Diminished Reality Based on Image Inpainting Considering Background Geometry

Norihiko Kawai, *Member, IEEE,* Tomokazu Sato, *Member, IEEE,* and Naokazu Yokoya, *Member, IEEE*

Abstract—Diminished reality aims to remove real objects from video images and fill in the missing regions with plausible background textures in real time. Most conventional methods based on image inpainting achieve diminished reality by assuming that the background around a target object is almost planar. This paper proposes a new diminished reality method that considers background geometries with less constraints than the conventional ones. In this study, we approximate the background geometry by combining local planes, and improve the quality of image inpainting by correcting the perspective distortion of texture and limiting the search area for finding similar textures as exemplars. The temporal coherence of texture is preserved using the geometries and camera pose estimated by visual-SLAM (Simultaneous Localization and Mapping). The mask region that includes a target object is robustly set in each frame by projecting a 3D region, rather than tracking the object in 2D image space. The effectiveness of the proposed method is successfully demonstrated using several experimental environments.

Index Terms—Diminished reality, Object removal, Image inpainting, Video inpainting, Real-time

1 Introduction

Diminished reality, which visually removes real objects from video images by filling in the missing regions with background textures in real time, 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], [2], [3]. Diminished reality methods can be classified into two categories: One uses actual background images, and the other generates a plausible background image from information around the target object.

The former methods [4], [5], [6], [7], [8], [9], [10] remove real objects by using actual background scenes. In these methods, there are several ways to capture the actual background. Cosco et al. [4] captured the background scene beforehand, Li et al. [8] collected photos from the internet, and Lepetit et al. [7] used past frames in which the background was captured. Other methods [5], [6], [9], [10] have employed multiple cameras. These approaches can be used for applications that need to show actual background images.

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, it is necessary to generate plausible background textures.

(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.

For this task, image inpainting techniques, which generate plausible textures using information from around the target objects in a static image, have often been used [1], [2], [3], [11], [12]. In the image

- *N. Kawai is with the Graduate School of Information Science, Nara Institute of Science and Technology, Ikoma, Nara, Japan, 630-0192. E-mail: norihi-k@is.naist.jp*
- *T. Sato and N. Yokoya are with Nara Institute of Science and Technology.*

inpainting-based methods, most use the assumption that the background geometry is planar, and the motion of a camera is constrained so that the whole target object exists in a frame. The inpainted results are not always comparable to those by the-state-of-art image inpainting methods.

For these problems, we propose an image inpainting-based diminished reality method, which is an extension of [13], assuming that target objects are fixed in the 3D environment and no dynamic objects exist in front of or behind those. The proposed method uses camera poses and geometry in the scene estimated by Visual-SLAM (Simultaneous Localization and Mapping), which enables to move a camera with comparatively free motion and generate high-quality inpainted results. Especially, we pay attention to three issues to achieve high-quality diminished reality: (1) the temporal coherence of textures, (2) the quality of image inpainting, and (3) the determination of mask regions in which foreground textures are to be replaced with background ones. For these issues, we employ the following approaches: (1) The temporal coherence for inpainted textures is preserved using visual-SLAM in conjunction with a background geometry that is approximated from multiple local planes. (2) We employ a semi-dynamic approach which can adopt a high-quality image inpainting method such as one that considers geometric and photometric extensions of texture patterns but usually takes several seconds. The quality of image inpainting is also improved based on the generation of multiple rectified images and a truncated search region. (3) Mask regions are robustly determined in each frame based on 3D tracking of the region of interest.

The rest of this paper is organized as follows: The next section introduces related work and our contribution. Sections 3 and 4 describe the proposed method. In Section 5, we show and discuss experimental results. We conclude this paper in Section 6.

2 Related Work

In this section, we review some related work in consideration of the three issues described above and describe our contribution.

2.1 Temporal coherence of textures

The method in [11] attempts to reduce texture flickering between frames by propagating patch correspondences in image inpainting from frame to frame. However, it is insufficient to achieve geometric consistency between frames taken with large camera motion. To overcome this problem, Herling et al. [12] improved their original method [11] 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. Since the methods in [1], [3] assume that a target object is an AR marker and placed on a plane, they also used a homography calculated from the marker to generate a background image or synthesize an inpainted result on an AR marker. These methods successfully preserve the temporal coherence of planar scenes.

However, if the background of the target scene is not planar, and the scene is observed from quite different viewpoints, as shown in Fig. 1(a), changes in the appearance of textures on different geometries cannot be compensated using a single homography. This results in the unnatural motion of inpainted textures. In the proposed method, we approximate the background geometry 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 by considering the estimated planes and camera-pose given by visual-SLAM.

2.2 Quality of image inpainting

Siltanen [1] mixed several specific pixel values in a marker-hiding application. Although this method can rapidly generate textures, it is difficult to generate natural and complex textures using such a simple approach. To synthesize more natural textures for diminished reality, Herling et al. [11] applied an example-based image inpainting method [14] and a high-speed approximate nearest neighbor search [15] with use of grayscale and reduction of resolution. Moreover, Herling et al. [12] have improved their energy function by considering spatial costs, enhancing the quality of inpainting while preserving real-time processing. However, geometric and photometric expansions of exemplar texture patterns, which are considered in the recent image inpainting methods [16], [17] to deal with flipping and rotating patterns with different brightness, are difficult to be employed due to the computational cost. Therefore, results inpainted by the methods in [11], [12] do not always comparable to those by such recent image inpainting methods. In addition, although the methods in [11], [12] apply image inpainting to the original appearance of the input image, they often produce unnatural results especially when an image's regular patterns have a distorted perspective and the amount of perspective distortion is large.

To overcome the problem of computational cost when using recent image inpainting methods, we employ a semi-dynamic approach, which conducts two processes concurrently, as in methods [2], [3]: image inpainting for a key frame, and the overlay of the inpainted texture for each frame. In this approach, though target objects are hidden with incomplete textures until the image inpainting finishes, advanced image inpainting methods can be applied.

To solve the problem of the image quality influenced by perspective distortion, some researchers including us have tried to remove the influence. Our previous marker-hiding method [3] corrected the perspective distortion using an AR marker, meaning that the size of regular texture patterns could be unified based on the assumption that an AR marker exists on a plane. In the proposed method, we extend this idea for more general scenes. In the field of image inpainting for a static image, some methods are proposed for perspective correction [18], [19]. These methods successfully correct perspective distortions by manually giving some interactions. However, such interactions give the user burdens and take some time before starting diminished reality in our situation. Therefore, we use a 3D geometry to deal with perspective correction. Specifically, we generate multiple rectified images, one for each of the estimated planes. In addition to this, we add a constraint to automatically limit the search region using geometries around a target object, thus increasing the quality of inpainted textures, while the whole input image is searched for similar texture patterns in the conventional methods.

2.3 Determination of mask regions

Mask regions (those that include target objects) have to be found in each frame to ensure that the objects are removed from the image. Objects such as AR markers [1], [2], [3] can easily be tracked using software libraries (e.g., ARToolkit [20]), 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 [11], but this method is not robust for textured backgrounds. For this problem, Herling et al. [12] use their original feature points that store the appearance of the image, and distribute them around the target objects. Tracking the feature points segments the image into the mask and other regions in each frame. Although this method works well for scenes with textured backgrounds, it has the limitation that the entire object must always be in the video frame. In the previous version of our research [13], we did not update mask regions through all frames. Therefore, target objects have to exist almost on the background object so that the target objects can be in the mask regions.

In the proposed method, we robustly determine the mask regions in all frames by tracking the 3D region including target objects in 3D space that are generated in a key frame. Projecting the 3D region from the camera position onto the estimated planes sets mask regions in each frame of the rectified images, and ensures that the target objects do not exit the mask regions. In addition, the target objects do not always have to be in the video frame, unlike in conventional methods [11], [12].

