Received February 24, 2020, accepted March 12, 2020, date of publication March 17, 2020, date of current version March 26, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2981378

Hole-Filling of RealSense Depth Images Using a Color Edge Map

JI-MIN CHO, SOON-YONG PARK, AND SUNG-IL CHIEN
School of Electronics Engineering, Kyungpook National University, Daegu 41566, South Korea

Corresponding authors: Soon-Yong Park (sypark@knu.ac.kr) and Sung-Il Chien (sichien@ee.knu.ac.kr)

This work was supported in part by the Cross-Ministry Giga Korea Project Grant by the Korean Government [Ministry of Science and ICT (MIST)] under Grant GK17P0300, and in part by the BK Plus Project funded by the Ministry of Education under Grant 21A20131600011.

ABSTRACT The RealSense camera frequently includes hole regions—unfilled regions where depth information are missing in the depth image. The holes cause serious problems in applications that use depth sensors and often straddle the background and object simultaneously. Accordingly, a hole must be filled by considering the boundary of the object. Fortunately, the object boundary inside the hole can be estimated by the RealSense color image. This paper proposes a hole-filling method in which filling converges to the object boundary from both border pixels. However, the color edge elements of the object boundary are often missing. Because the missing edge highly influences the hole-filling procedure, the boundary of the object close to the camera should be filled as completely as possible. Consequently, we introduce a simple method of filling the gap between two edge endpoints. Furthermore, we can address the situation in which there are one or more edge pixels between two border pixels. We repeatedly change the filling direction by alternating between the horizontal and vertical directions until no hole remains. Our experiment reveals that the proposed method outperforms other filling methods when numerically comparing the hole-filled depth image with the ground truth image for four methods: root-mean-square error (RMSE), peak signal-to-noise ratio (PSNR), structural similarity index map (SSIM), and correlation coefficient (CC). The proposed method also produces a superior visual when we observe the 3D stereo image pair constructed using an original color image and a left-view color image generated with depth-image-based rendering (DIBR).

INDEX TERMS Hole-filling method, inpainting, Intel RealSense camera D435, depth image, multiview generation.

I. INTRODUCTION

Depth sensors have become important sensors for robotics, machine, and computer vision, but high cost has limited their wider adoption. However, since Microsoft released the low-cost Kinect V1 camera, research using depth sensors has increased, and several low-cost depth sensors have been launched [1]–[6]. Some depth sensors provide not only the depth image but also the corresponding color image, referred to as an RGB-D sensor. RGB-D sensors are used in many applications such as 3D simultaneous localization and mapping (SLAM), 3D reconstruction, virtual reality (VR), gesture analysis, self-driving cars, and RGB-D fusion [7]–[15].

In our experiment, we obtain a depth image and a color image aligned to the depth image from an Intel® RealSenseTM D435 camera (RealSense) [6]. The RealSense is a low-cost RGB-D camera released in 2018 by Intel that can measure depth information within a range of 10 m. The process by which RealSense measures the depth value of each point in the depth image is summarized in [6], [16]. The camera provides a smaller depth value when the distance from the camera to the point in a scene is closer. One serious problem with the RealSense camera in actual applications is unfilled regions in the depth image, typically referred to as *holes*. These holes are pixels of zero value in the depth image: the sensor does not provide any depth information. If the area of the hole is small, we can fill the hole using a simple method. However, when we observe the indoor database, there are not only holes with small areas but also holes with large areas. Based on the areas of the holes in the captured RealSense depth images, the area of the largest hole was approximately 9% of the area of the depth image. Moreover, the average area of the largest holes in the depth images occupies approximately 2.23% of the area

The associate editor coordinating the review of this manuscript and approving it for publication was Larbi Boubchir.

of the depth image. Holes cause serious errors or degradation of quality in applications that use the depth image. For example, when reconstructing the 3D model using the depth image [8] or generating a multiview image based on the depth image using the DIBR [15], the hole significantly influences the quality of the results in these applications. Accordingly, the hole must be filled based on suitable depths and methods of filling the holes that have been actively researched. The method of filling holes is often referred to as the inpainting method, but we call it the hole-filling method.

