perspective correction in image inpainting [12, 13], our previous method [3] corrected the perspective distortion using an AR marker, meaning that the size of regular texture patterns could be unified. Unlike the methods [12, 13] that requires manual interactions, we calculated a homography based on the assumption that an AR marker exists on a plane. In our most recent method [4], we have extended this idea for 3D scenes using 3D geometry to deal with perspective correction in 3D scenes. Specifically, we have generated multiple rectified images, one for each of the estimated planes. In addition to this, we have added a constraint to automatically limit the search region using structures around a target object, thus increasing the quality of inpainted textures.

3.4 Determination of Mask Region

Mask regions (those that include target objects) have to be found in every frame to ensure that the objects are removed from the image. Objects such as AR markers [1–3] can easily be tracked using software libraries (e.g., ARToolkit [19]), allowing the mask regions to be determined in real time. In other cases, various approaches are used to track the target objects and find the mask regions. For example, an active contour algorithm has been applied to detect and track objects [16], but this method is not robust for textured backgrounds. For this problem, several feature points that store the appearance of the image are set around the target objects, and the image is segmented into the mask and other regions in every frame by tracking the feature points [17]. Although this method

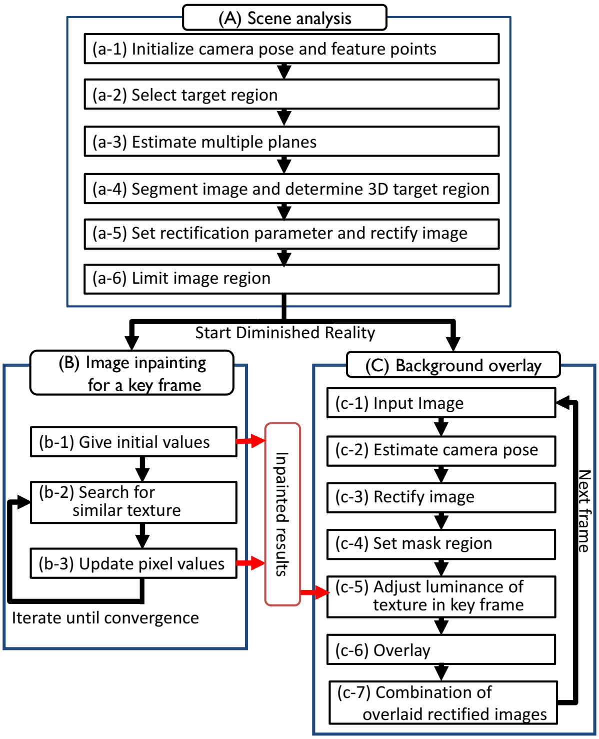

(A) Scene analysis
(a-1) Initialize camera pose and feature points
(a-2) Select target region
(a-3) Estimate multiple planes
(a-4) Segment image and determine 3D target region
(a-5) Set rectification parameter and rectify image
(a-6) Limit image region
Start Diminished Reality
(B) Image inpainting for a key frame
(b-1) Give initial values
(b-2) Search for similar texture
(b-3) Update pixel values
Iterate until convergence
Inpainted results
(C) Background overlay
(c-1) Input Image
(c-2) Estimate camera pose
(c-3) Rectify image
(c-4) Set mask region
(c-5) Adjust luminance of texture in key frame
(c-6) Overlay
(c-7) Combination of overlaid rectified images
Next frame

Fig. 3. Pipeline of our diminished reality technique

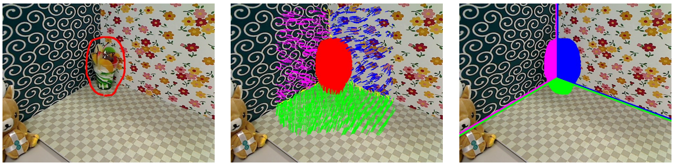
works well for scenes with textured backgrounds, it has the limitation that the entire object must always be in the video frame. In our method [4], we robustly determine the mask regions in all frames by tracking the 3D volume that includes target objects in 3D space using camera pose, rather than by tracking the object in 2D space. In this approach, the target objects do not always have to be in the video frame.