However, since the initial 3D region is generated only from the key frame, the projected mask regions often become large when the target objects are distant from the background object and the camera largely moves from the key frame position, which reduces the frame rate. For such a case, our method also provides a user an option to crop the 3D region, which works with re-inpainting the image with a smaller missing region.

2.4 Contribution

In summary, our primary contributions to the field of diminished reality are: (1) the preservation of temporal coherence for inpainted textures in more general scenes than conventional methods can handle, (2) improved quality of image inpainting, and (3) robust determination of mask regions in each frame. The first point uses visual-SLAM in conjunction with a background geometry that is approximated from multiple local planes. The second relies on the generation of multiple rectified images and a truncated search region, the employment of a semi-dynamic approach with a high-quality image inpainting method, and the update of inpainted results when the actual background can be observed. The third contribution is based on 3D tracking of the region of interest.

3 Diminished Reality Considering Background Geometry

Fig. 2 shows the pipeline of the proposed diminished reality technique. Our method first analyzes the target scene and carries out the pre-processing of diminished reality (I). Diminished reality is then achieved by a semi-dynamic approach that conducts two processes concurrently, as in our previous marker-hiding method [3]: example-based image inpainting for a key frame (II), and the overlay of the inpainted texture for each frame (III). Although process (II) is not performed in real-time, users can start applications immediately by performing processes (II) and (III) concurrently. Within several seconds of starting process (II), users can experience diminished reality with a completely inpainted result.

In the proposed framework, the background geometry is required to have a multiple-plane representation, and the background must have some textures to enable the extraction of planes and estimation of camera poses by visual-SLAM. Therefore, the target scene is a little restricted. However, the proposed method is applicable in many real environments, because artificial and natural geometries, such as artificial tiles, garden trees, and grass ground, often consist of multiple local planes and have some texture. In the following, we describe processes (I), (II), and (III) in detail.

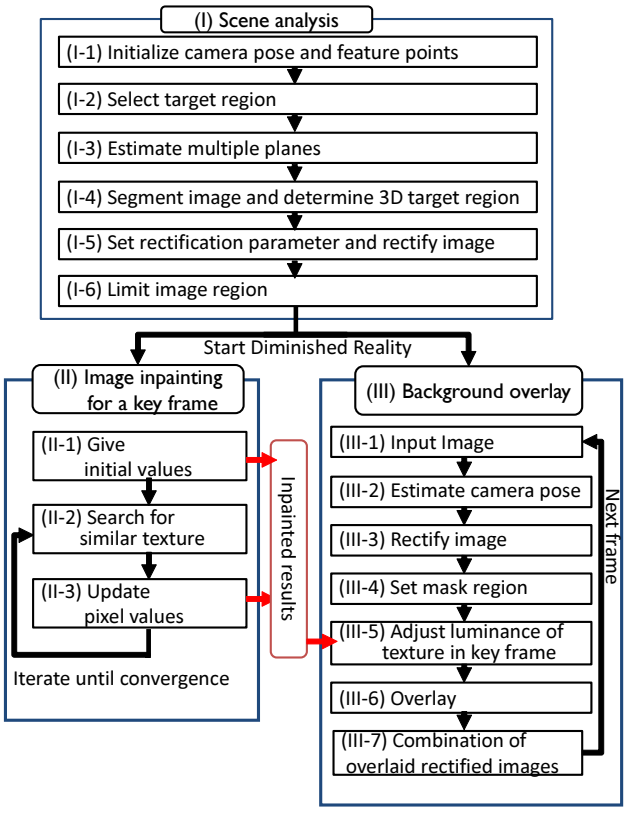


Fig. 2. Pipeline of the proposed diminished reality technique.

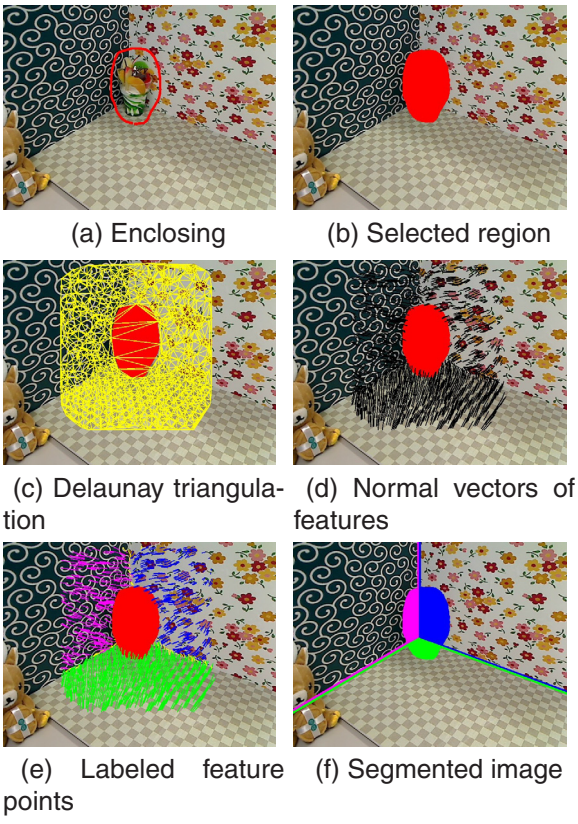


Fig. 3. Scene analysis.

3.1 Scene Analysis

During pre-processing for the proposed diminished reality method, 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 (I-1). A user then manually selects a region that includes target objects by enclosing the region, as shown in Figs. 3(a) and (b) (I-2). In this step, pixels on the line are periodically and sequentially stored while a user drags the line, and the pixels and lines connecting them are stored as the boundary of the target region. The frame when the user finishes enclosing the region is set as a key frame and is used for image inpainting in process (II). The selected 2D region is converted to a 3D target region, and this is used to set the mask regions in each frame in process (III). The conversion is performed in the later step of process (I), described later in this section.

Next, feature points that are outside the target region and whose chessboard distance from the boundary of the target region is less than a threshold, which is empirically determined, are picked up, and neighboring feature points are connected by Delaunay triangulation [21] from their 2D image coordinates, as shown in Fig. 3(c). Normal vectors of the generated triangles are calculated using the 3D coordinates of feature points, and normal vectors of feature points are determined by averaging the normal vectors of all triangles including the feature points, as shown in Fig. 3(d). Next, each feature point is classified into multiple groups based on the difference between its normal vector and the mean normal vector of all feature points in a group, which is iteratively updated using mean-shift [22]. It should be noted that the number of groups is automatically determined. Specifically, a feature point is first selected at random as $\mathbf{x}_i$ (i is the index of feature points), and the normal vector $\mathbf{m}$ is calculated in an iterative manner as:

$$\mathbf{m}_t(\mathbf{n}_i) = \frac{\Sigma_{j=1}^{N} w(\mathbf{n}_j)\mathbf{n}_j}{M}, \tag{1}$$

$$w(\mathbf{n}_j) = \begin{cases} 1 & (\mathbf{n}_j \cdot \mathbf{m}_{t-1}(\mathbf{n}_i) > C) \\ 0 & (otherwise)\,, \end{cases} \tag{2}$$

where $\mathbf{n}_i$ and $\mathbf{n}_j$ indicate the unit normal vectors of feature points $\mathbf{x}_i$ and $\mathbf{x}_j$. $\mathbf{m}_t(\mathbf{n}_i)$ denotes the unit mean vector in the t-th iteration, and the process starts with $\mathbf{m}_0(\mathbf{n}_i) = \mathbf{n}_i$. M is some normalization factor such that $\mathbf{m}_t(\mathbf{n}_i)$ becomes a unit vector. N is the number of feature points picked up in the above process, and C is a constant threshold. After several iterations, feature points $\mathbf{n}_j$ satisfying $\mathbf{n}_j \cdot \mathbf{m}_t(\mathbf{n}_i) > C$ are labeled as belonging to the same group, and these points are removed. Among those that remain, a new feature point is randomly selected, and the above process is

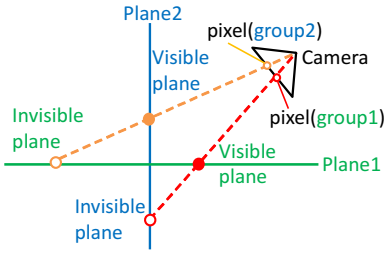


Fig. 4. Pixel grouping.