Hole-filling methods can be classified as diffusion-based method or exemplar-based method [17]. Telea method [18] is a representative diffusion-based method that fills the hole one pixel at a time using surrounding depth values of the hole pixel to be filled, calculating the average depth value to fill in and weighting them by the distance from the hole pixel to the pixel being considered. Chen *et al.* [19] also proposed a diffusion-based method that, unlike Telea method of filling the hole from the border to the center, prioritizes filling on the edge component inside the hole obtained from the RGB-D sensor. Instead of simple averaging, interpolation using a bilateral filter [20] has been introduced. Diffusion-based methods can fill the hole quickly and are efficient for small or thin holes. However, when the area of the hole is large, the filled values frequently tend to blur.

The most popular exemplar-based method, in which the most similar patch of the image is copied into the empty hole region, was proposed by Criminisi *et al.* [21]. This method covers the mask—occasionally referred to as a *patch*—over a hole pixel. Given a hole pixel, the region of the image surrounding the pixel then becomes a patch, scrolling over the entire image to identify the best-matching region. After the patch is found, the depth values of pixels uncovered by a patch corresponding to the hole region inside the patch are copied into the hole region. Criminisi *et al.* found that the filling order is critical in this method because the quality of the hole-filled depth image is influenced significantly by the filling order. The researchers have decided that the filling order should have higher priority when strong edges exist in the patch and the center of the patch inside the hole is close to the hole border. Although they fill the hole more appropriately than diffusion-based methods, the filling speed of the exemplar-based method is slower than the diffusion-based method because the researchers employ relatively large patch scrolling through the entire image to identify maximum similarity.

To solve the speed problem of Criminisi method, Patel and Sarode [22] limit the search region to identify a similar patch. A search region is adjusted to be located close to the hole border because related information is typically found in the nearby pixels. This filling method has similar performance as Criminisi method but fills the holes more quickly. Choi and Ham [23] have worked with holes in moving pictures. The background is first segmented using random walker segmen tation. Because the researchers assume that the holes occur primarily in the background near the left or right sides of the object, the searching region is limited to the segmented background. Furthermore, this method adds one condition for determining the filling order of the Criminisi method, assigning a lower priority when a patch lies on the intersection including the foreground and background region or when a patch is located closer to the foreground than the background. This method demonstrates a filling result superior to Criminisi method because the penetration of depths of an object to the hole is discouraged by the added condition. However, when the hole is straddling the background and object simultaneously, the priority on the background enables the background depth value to cross over the object boundary into the object itself.

Mathematical morphology is a well-known approach for the processing of geometrical structures, with growing research that uses this operator as a hole-filling method [24], [25]. The morphological operator fills both narrow and small holes. A large hole is likely to contain important object boundaries. Because hole-filling methods based on morphology do not consider the object boundary inside the hole, this method can cause the similar problems as those caused by the Telea method.

The unknown object boundary inside the hole of the depth image can be known because an RGB-D sensor also provides the color image. In this paper, to prevent the penetration of depths of an object, we propose a filling method that fills the holes from the hole border toward the object boundary known from the color image. The proposed method consists of three steps: setting the strong edge map, removal of depth error regions, and the hole-filling procedure. To set the strong edge map, we first obtain the color edge components using a Canny edge operator [26]. Then, we overlap the magnitudes and angles computed from the Sobel edge operator [27] into Canny edge pixels because the magnitudes and angles of the Canny edge operator are obtained from a Gaussian-blurred image and are less accurate. The large holes are typically observable around the object close to the camera. Accordingly, it is more desirable to focus on the edges located in the close distance from the camera, and the Sobel magnitudes are weighted inversely based on the distance. Although Canny links the broken edges, many edge elements are often undetected. We binarize the weighted edge and simply fill the gap between two edge endpoints by reaching the gap area and identifying suitable edge elements based on magnitude and direction information. This filled edge map is referred to as the strong edge map. The next step is the removal of depth error regions. We have identified both holes and depth error regions—incorrectly measured depths from the camera to a scene point. The depth error region appears as a small area inside the hole or a region of incorrect depth values spreading out from the object boundary to the background. These cause serious errors with our hole-filling that fills the hole from the hole border to the boundary of an object. Consequently, we remove the depth error regions to reduce this hole-filling error.