4 Diminished Reality Considering Background Structures

In this section, we briefly introduce our method [4], which achieve real-time diminished reality for 3D scenes by approximating the background by multiple local planes, and show experimental results of the method.

4.1 Pipeline of Our Diminished Reality Technique

Figure 3 shows the pipeline of our diminished reality technique. Our method first analyzes the target scene (A). Diminished reality is then achieved by a semi-dynamic approach that conducts two processes concurrently: example-based image inpainting for a key frame (B), and the overlay of the inpainted texture



(a) Enclosing (b) Labeling (c) Segmentation

Fig. 4. Scene analysis

for every frame (C). Although process (B) is not performed in real-time, users can start applications immediately by performing processes (B) and (C) concurrently. Within several seconds of starting process (B), we can experience diminished reality with a completely inpainted result. In the following, we describe processes (A), (B), and (C) in detail.

4.2 Scene Analysis

As pre-processing for diminished reality, the target scene is analyzed and the image is divided into multiple images to improve the quality of image inpainting. Specifically, the camera pose and 3D coordinates of feature points are first estimated by initializing visual-SLAM (a-1). A user then manually selects a region that includes target objects by enclosing the region, as shown in Fig. 4(a) (a-2). The frame when the user finishes enclosing the region is set as a key frame and is used for image inpainting in process (B). Next, feature points around the target region are picked up, and normal vectors of the feature points are calculated using the 3D coordinates of feature points. Each feature point is then classified into multiple groups based on mean-shift clustering using the normal vectors as shown in Fig. 4(b), and a plane is fitted to the feature points of each group using LMedS (Least Median of Squares) (a-3). All the fitted planes are projected onto the image plane, and each pixel is assigned to the plane that is nearest to the camera. According to this assignment, the whole image, including the missing region, is segmented as shown in Fig. 4(c). In addition, the 3D target region is generated from the 2D selected region using the fitted planes and feature points on the target object, so the 3D region must include the target object (a-4). Next, as shown in Fig. 5, the perspective distortion of the key frame is corrected by calculating a homography matrix for each plane as if each plane was captured by a camera in front of it, and the information for rectifying subsequent frames is stored (a-5). Finally, we limit the search region in which textures can be used as exemplars for inpainting in process (B) based on the segmented image (a-6).

4.3 Image Inpainting for Multiple Rectified Images

We apply an example-based image inpainting method to each rectified and limited image of the key frame. Our framework can adopt arbitrary example-based

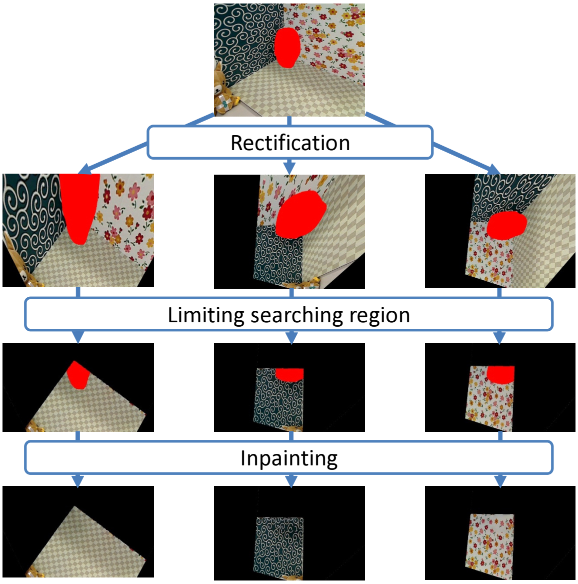


Fig. 5. Rectification and region limitation for inpainting

methods that use global optimization. After initializing parameters of the missing regions, e.g., the average value of boundary pixels, the inpainting method iterates two processes in order to minimize an energy function based on the similarity between missing regions and the remainder of the image. The first process searches for similar patterns (b-2), and the second updates pixel values in the missing regions (b-3). In each iteration, process (B) stores the tentative inpainted result in the memory shared with process (C). After the energy converges, the completely inpainted result is stored and used in process (C).