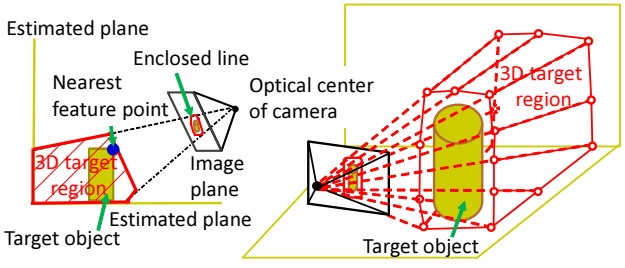


Fig. 5. 3D target region.

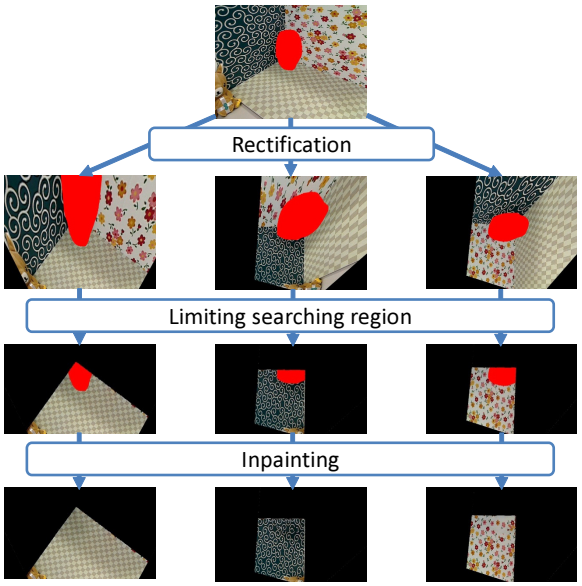


Fig. 6. Rectification and region limitation for inpainting.

repeated until all feature points are labeled, as shown in Fig. 3(e). Next, a plane is fitted to the feature points of each group using LMeds (Least Median of Squares) [23]. The number of planes K is basically the same as that of labeled groups, but if the number of feature points in a particular group is much smaller than in others, the group is removed.

All the fitted planes are projected onto the image plane, and each pixel is grouped according to which plane is visible at the pixel of the camera image as shown in Fig. 4. According to the pixel grouping, the whole image, including the missing region, is segmented as shown in Fig. 3(f). In addition, the 3D target region is generated from the 2D selected region, so the 3D region must include the target object. Specifically, as shown in Fig. 5, the feature point on the target object that is nearest the camera position is first found. The 3D mesh model generated by the following planes is then set as the 3D target region.

- Estimated plane for background region.
- Plane that includes the nearest feature point and is parallel to the image plane of the camera.
- Planes between any pair of two rays that pass through the optical center of the camera and adjacent two points on the line drawn by the user around the object to remove (Fig. 3(a)).

Next, the rectified images are generated from the key frame, and the information for rectifying the subsequent frames are stored at the same time (I-5). As shown in Fig. 6, the perspective distortion of input images is corrected by calculating homography matrices for K planes as if each plane was captured by a camera in front of it. The number of rectified images is the same as the number of planes. We now describe how to calculate a homography matrix $\mathbf{H}_k$ for plane k.

Here, we associate a 3D length, two directions, and a position in the scene with a pixel length, horizontal and vertical axes, and a center of the rectified image. Specifically, we calculate the 3D length l_k corresponding to a single pixel at the center of gravity $\mathbf{g}_k$ of the feature points on plane k by projecting a pixel corresponding to the gravity center and one of its adjacent pixels onto plane k. We then use two arbitrary perpendicular vectors of length λl_k on plane k as basis vectors $(\mathbf{u}_k, \mathbf{v}_k)$ for the x and y axes of the rectified image, where λ is a constant that influences the resolution of the rectified image. The larger the value of λ, the lower the resolution of the rectified image. The center of gravity is set as the center of the rectified image.

Finally, $\mathbf{H}_k$ is calculated from the four pairs of pixels determined by projecting the four 3D points $(\mathbf{g}_k, \mathbf{g}_k + \mathbf{u}_k, \mathbf{g}_k + \mathbf{v}_k, \mathbf{g}_k + \mathbf{u}_k + \mathbf{v}_k)$ on plane k to the rectified image and the key frame's input image plane. Homography matrices are calculated for all the planes in the same way. The 3D coordinates of the four 3D points on each plane k and the 2D coordinates of the corresponding four pixels in each rectified image are stored and later used for rectifying the subsequent frames in process (III).

In the final step (I-6), we limit the search region in which textures can be used as exemplars for inpainting in process (II) based on the segmented image. Thus, textures from other planes cannot be used as exemplars.

3.2 Image Inpainting for Multiple Rectified Images

We apply an example-based image inpainting method to each rectified image in which a region is limited.

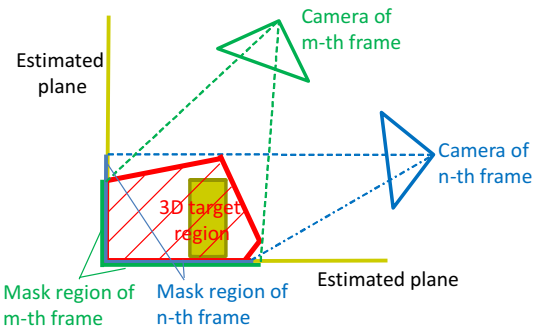


Fig. 7. Determination of mask region on each rectified image by projecting the 3D target region to each plane.

The rectified images of the key frame in which missing regions are inpainted are retained for the background overlay process (III). For image inpainting, the proposed framework can adopt arbitrary example-based methods that use global optimization, such as those in [16], [17], [24] and the inpainting part in [12]. In Figs. 1 and 6, we employ an existing image inpainting method [17], which allows for brightness compensation and the symmetric transformation of texture patterns, and runs at relatively high speed.

Specifically in process (II), 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 (II-2), and the second updates pixel values in the missing regions (II-3) in a similar way to [17]. In each iteration, process (II) stores the tentative inpainted result in the memory shared with process (III). After the energy converges, the completely inpainted result is stored and used in process (III).

3.3 Real-time Overlay of Inpainted Textures

In process (III), after capturing an image (III-1) and calculating a camera pose using visual-SLAM (III-2), a rectified image is generated for each plane using the current camera pose and information for rectification, which is stored in step (I-5), (III-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, as shown in Fig. 7 (III-4). The right-hand images of Fig. 8 show an example of the 3D target region and the projected mask regions. It should be noted that the target objects always exist within the projected mask regions, regardless of the camera pose and projected mask shapes, because the 3D target region includes the target objects.