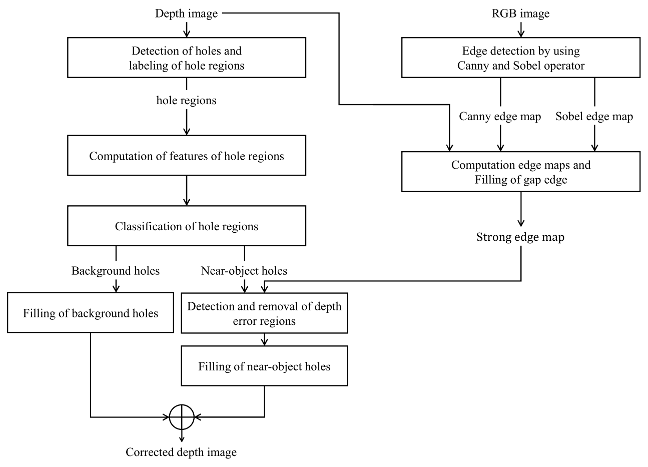
Depth image
RGB image
Detection of holes and labeling of hole regions
Edge detection by using Canny and Sobel operator
hole regions
Canny edge map
Sobel edge map
Computation of features of hole regions
Computation edge maps and Filling of gap edge
Classification of hole regions
Strong edge map
Background holes
Near-object holes
Filling of background holes
Detection and removal of depth error regions
Filling of near-object holes
Corrected depth image

FIGURE 1. Proposed hole-filling procedure.

The first step of the hole-filling procedure is to classify the holes into background holes and near-object holes using the features of the depths of the hole border; holes near the camera must be carefully managed. The background hole is simply filled with the average of the depths of the hole border because the variation of the depth value around the hole is small. Furthermore, the near-object hole is filled horizontally by copying the depth of the hole border from the hole border toward the edge pixel defined in the strong edge map. The remaining part of the hole is then filled vertically. The filling procedure is performed repeatedly in alternating filling directions between the horizontal and vertical directions until the entire of all hole pixels are filled or the holes completely surrounded by the edges remain. The hole completely surrounded by the edges is then filled with the average of the depths of the hole border because it is difficult to identify the optimal depth value. Based on the experimental results, our hole-filling method fills the holes without allowing the depths of an object to penetrate the background.

In Section III, to objectively evaluate the result, we construct a ground truth depth image in which the holes are filled manually. We evaluate the hole-filled depth images by visually comparing them with those of conventional methods. Also, we analyze numerically the quality of hole-filled depth images using the root-mean-square error (RMSE), peak signal-to-noise ratio (PSNR), structural similarity index map (SSIM), and correlation coefficient (CC) [28]–[30]. Finally, we generate a second-view color image based on the ground truth depth image and the hole-filled depth image with depth-image-based rendering (DIBR) [15]. This second-view color image and the original color image are used to construct a 3D stereo image pair, which is observed on a 3D monitor. We use this application as a test method to evaluate the quality of the hole-filled depth images.

II. HOLE-FILLING ALGORITHM

This paper proposes filling the holes or unfilled regions in a depth image that often occur when using the RealSense. Fig. 1 is a block diagram representing the overall hole-filling procedure. First, we detect hole pixels with zero-depth and divide all hole pixels into the connected hole regions with the connected component labeling method [31] because the RealSense SDK fills the depth value of the hole with zero. Next, we construct the edge map and fill the gaps in broken edges. We first classify holes as either background holes or near-object holes. The classification is performed with the average, standard deviation, maximum, and minimum of depth around the hole border. The background hole is simply filled with the average of the depths around the hole border. If the hole is classified as a near-object hole, we identify the depth error regions such as island regions and protruding regions caused by the incorrect measurement of the depth sensor and remove such error regions before these holes are filled. This pre-processing step is essential to avoid filling the hole with an incorrect depth value. Then, because the edges are separating the object from the background, we fill the near-object hole with the neighboring depth from the hole border until the edge points inside the hole. Each step of our method is detailed in the following sections.