4.4 Real-Time Overlay of Inpainted Textures

In process (C), after capturing an image (c-1) and calculating a camera pose using visual-SLAM (c-2), a rectified image is generated for every plane using the current camera pose and information for rectification (c-3). On each rectified image, a mask region is then determined by projecting the 3D target region from the optical center of the current frame's camera onto each plane (c-4). Next, the mask regions are filled in using the texture in the rectified images of the key frame in which the object regions are inpainted. Because there is usually some difference in the luminance of the key frame and the current frame, we adjust the luminance of the key frame's texture (c-5). Here, we estimate luminance changes in the mask region from the changes in the surrounding region using rectified images between the key frame and the current frame. Finally, the texture of each rectified image of the key frame is overlaid on the mask region of each rectified image of the current frame. The rectified images are transformed to the original

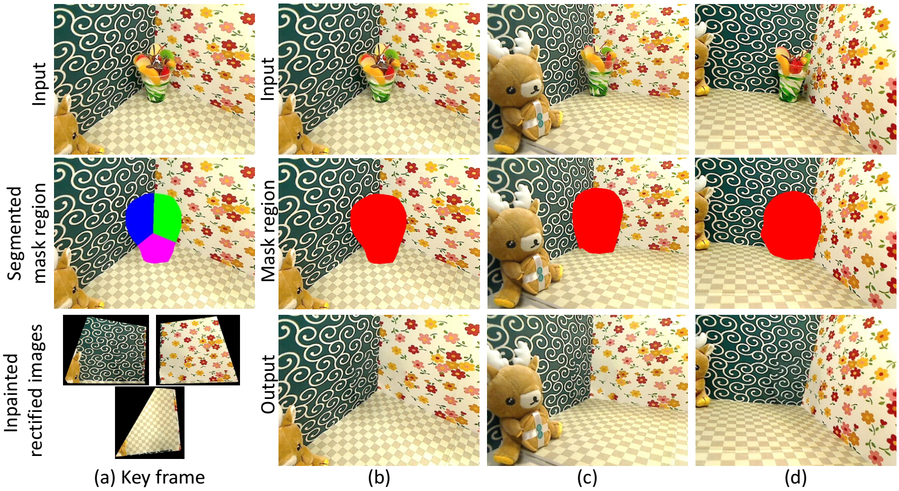


Fig. 6. Experiment for the scene with three textured planes: (a) key frame, (b)–(e) subsequent frames

appearance of the current frame using homographies (c-6), and these are then combined to produce the final output (c-7).

4.5 Experimental Results

We show experimental results in three environments. In the experiment, we used a PC with Windows 7, Core i7-3820QM 2.7 GHz CPU, 8 GB of memory, and a GeForce GT 650M GPU for input images of resolution 640 × 480 captured by a USB camera (Logicool Qcam Pro 9000). The GPU was used for image rectification in process (C). We used PTAM [20] for the visual-SLAM and inpainting methods [10]. In Figs. 6 to 8, Fig. (a) shows the key frame, with the top row showing the input image, the middle row showing the segmented mask region, and the bottom row showing the inpainted results of rectified images. Figs. (b) to (d) show subsequent frames captured from various viewpoints; the top row shows input images, the middle row shows the mask regions, and the bottom row shows output images.

First, we show the results of the indoor scene in Figs. 6. In this scene, textures are successfully generated in the target region, and the temporal coherence is preserved. Second, we show the results for the outdoor scene in Fig. 7, in which the optical parameter of the camera automatically changes with camera motion because of the large difference in luminance between sunny and shady areas and the low dynamic range of the camera. In this scene, the mask region of the key frame is inpainted when the camera's optical parameter adjusts to a shady area, as shown in Fig. (a). The optical parameter is adjusted according to this shady area in Figs. (b) and (d), and to sunny areas in Fig. (c).