Next, the mask regions of rectified images are filled in using the texture in the rectified images of the

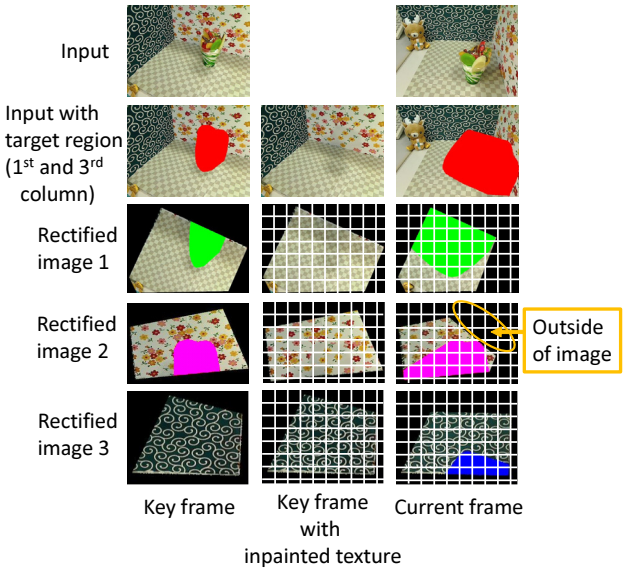


Fig. 8. Divided grids for calculating luminance changes.

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 (III-5) before overlaying this texture on the mask regions (III-6). In this study, we use a similar approach as in our previous marker-hiding method [3], whereby 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. We do not directly use the amplitude of luminance changes in the target object, but instead estimate this from the surrounding region, because the photometric properties of the object and the surrounding materials are different.

The difference from our previous marker-hiding method [3] is that the luminance change of each rectified image is calculated separately, because the degree of luminance change on each plane is thought to differ according to the plane's normal direction. In addition, the number of pixels in the mask regions varies in each frame because of the projection of the 3D target region. In the following, we describe the method for calculating the luminance change coefficients by which pixel values in the rectified images of the key frame are multiplied when they are overlaid on the mask regions in the rectified images of the current frame.

As shown in Fig. 8, we first divide the rectified images of the key frame with inpainted texture and a current frame into grids of $s \times s$ pixels. For efficient processing, the grid-wise luminance change coefficients are calculated before the pixel-wise coefficients are determined. We then calculate luminance change coefficients of the grid according to the following two

conditions.

- The grid does not include the mask region or areas outside the image in the current frame.
- The adjacent grid includes the mask region in the current frame.

The luminance change coefficient is calculated by dividing the sum of pixel values in the grid for the current frame by that for the key frame. Next, the coefficients of grids that include the mask region in the current frame are estimated using the coefficients calculated for surrounding grids. In this study, we assume that adjacent grids have small luminance changes because of ambient illumination changes and soft shadows caused by users, and therefore the luminance change coefficient of each grid is calculated so as to minimize the following cost function E:

$$E = \sum_{(p,q)\in R} (\hat{\alpha_p} - \hat{\alpha_q})^2, \tag{3}$$

where $\hat{\alpha}_p$ and $\hat{\alpha}_q$ are luminance change coefficients of grids p and q, and R is the set of pairs of grid indexes from the eight neighboring grids. It should be noted that some surrounding grids extend outside of the image because of camera motion, as shown in rectified image 2 of the current frame in Fig. 8. In such cases, the coefficients of the grids including this outside area are not included in Eq. (3). The luminance change coefficients that minimize E are analytically calculated by differentiating E with respect to each luminance ratio and solving the system of equations. After calculating the luminance change coefficient for each grid, the luminance change coefficient $\hat{\alpha}_p$ of grid p is copied to pixel-wise luminance change coefficient α_j of the central pixel j in grid p, and the pixel-wise luminance change coefficients for the remaining pixels, excluding the central pixels in the grids, in the mask region are calculated by bilinear interpolation. In this process, the coefficients for RGB are independently calculated and multiplied in order to deal with chromatic changes.

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

4 Optional Update of 3D Target Region and Inpainted Images

When a target object is distant from the background objects, the projected mask region is larger, as the camera position is farther from the key frame position as shown in Fig. 9. Such a situation reduces the frame rate because the number of parameters in Eq. (3) increases.

For such a case, our method provides users an option to crop a part of the 3D target region with

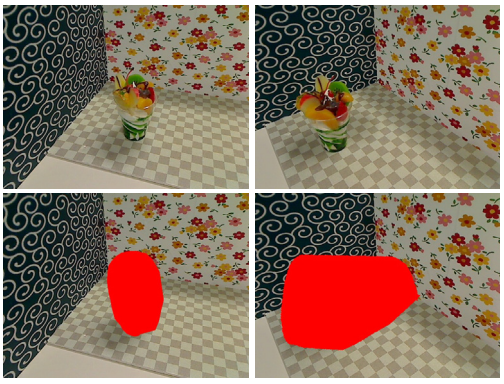


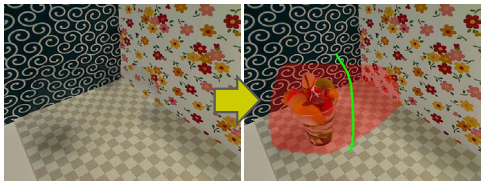
Fig. 9. Inputs and mask regions varying according to camera poses.

a user interaction based on the idea of Shape-from-Silhouette to reduce the projected mask regions. This process is also accompanied by re-inpainting the key frame with the additional exemplar textures, which are newly visible background, and a smaller missing region, which may yield better results than the first inpainting.

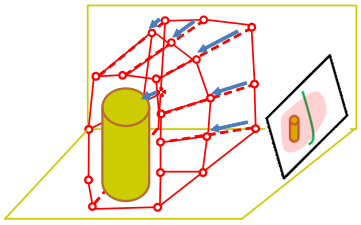
Specifically, a user first switches the display mode from showing the target object-less image to showing a translucent mask, and crop the 3D target region by drawing a curve as shown in Fig. 10(a). For cropping the 3D region, our method calculates the intersection positions of the drawn curve with lines on the image plane projected from 3D ones connecting the front polygon and the background plane, and moves the points on the background plane to 3D positions corresponding to the intersections as shown in Fig. 10(b). Next, the image regions corresponding to the removed 3D region are added to the missing regions in the key frame images with Poisson blending [25] as shown in Figs. 10(c) and (d), because the intensities of the current and key frames may differ. After that, image inpainting is again applied to the key frame images that still have missing regions.

5 Experiments

We carried out experiments in several environments using 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 $\times$ 480 captured by a USB camera (Logicool Qcam Pro 9000). We used the GPU-based function provided by OpenCV 2.4.5 with CUDA support for image rectification in process (III). We used PTAM [26] for the visual-SLAM and inpainting methods [17]. We set $\lambda = 1.5$ for the rectified image resolution, and set $s = 20$ for luminance adjustment by considering the trade-off of quality and processing time. We set the segmentation threshold as $C = cos(\pi/6)$.



(a) Switching display mode and drawing a curve



(b) Updating 3D target region

(c) Rectified images with missing regions for first inpainting

(d) Rectified images with missing regions for second inpainting after adding visible background

Fig. 10. Cropping of 3D target region and updating of rectified key frame images.

5.1 Comparison of Proposed and Conventional Methods

We compare the results given by the proposed method with those from a conventional approach using a single homography with respect to the quality of image inpainting and temporal coherence of textures for two scenes.

First, we demonstrate the effectiveness of image rectification and search region limitation in terms of the quality of image inpainting. Fig. 11 compares the inpainted results. Figs. 11(f) show the result obtained by applying the inpainting method [17] to the original input image in Figs. 11(b). For scene 1, as we can see, unnatural textures were generated in the target region due to the perspective distortion and the complexity of textures. For scene 2, plausible textures were generated for the garden plants, but slightly blurry texture were generated on the ground due to the perspective distortion. On the other hand, as shown in Figs. 11(c) and (d), the proposed method successfully segmented the images by finding three and two planes for scenes 1 and 2, respectively, and gave natural textures to the target regions in the rectified images, which have various regular and random textures. By combining the inpainted rectified images, natural textures could be generated in the target region, as shown in Figs. 11(e).