A. CLASSIFICATION OF HOLES

The hole region often appears between the object and background, fatally damaging the border of an object in the depth image, necessitating filling the holes with suitable depth values. However, filling the holes is challenging. For simpler processing at a later stage, we must classify holes as either background holes or near-object holes around the objects found near the sensor.

First, we label the connected hole-pixels as a region using a connected component labeling method [31]. Then, we calculate the features of each hole region to enable subsequent classification of the hole regions. We detect the boundary of a hole region with the border following method [32] and label a group of the pixels surrounding the hole boundary as the hole border. Because the hole has lost its depth, we trace the depths of the hole border pixels in the ith hole region. Assuming N_i is the number of the border pixels of the ith hole, we define the depth value of the jth pixel as d_j^i. Next, we calculate the average of surrounding depth values (μ_i), the standard deviation (σ_i), and the maximum and minimum depth values (d_{max}^i and d_{min}^i) from the ith hole region.

We then classify the hole into two groups. The holes typically appear near the object but can also appear inside the object or in the background. When the depth values of the hole located in the background or inside the object are simply replaced by the average of the surrounding depth values (μ_i) without considering the color edge map, this replacement does not cause a serious problem in an application that uses depth values. This is not true for the hole located close to the nearby objects. Thus, we must classify the holes as near-object holes and background holes. Near-object holes appear primarily close to the camera, but near-object holes in the middle distance are often misclassified as background holes. Therefore, we first divide the location of hole regions into three distances based on the average of the surrounding depth values (μ_i): close distance, middle distance, and far distance. We can measure the depths from 0.16 to 10 m with the RealSense camera. In RealSense's depth image, the depth value of a point represents the distance from the camera to the point in millimeters, and the resolution of the depth value is less than 1% of the distance from the camera. Consequently, it is not difficult to sub-divide the location of the hole regions. Close distance refers to an object within 2 m of the camera, middle distance refers to an object 2 to 4 m from the camera, and far distance refers to an object more than 4 m from the camera. The holes in these three groups are now classified into near-object holes and background holes.

First, we prefer to classify most of hole regions in a close distance as near-object holes because they typically appear near the object border, even if the difference between the maximum depth (d^i_{max}) and minimum depth (d^i_{min}) is small. If this difference ($d^i_{max} - d^i_{min}$) is more than 0.3 m, the hole region is classified as a near-object hole. Because our hole-filling algorithm fills the near-object holes more elaborately than the background holes, we must prioritize the near-object holes. Accordingly, the threshold is intentionally determined to be 0.3 m, slightly larger than the maximum value among the averages of the difference ($d^i_{\max} - d^{\mathrm{i}}_{\min}$) of the background holes in the close distance from the experimental analysis. Otherwise, the remaining hole regions are background holes. Based on the empirical analysis, an interval exists within which the above feature ($d^i_{\max} - d^{\mathrm{i}}_{\min}$) does not effectively select the near-object holes. This interval ranges from 2 to 4 m and is referred to as the middle distance. Next, we focus on the middle distance. The hole regions in a middle distance contain more background holes than those in a close distance. Moreover, because the hole regions in the middle distance are farther away from the camera than the hole regions in the close distance, the difference of the maximum and minimum depth value of the hole border becomes small, which classifies most holes as background holes even if they are near-object holes.

To solve this problem, we consider the standard deviation (σ_i) of the pixels surrounding the hole region. Larger standard deviation implies a large variation in the depths of the hole border: this hole is likely to be located near the object. Accordingly, if the standard deviation of a hole region is larger than the threshold, it is classified as a near-object hole. The threshold is determined as the average of standard deviations of all the holes in the middle distance. Finally, the holes in the far distance belong to the background holes. Because human eyes are not sensitive to the boundary of an object located in the far distance from the camera, these hole regions can be filled with the average of the depth values of hole border pixels. This replacement is safely performed for the holes inside the object, classified as a background hole in the first stage. The four examples of the hole classification result are depicted in Fig. 2, in which the near-object holes are marked in red and the background holes are marked in blue.