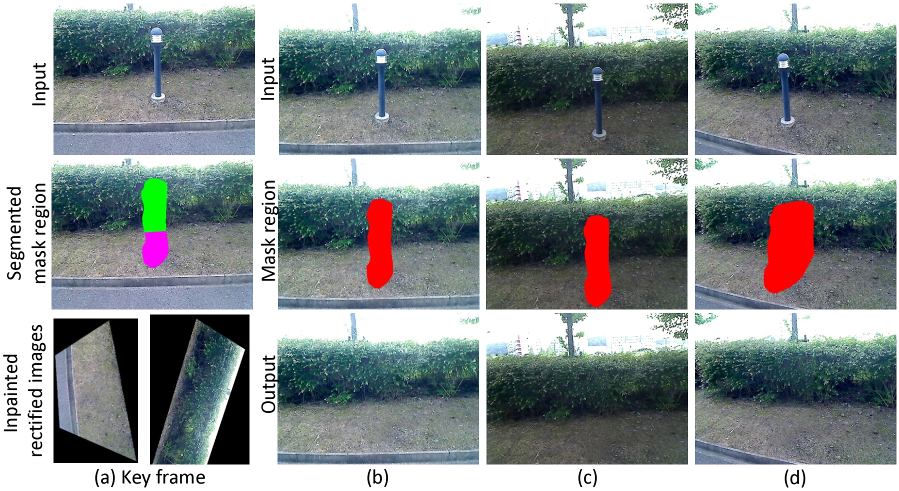


Fig. 7. Experiment for a scene with the camera's optical parameter changed: (a) key frame, (b) to (e) subsequent frames

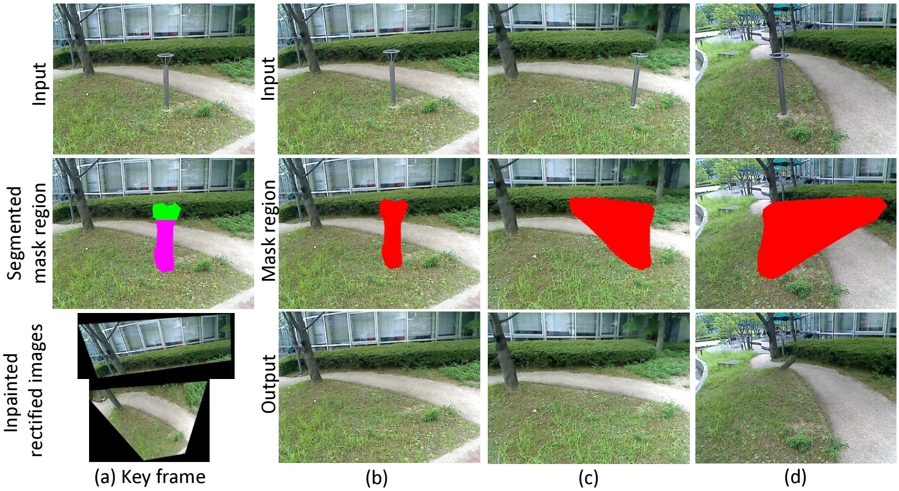


Fig. 8. Experiment for a scene in which the target object is distant from the background objects: (a) key frame, (b) to (e) subsequent frames

Finally, we show the results for the outdoor scene in Fig. 8, in which the target object is distant from the background objects. In this scene, the mask region is larger, as the camera position is farther from the key frame position, as shown in the middle images of Fig. (c). Nevertheless, plausible textures are overlaid on the mask region, as shown in the bottom images of Fig. (c). However, when the target object overlaps the background tree, as in Fig. (d), the tree texture is distorted in the mask region. This is because our method assumes that objects in each rectified image, such as the bottom image of Fig. (a), exist on each plane, and the textures of the current frame are generated by transforming the texture in each rectified image of the key frame using the relevant homography.

The computational time for scene analysis is less than 100 milliseconds, and the frame rate of diminished reality is about 20 to 30 fps. The frame rate decreases as more planes are estimated.

5 Conclusion

This paper introduced image inpainting methods and their application to diminished reality. In addition, we introduced our image inpainting-based diminished reality method, which conducts image inpainting and overlay processes concurrently, and showed experimental results for various environments.

Currently, in the field of image inpainting-based diminished reality, there are still only a few methods, and the applicable situation is limited to some extent. In future, we should deal with various situations. For example, target objects are moving, the structure and texture of background are complex, illumination variously changes. To achieve this, diminished reality techniques will be developed with techniques of Computer Vision, Augmented Reality, and Computer Graphics.