Next, we consider the temporal coherence of textures by comparing results using the proposed method with those from a conventional approach using a single homography [12], [3] using the same sequences as Fig. 11. In the conventional approach, we fitted one plane to the feature points near a mask region using the least-squares method, and the inpainted result generated by the proposed method (Fig. 11(e)) was geometrically transformed using the plane for each frame. To focus on the temporal coherence, we used the same inpainted result. Fig. 12 compares the output. As shown in the bottom images of Fig. 12, camera motion causes edges to become disconnected on the boundaries of the target regions because of different geometries between the target and surrounding regions. In contrast, as shown in the middle images of Fig. 12, the temporal coherence of the inpainted textures is preserved under various camera movements. Conventional methods [12], [3] have the limitation that the whole of a target object must exist in an image for each frame to continue visually removing the object. However, the proposed method can successfully overlay natural textures on the mask region using the extracted planes and camera pose estimated by visual-SLAM, even if the camera motion causes the target object to leave the image, as in the third and sixth column images of Fig. 12.

5.2 Results for Various Scenes

In this section, we report the results of experiments for additional two scenes under different conditions. In Figs. 13 and 14, 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 (e) 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 discuss the experiment for the scene in Fig. 13, 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 Figs. (c) and (e). From the results, we can confirm that the inpainted textures in the target region are successfully adjusted by the luminance adjustment of the proposed method.

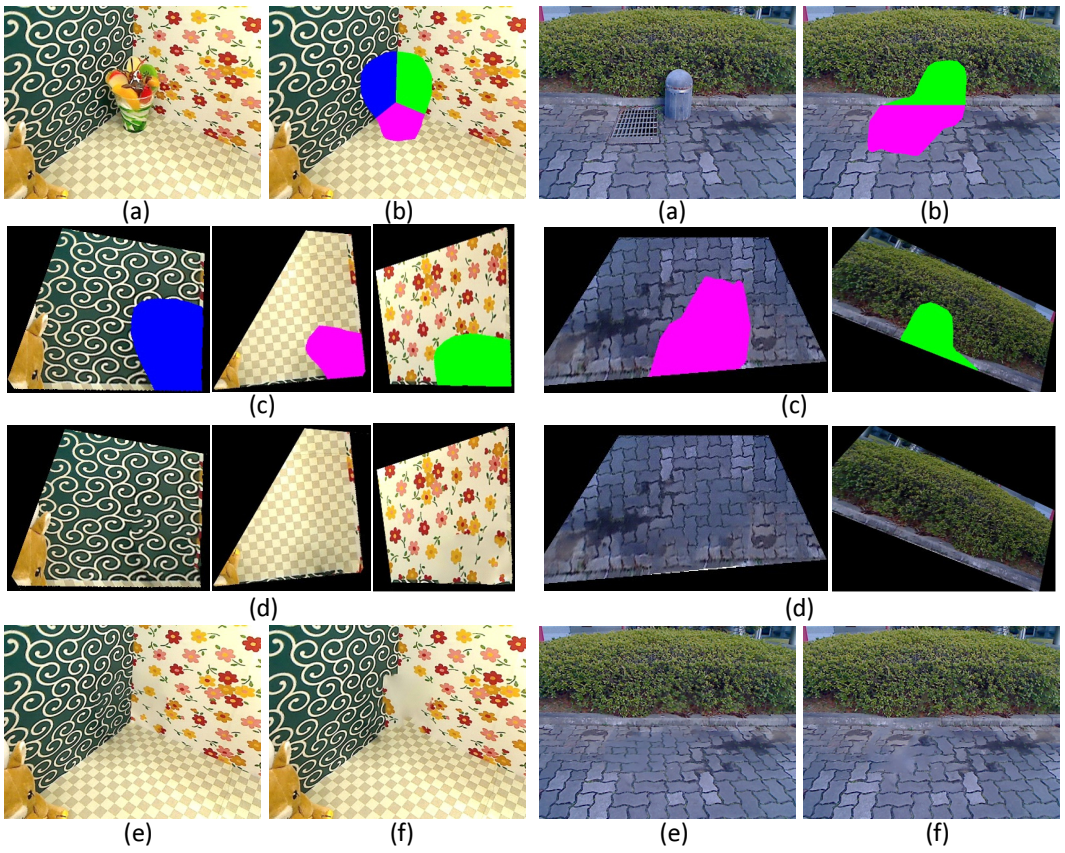


Fig. 11. Comparison of inpainted results for two scenes: (a) input image, (b) segmented mask region, (c) three rectified and region-limited images, (d) inpainted results for rectified images, (e) result from the combination of rectified images, and (f) result of applying the inpainting method to original appearance of input image (b).

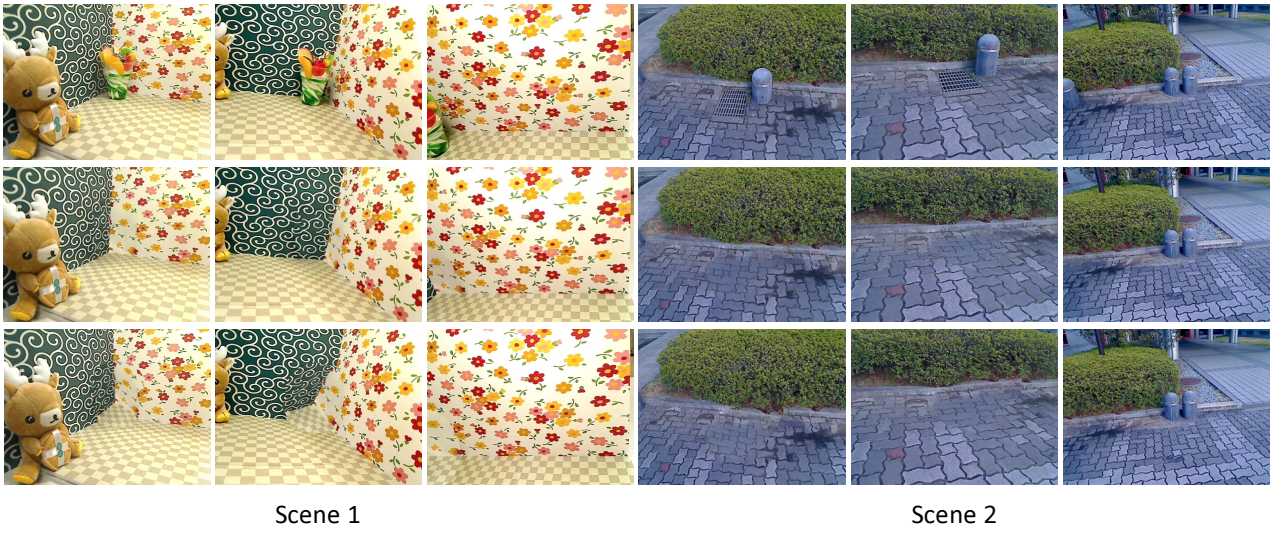


Fig. 12. Comparison of results from different viewpoints by the proposed method and the conventional approach using one homography for two scenes. Top: input images. Middle: results given by the proposed method. Bottom: results given by the conventional approach.