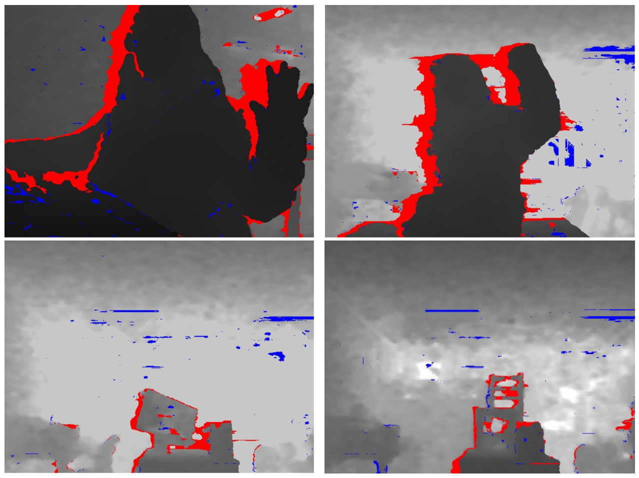
FIGURE 2. **Four examples of classification of holes. Background holes are marked in blue and near-object holes are marked in red. Persons in the first row are located in close distance. Objects in the second row are located in the middle distance.**

B. DEFINITION OF EDGE MAPS FOR HOLE-FILLING PROCEDURE

With the background holes having been filled in the previous step, the remaining step is to fill the near-object holes. We propose to fill the hole from the border pixels to the central point of the holes until we meet the real object boundary. Fortunately, the RealSense camera provides the color image aligned to a depth image. Thus, we must build an edge map from the color image representing the boundary of the object, which might be missing because of the hole in the depth image. The Canny edge operator [26] produces thin edges, which suits our purpose. We then map the magnitude and angle of the Sobel edge operator [27] into the edges produced by the Canny operator. During hole-filling from the hole-border to the edge point, edges of small magnitudes are ignored, with the strong edges used to stop the filling. We first define the edge map in which the edge magnitude of the Sobel operator is mapped to the Canny edge map $M(i,j)$, where (i,j) are the pixel coordinates of the image.

When we are processing near-object holes, we must focus on the strong edges originating from nearby objects rather than the edges in the far distance by modifying the edge map M_w as follows:

$$M_w(i,j) = \begin{cases} \dfrac{M(i,j)}{d(i,j)}, & M \geq T_w \\ 0, & otherwise \end{cases}. \tag{1}$$

where $d(i,j)$ is the depth value of a pixel at (i,j) and becomes larger when the distance from the camera is larger. T_w is the threshold by which the edges are classified as either strong or weak. As depicted in Fig. 3(b), the magnitudes of edges in the background become weaker, whereas the edges in the close distance become stronger. A dominant edge map E_d is a binary image that represents only strong edges in the close distance. The threshold of binarization is selected