Acknowledgements. This work was partially supported by Grants-in-Aid for Scientific Research Nos. 23240024 and 24700118 from the Ministry of Education, Culture, Sports, Science and Technology.

References

1. Siltanen, S.: Texture generation over the marker area. In: Proc. Int. Symp. on Mixed and Augmented Reality, pp. 253–254 (2006)
2. Korkalo, O., Aittala, M., Siltanen, S.: Light-weight marker hiding for augmented reality. In: Proc. Int. Symp. on Mixed and Augmented Reality, pp. 247–248 (2010)
3. Kawai, N., Yamasaki, M., Sato, T., Yokoya, N.: Diminished reality for AR marker hiding based on image inpainting with reflection of luminance changes. ITE Trans. on Media Technology and Applications 1(4), 343–353 (2013)
4. Kawai, N., Sato, T., Yokoya, N.: Diminished reality considering background structures. In: Proc. Int. Symp. on Mixed and Augmented Reality, pp. 259–260 (2013)
5. Bertalmio, M., Sapiro, G., Caselles, V., Ballester, C.: Image inpainting. In: Proc. SIGGRAPH, pp. 417–424 (2000)

6. Efros, A.A., Leung, T.K.: Texture synthesis by non-parametric sampling. In: Proc. Int. Conf. on Computer Vision, pp. 1033–1038 (1999)
7. Criminisi, A., Pérez, P., Toyama, K.: Region filling and object removal by exemplar-based image inpainting. IEEE Trans. on Image Processing 13(9), 1200–1212 (2004)
8. Wexler, Y., Shechtman, E., Irani, M.: Space-time completion of video. IEEE Trans. on Pattern Analysis and Machine Intelligence 29(3), 463–476 (2007)
9. Kawai, N., Sato, T., Yokoya, N.: Image inpainting considering brightness change and spatial locality of textures and its evaluation. In: Proc. Pacific-Rim Symposium on Image and Video Technology, pp. 271–282 (2009)
10. Kawai, N., Yokoya, N.: Image inpainting considering symmetric patterns. In: Proc. Int. Conf. on Pattern Recognition, pp. 2744–2747 (2012)
11. Darabi, S., Shechtman, E., Barnes, C., Goldman, D.B., Sen, P.: Image melding: Combining inconsistent images using patch-based synthesis. ACM Trans. on Graphics 31(4), 82:1–82:10 (2012)
12. Pavić, D., Schönefeld, V., Kobbelt, L.: Interactive image completion with perspective correction. The Visual Computer 22(9), 671–681 (2006)
13. Huang, J.B., Kopf, J., Ahuja, N., Kang, S.B.: Transformation guided image completion. In: Proc. Int. Conf. on Computational Photography, pp. 1–9 (2013)
14. Barnes, C., Shechtman, E., Finkelstein, A., Goldman, D.B.: PatchMatch: A randomized correspondence algorithm for structural image editing. ACM Trans. on Graphics 28(3), 1–11 (2009)
15. Barnes, C., Shechtman, E., Goldman, D.B., Finkelstein, A.: The Generalized PatchMatch Correspondence Algorithm. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part III. LNCS, vol. 6313, pp. 29–43. Springer, Heidelberg (2010)
16. Herling, J., Broll, W.: Advanced self-contained object removal for realizing real-time diminished reality in unconstrained environments. In: Proc. Int. Symp. on Mixed and Augmented Reality, pp. 207–212 (2010)
17. Herling, J., Broll, W.: Pixmix: A real-time approach to high-quality diminished reality. In: Proc. Int. Symp. on Mixed and Augmented Reality, pp. 141–150 (2012)
18. Simakov, D., Caspi, Y., Shechtman, E., Irani, M.: Summarizing visual data using bidirectional similarity. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, pp. 1–8 (2008)
19. Kato, H., Billinghurst, M.: Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In: Proc. Int. Workshop on Augmented Reality, pp. 85–94 (1999)
20. Klein, G., Murray, D.: Parallel tracking and mapping for small AR workspaces. In: Proc. Int. Symp. on Mixed and Augmented Reality, pp. 225–234 (2007)