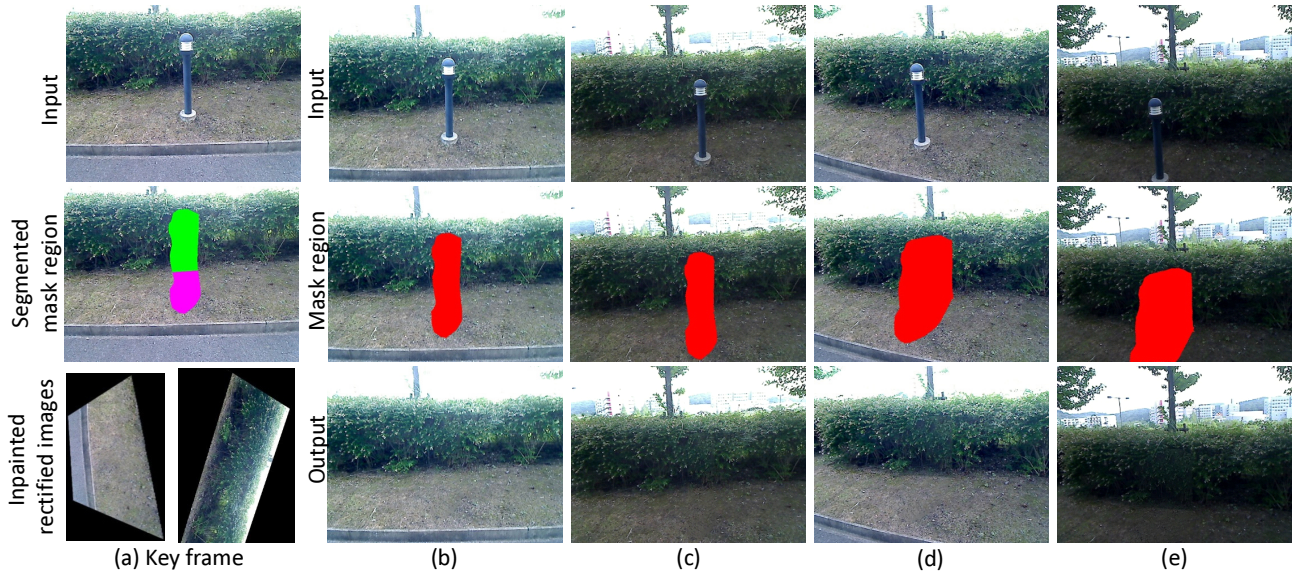


Fig. 13. Experiment for a scene with the camera's optical parameter changed: (a) key frame, (b) to (e) subsequent frames.

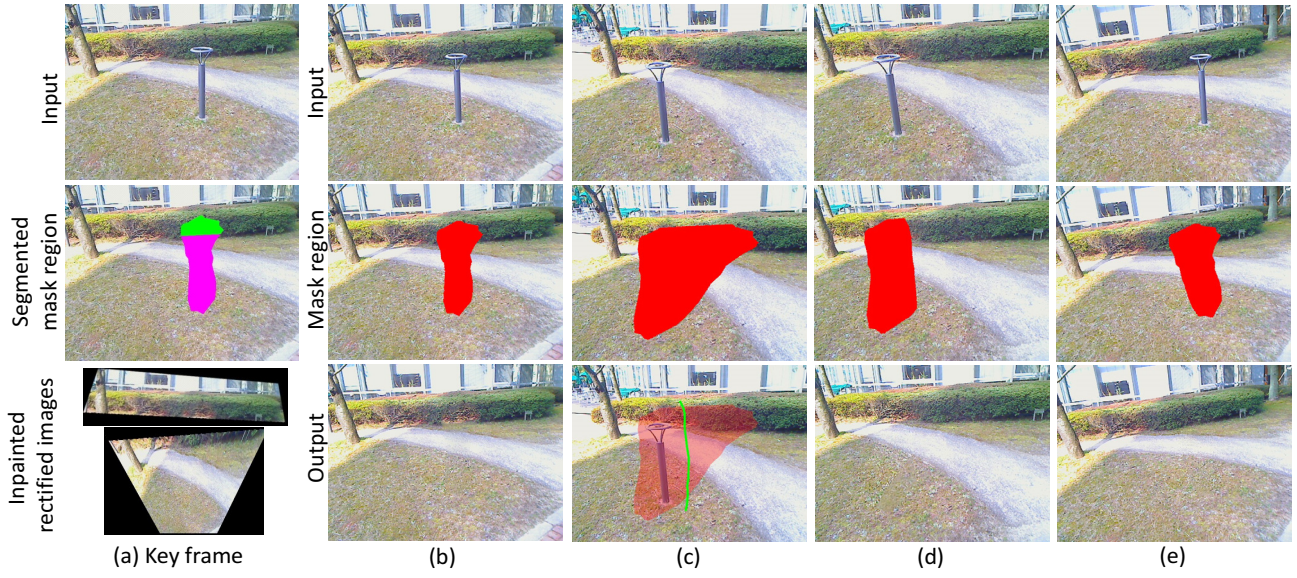


Fig. 14. Experiment for a scene in which the target object is distant from the background objects: (a) key frame, (b) to (e) subsequent frames.

Second, we consider the experiment for the scene in Fig. 14, in which the target object is distant from the background objects. In this scene, we cropped the 3D target region because the mask region became large when the camera position moved from the key frame position, as shown in Fig. (c). As a result, the projected mask region reduced as shown in Fig. (d). Comparing the result before cropping in Fig. (b) with that in Fig. (e) in which the camera moved back near to the key frame position after cropping, we can confirm that the inpainted texture was replaced with the newly visible background in the large area of the mask region.

5.3 Parameters and Computational Time

We now study the influence of the number of planes and parameters on computational time and quality. First, we discuss the computational time with different number of planes using three scenes in which one, two, and three planes are estimated, as shown in Fig. 15. Table 1 shows the computational time of each process, as well as the frame rate for 2,000 frames with and without GPU after commencing the diminished reality method. The proposed method implements

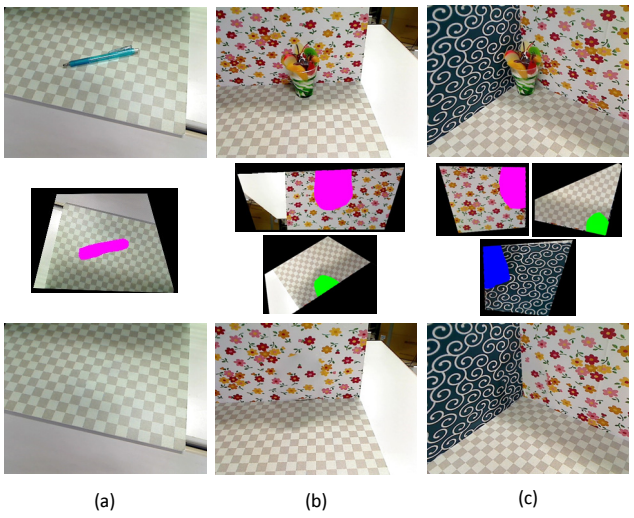


Fig. 15. Scenes for measuring computational time. From top to bottom, inputs, rectified images, results of scenes with (a) one plane, (b) two planes, (c) three planes.

TABLE 1
Computational time.

Scene	Scene Analysis (a-3) to (a-6)	Inpainting (Max. No. of missing pixels in rectified images)
(a) 1 plane	30 msec.	3.30 sec. (8091 px)
(b) 2 planes	78 msec.	4.52 sec. (12108 px)
(c) 3 planes	95 msec.	4.42 sec. (9466 px)
Scene	Frame rate with GPU	Frame rate without GPU
(a) 1 plane	32.39 fps	27.48 fps
(b) 2 planes	24.80 fps	20.48 fps
(c) 3 planes	21.79 fps	15.48 fps

TABLE 2
Computational time with different parameters.