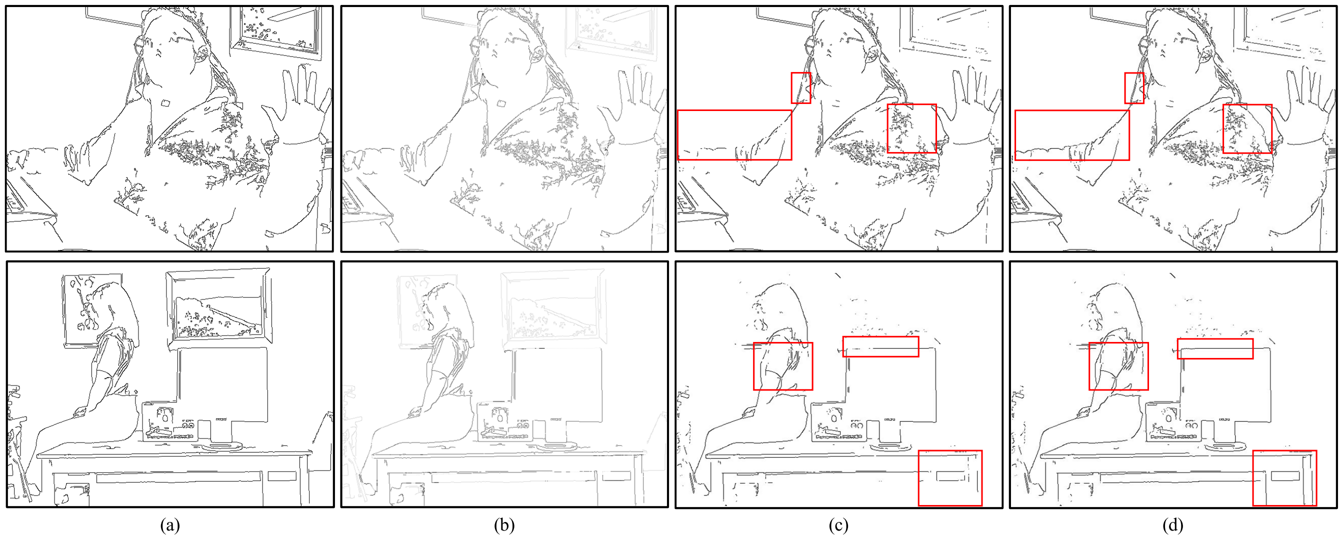

FIGURE 3. Example images of edge maps (a) edge map M, (b) weighted magnitude map M_W, (c) dominant edge map E_d, and (d) strong edge map E_S.

as 50 because a higher threshold would severely break the object boundary. Nonetheless, the missing edges are found often at the object boundary as depicted in Fig. 3(c). These missing edges lead to an incomplete object boundary, which is highlighted with rectangles in Fig. 3(c). The gaps in an incomplete object boundary introduce significant errors in the later hole-filling steps. By lowering the low threshold of the Canny operator, additional missing edges can be identified. However, a low threshold detects not only the critical object boundary but also unwanted stray edges with small magnitudes. To link the edge gap to a complete object boundary, many edge-linking methods have been proposed [33], [34]. These methods are relatively complicated because of the high number of edge branches when tracing the edge elements. Consequently, we must devise a method of identifying as many missing edge elements inside the hole as possible by focusing on the gap of the boundary.

Connected component labeling has been performed on the dominant edge map E_d. Moreover, each connected edge element is defined as an edge line—a linear or cursive line between two endpoints or a fork in shape among three or more endpoints. The length of an edge line is defined as the number of pixels included in this edge line. When we locate the endpoints of each edge line, we calculate the direction of each endpoint, as defined by the average of angles of 5 pixels backtracked from the endpoint, to increase the accuracy of the endpoint angle and minimize the effects of noise. Recall that angle information is also mapped into the Canny edge elements. If the length is shorter than 5 pixels, we ignore the endpoints of this short edge line when defining the edge gap. We then place an additional constraint on a pair of endpoints: the distance between two endpoints should be smaller than approximately 5% of the width of an entire image because gaps that are too wide are not suitable for edge filling. This value was heuristically determined through the careful observation of the edge maps used in our experiment. When one or more candidate endpoints are found for an endpoint we consider, we select the endpoint with the most similar direction to a reference endpoint. Next, we set up a hypothetical line between two endpoints. We search the missing edge elements around the hypothetical line. We can divide the hypothetical line into two groups: a vertical line and a horizontal or oblique line. When the hypothetical line is vertical, the vertical line between two endpoints becomes a hypothetical line. When a hypothetical line is horizontal or oblique, we calculate the y value by repeatedly incrementing the x value by one between the x interval of two endpoints. Because the edge element can be found around the points on the hypothetical line, we search for four or more pixels as a safety margin in the normal direction of this hypothetical line. Among five points, we first check whether their edge magnitudes are larger than a predefined threshold. If we identify more than one selected pixel, we declare the pixel that has the most similar angle to that of a hypothetical line as a new edge.

Occasionally, when we do not identify another endpoint for an endpoint, we consider it to be an isolated endpoint. We then attempt to extend the edge from the endpoint. To prevent the noise pixel from being chosen as an edge, the expansion of the edge pixels should be performed more strictly. The threshold should be larger than the threshold used in filling the edge gap. When the candidate edge pixel has a magnitude larger than the threshold and the same direction as that of the original edge lines, we consider this pixel to belong to the edge line. We repeat this procedure until the length of the newly added line is smaller than 25% of the length of the original line or the gap edge pixel is no longer found. This parameter was selected heuristically through careful experimentation.