Parameters	Inpainting (No. of missing pixels)	Frame rate with GPU
$\lambda = 1.5$ $s = 5$	2.99 sec (7654 px)	29.94 fps
$\lambda = 3.0$ $s = 5$	1.22 sec. (1905 px)	31.46 fps
$\lambda = 3.0$ $s = 20$	1.87 sec. (1884 px)	32.49 fps

parallel processing to inpaint rectified images and adjust the luminance of inpainted textures in the case of multiple planes. The time taken for image inpainting is recorded as the time when the inpainting of all rectified images is complete. For scene analysis, the computational time for the three cases is small enough for a user to wait for diminished reality to start. For inpainting, the computational time mainly depends on the size of the missing region, rather than the number of planes. In terms of frame rate, the GPU effectively speeds up the processing, but the processing speed decreases as more planes are estimated because of sequential rectification for each plane. Since our implementation employed OpenCV to use the GPU, the frame rate may be improved with more sophisticated management of multi-core CPU in conjunction with GPU.

Next, we discuss the influence of the parameters λ for the resolution of the rectified image and grid size $s \times s$ for luminance adjustment on the quality of overlaid texture and computational time. In this experiment, we use the environment in which one plane is estimated, like in Fig. 15(a). First, we show the computational time with different λ and s in Table 2. Because larger λ lowers the resolution of a rectified image, the inpainting speeds up and the frame rate is increased with larger λ. Because larger s decreases the number of parameters in Eq. (3), the frame rate is increased with larger s. Next, we compare the quality of overlaid textures with different parameters in terms of resolution and luminance. First, we discuss the resolution with different values of λ in Fig. 16. When the camera comes close to the target object, we can realize that the resolution of overlaid texture is insufficient with $\lambda = 3.0$, as shown in Fig. 16(b), meaning the resolution of the rectified image is too low. Second, Fig. 17 compares the quality of luminance adjustment. In indoor environments, there are often local luminance changes caused by object occlusion and near light sources. Especially in such an environment, the adjustment of luminance of inpainted texture is insufficient with the large grid size, as shown in the boundary of the missing region in Fig. 17(b). From these results, we have to carefully set the parameters considering the supposed environment in conjunction with considering the trade-off between the computational time and the quality.

5.4 Discussion and Limitations

As described in sections 5.1 and 5.2, the proposed method works well for the various real environments as shown in Figs. 12, 13 and 14. This means that the various real scenes can be represented by a small number of planes and image inpainting is effective for such scenes. However, the proposed method has theoretical limitations, which were not demonstrated in the experiments. This section summarizes and discusses the limitations of the proposed method.

5.4.1 SLAM

Since we used PTAM [26] as a visual-SLAM method, the target scenes must have some textures to enable the extraction of feature points to calculate camera poses. For plane detection, each plane must have some textures for extracting a sufficient number of feature points to fit an appropriate plane. If, for example, a scene consists of two planes and one of the planes has few textures, the proposed method may detect only one plane and not achieve the temporal coherence. In addition, SLAM methods such as PTAM usually

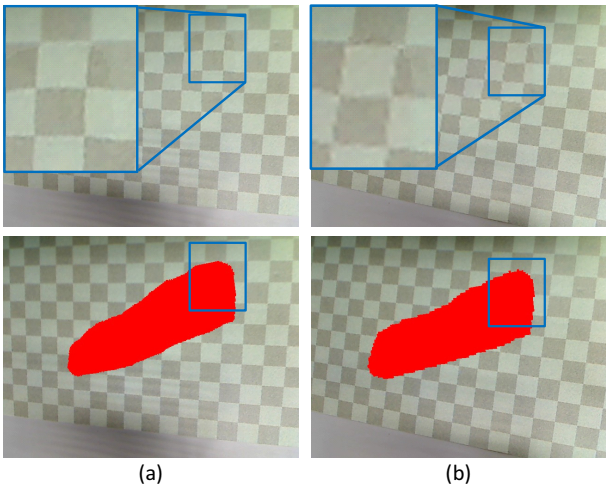


Fig. 16. Resolution of overlaid texture with different values of λ: (a) $\lambda = 1.5$ (b) $\lambda = 3.0$. Top: output. Bottom: input with a mask.

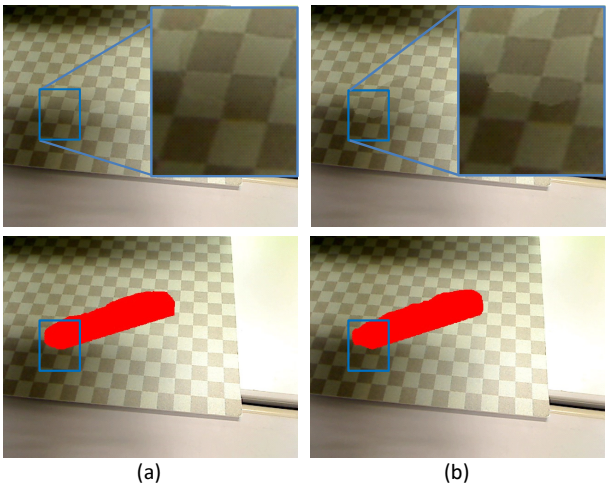


Fig. 17. Adjustment of luminance with different values of s: (a) $s = 5$ (b) $s = 20$. Top: output. Bottom: input with a mask.

require a user a specific start to initialize the camera pose, which may give some burden for the user.

5.4.2 Background geometry

Since the proposed method assumes that the background geometry is approximated by multiple planes, it cannot handle the scenes that do not satisfy the assumption such as a curved background geometry and no background objects with which we can see a horizontal line. For example, for a scene with a curved background geometry as shown in Fig. 18, while the background is divided into multiple planes, despite its curved shape, and the missing region of each rectified image is plausibly inpainted by the proposed method, disconnected texture appeared on the boundary of the mask region when the camera

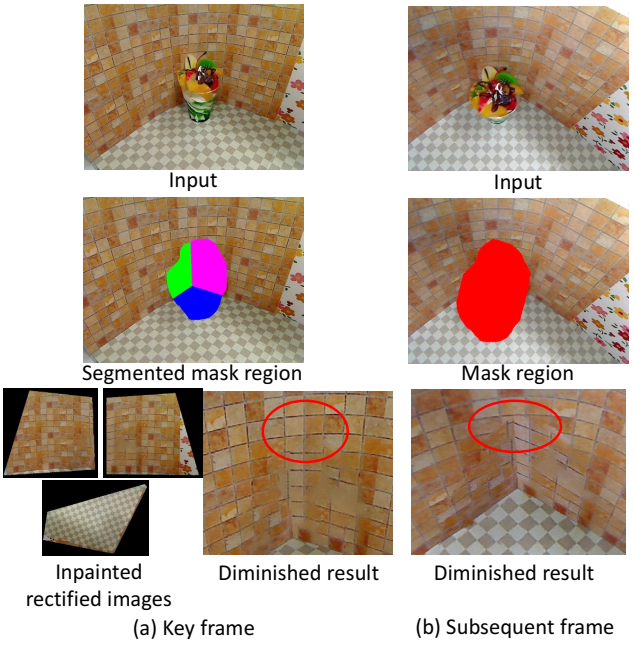


Fig. 18. Example of limitations: for a scene with a curved background: (a) key frame, (b) subsequent frame.

moves in a subsequent frame. In addition, it cannot handle individual but parallel planes because our clustering of feature points uses only normal vectors and feature points on such planes are grouped into the same cluster. In such a case, only a single plane is fitted to the feature points despite the existence of multiple planes, and the inappropriate plane fitting cases texture disconnection according to camera motion.