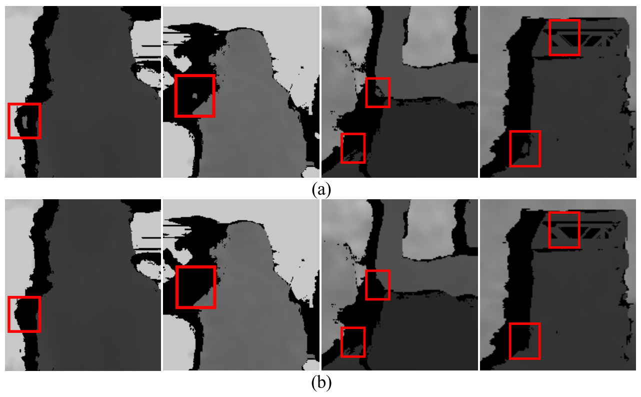

FIGURE 4. Removal of the island regions: (a) original depth images and (b) corrected depth images.

The filled gaps are depicted and highlighted with rectangles in Fig. 3(d). After the gap pixels are filled, the final edge map is referred to as a strong edge map and denoted as E_s.

C. PRE-PROCESSING FOR THE HOLE-FIILING PROCED URE

The RealSense sensor occasionally measures the depth of a scene point incorrectly. The incorrect depth value from the sensor occasionally damages the quality of the resultant image. Therefore, before we start filling the near-object hole, we locate and remove two incorrect depth regions—an island region and a protruding region—in the RealSense depth image.

1) REMOVAL OF THE ISLAND REGION

The small regions are occasionally observed inside a hole as depicted in the highlighted rectangles of Fig. 4(a). Although the area of such a region inside the hole is very small compared to that of the surrounding hole, this region causes significant error in hole-filling. When we fill the hole with depth values surrounding the hole (detailed in Section II-D), this small region is a starting points and produces incorrect depth values. Therefore, we must remove the small region inside a hole. We define this small non-hole region inside a hole as an island region. Because we label these regions with connected component labeling, the area of each region is known. The threshold of removing an island region is 5% of the largest hole area among detected holes. The parameter used to select the island region was carefully determined by observing the depth images used in our experiment.

2) REMOVAL OF THE PROTRUDING REGION

In a RealSense depth image, the depth values of the object occasionally penetrate the boundary into a region touching a hole, as depicted in Fig. 5. This type of depth error causes the depth values of an object to be filled into the hole region, which should be filled with the depth values of the background. This protruding depth leads to significant error in hole-filling and should be detected and removed.

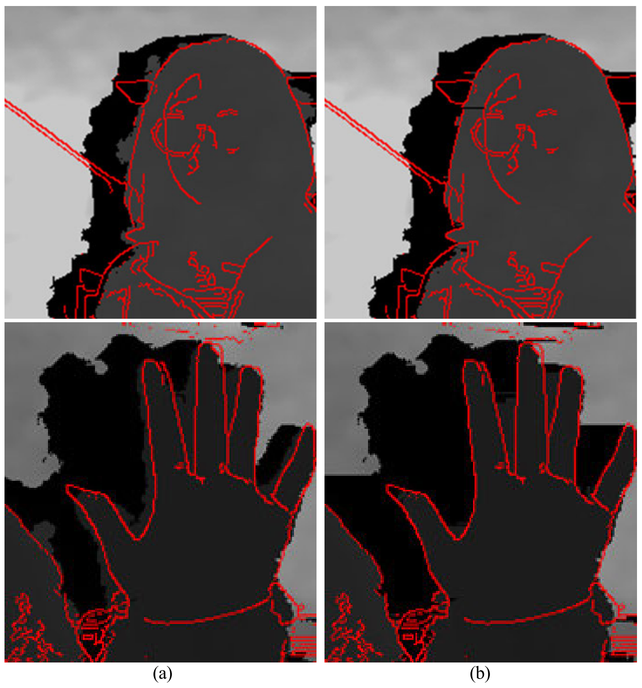

FIGURE 5. Example of removal of the protruding region: (a) depth image with protruding region that is smeared out of the object and (b) depth image in which protruding region is removed.