5.4.3 Quality of overlaid textures

The quality of overlaid texture in each frame largely depends on the inpainted result for a key frame. Therefore, the key frame should include sufficiently large amount of sample textures with high resolution and good focus to obtain good results by inpainting. In addition, the geometry should be correctly estimated to correct perspective distortion and preserve temporal coherence of textures. However, even if the requirement is satisfied, the quality of the output frame is often limited because of the following reasons:

- A part of the mask region is not covered by the background texture because the partial area is corresponding to a region outside the rectified image of the key frame when the camera largely moves.
- The overlaid image blurs when the camera moves close to the background object because the resolution of the inpainted result is fixed.
- The overlaid image cannot represent the strong cast shadows and spotlights when they come on

the mask region because our method forces the luminance change coefficients for adjacent grids to be similar.
- The overlaid textures with high quality may cause visual inconsistency when the quality of the captured frame is deteriorated by some noise such as motion blurs due to camera rapid motion.

5.4.4 Selection of target objects

Drawing a contour around a target object as precisely as possible is desirable because a small target region tends to enable the better inpainted result. Suppose that the proposed method runs on smartphones and tablet PC, it may be difficult for a user to draw it precisely. Therefore, we should consider such an approach as [12] in which the contour around the target object shrinks down to the boundary of it after user's rough selection.

6 Conclusion

We have proposed a new diminished reality method for more general scenes than conventional methods. By approximating the background geometry using a combination of local planes, the perspective distortion of texture is corrected and the search area is limited, thus improving the quality of image inpainting. The temporal coherence of texture is also preserved using the geometries and camera pose estimated by visual-SLAM. Mask regions are robustly set in each frame using a 3D region, instead of tracking target objects in 2D image space, and the 3D region is interactively cropped and simultaneously inpainted results are updated with the additional visible background.

In the experiments, we compared the results given by the proposed method with those from a conventional approach using a single homography. This confirmed that the proposed method gives more natural results with comparatively unrestricted camera motion. In addition, we demonstrated the effectiveness of the proposed method for various scenes. However, the proposed method has the limitations that the camera movement is still restricted by various background geometries and the management of a key frame, and the quality of diminished reality largely depends on the robustness of SLAM.

To address the former problem, we must deal with additional geometries and extend the key frame using subsequent frames. As for the robustness of SLAM, we can employ a different method according to the scene. For example, in indoor scenes, we could apply an RGBD camera-based SLAM method, such as KinectFusion [27], as these robustly estimate background geometries even for non-textured regions.

Acknowledgments

This work was partially supported by Grants-in-Aid for Scientific Research Nos. 23240024, 24700118 and 15K16039 from the Japan Society for the Promotion of Science (JSPS).

References

[1] S. Siltanen, "Texture generation over the marker area," in *Proc. Int. Symp. on Mixed and Augmented Reality*, 2006, pp. 253–254.
[2] O. Korkalo, M. Aittala, and S. Siltanen, "Light-weight marker hiding for augmented reality," in *Proc. Int. Symp. on Mixed and Augmented Reality*, 2010, pp. 247–248.
[3] N. Kawai, M. Yamasaki, T. Sato, and N. Yokoya, "Diminished reality for AR marker hiding based on image inpainting with reflection of luminance changes," *ITE Trans. on Media Technology and Applications*, vol. 1, no. 4, pp. 343–353, 2013.
[4] F. I. Cosco, C. Garre, F. Bruno, M. Muzzupappa, and M. A. Otaduy, "Augmented touch without visual obtrusion," in *Proc. Int. Symp. on Mixed and Augmented Reality*, 2009, pp. 99–102.
[5] A. Enomoto and H. Saito, "Diminished reality using multiple handheld cameras," in *Proc. ACCV'07 Workshop on Multi-dimensional and Multi-view Image Processing*, 2007, pp. 130–135.
[6] S. Jarusirisawad, T. Hosokawa, and H. Saito, "Diminished reality using plane-sweep algorithm with weakly-calibrated cameras," *Progress in Informatics*, no. 7, pp. 11–20, 2010.
[7] V. Lepetit and M.-O. Berger, "An intuitive tool for outlining objects in video sequences: Applications to augmented and diminished reality," in *Proc. Int. Symp. on Mixed and Augmented Reality*, 2001, pp. 159–160.
[8] Z. Li, Y. Wang, J. Guo, L.-F. Cheong, and S. Z. Zhou, "Diminished reality using appearance and 3d geometry of internet photo collections," in *Proc. Int. Symp. on Mixed and Augmented Reality*, 2013, pp. 11–19.
[9] T. Tsuda, H. Yamamoto, Y. Kameda, and Y. Ohta, "Visualization methods for outdoor see-through vision," in *Proc. Int. Conf. on Artificial Reality and Telexistence*, 2005, pp. 62–69.
[10] S. Zokai, J. Esteve, Y. Genc, and N. Navab, "Multiview paraperspective projection model for diminished reality," in *Proc. Int. Symp. on Mixed and Augmented Reality*, 2003, pp. 217–226.
[11] J. Herling and W. Broll, "Advanced self-contained object removal for realizing real-time diminished reality in unconstrained environments," in *Proc. Int. Symp. on Mixed and Augmented Reality*, 2010, pp. 207–212.
[12] J. Herling and W. Broll, "High-quality real-time video inpainting with pixmix," *IEEE Trans. Visualization and Computer Graphics*, vol. 20, no. 6, pp. 866–879, 2014.
[13] N. Kawai, T. Sato, and N. Yokoya, "Diminished reality considering background structures," in *Proc. Int. Symp. on Mixed and Augmented Reality*, 2013, pp. 259–260.
[14] D. Simakov, Y. Caspi, E. Shechtman, and M. Irani, "Summarizing visual data using bidirectional similarity," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.
[15] 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. 1–11, 2009.
[16] 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.
[17] N. Kawai and N. Yokoya, "Image inpainting considering symmetric patterns," in *Proc. Int. Conf. on Pattern Recognition*, 2012, pp. 2744–2747.
[18] D. Pavić, V. Schönefeld, and L. Kobbelt, "Interactive image completion with perspective correction," *The Visual Computer*, vol. 22, no. 9, pp. 671–681, 2006.
[19] J.-B. Huang, J. Kopf, N. Ahuja, and S. B. Kang, "Transformation guided image completion," in *Proc. Int. Conf. on Computational Photography*, 2013, pp. 1–9.
[20] H. Kato and M. Billinghurst, "Marker tracking and hmd calibration for a video-based augmented reality conferencing system," in *Proc. Int. Workshop on Augmented Reality*, 1999, pp. 85–94.
[21] D. T. Lee and B. J. Schachter, "Two algorithms for constructing a delaunay triangulation," *Int. Journal of Computer and Information Sciences*, vol. 9, no. 3, pp. 219–242, 1980.

[22] K. Fukunaga and L. Hostetler, "The estimation of the gradient of a density function, with applications in pattern recognition," *IEEE Trans. on Information Theory*, vol. 21, no. 1, pp. 32–40, 1975.
[23] D. L. Massart, L. Kaufman, P. Rousseeuw, and A. Leroy, "Least median of squares: A robust method fo outlier and model error detection in regression and calibration," *Analytica Chimica Acta*, vol. 187, pp. 171–179, 1986.
[24] 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.
[25] P. Pérez, M. Gangnet, and A. Blake, "Poisson image editing," *ACM Trans. on Graphics*, vol. 22, no. 3, pp. 313–318, 2003.
[26] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. Int. Symp. on Mixed and Augmented Reality*, 2007, pp. 225–234.
[27] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shatton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon, "Kinectfusion: Real-time dense surface mapping and tracking," in *Proc. Int. Symp. on Mixed and Augmented Reality*, 2011, pp. 127–136.

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 postdoctoral fellow at the University of California at Berkeley in 2010-2011. He has been an assistant professor at NAIST since 2011.

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 sciences from Nara Institute of Science and Technology (NAIST) in 2001 and 2003. He was an assistant professor at NAIST in 2003-2011. He was a visiting resercher in Czech Technical University in Prague in 2010-2011. He has been an associate professor at NAIST since 2011.

Naokazu Tokoya 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 1986-1987 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.