Strong edges from the color image can function as a boundary separating the object and such a protruding region.

We first identify two border pixels of the hole at the current scanning line and verify whether a strong edge exists between them. If an edge exists, we assume that this depth does not belong to the protruding region, and we proceed to the next lower scanning line. If no edge element exists, this depth can be part of the protruding region. We then proceed from two border pixels to as many as 20 pixels and verify whether an edge element exists. If we identify an edge point, we erase the depth values to that edge point and return these to the hole. We then repeat the above procedure to the bottom of a hole. The results are reported in Fig. 5.

D. FILLING PROCEDURE FOR NEAR-OBJECT HOLES

We have classified the hole region into the background hole and the near-object hole with features obtained from this hole region (described in Section II-A). We have filled the background hole with the average of surrounding depth values because the variation of the depth values around the hole is small. The near-object hole cannot be filled similarly as the background hole because the near-object holes can infiltrate into the object boundary and the depth values around the hole exhibit large variation. Most of the near-object holes appear primarily around an object close to the camera and occasionally in the middle distance. Accordingly, near-object holes must be filled with attention to the boundary of a nearby object. Fortunately, a RealSense camera provides a color image from which we can detect exact edges. Because the

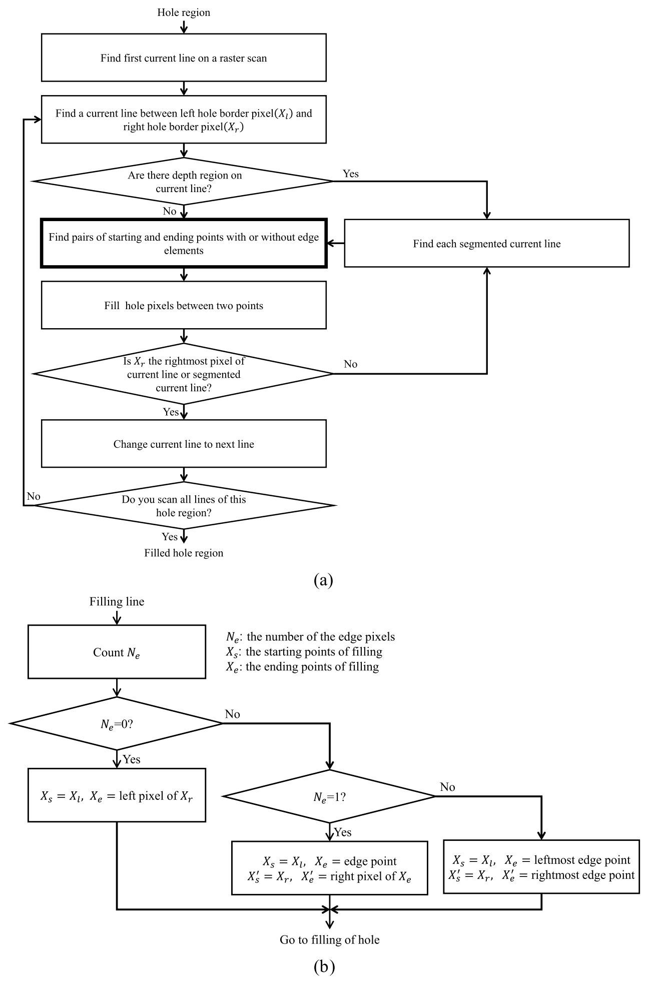
Hole region
Find first current line on a raster scan
Find a current line between left hole border pixel(X_l) and right hole border pixel(X_r)
Are there depth region on current line?
Yes
No
Find each segmented current line
Find pairs of starting and ending points with or without edge elements
Fill hole pixels between two points
Is X_r the rightmost pixel of current line or segmented current line?
No
Yes
Change current line to next line
Do you scan all lines of this hole region?
No
Yes
Filled hole region
(a)
Filling line
Count N_e
N_e: the number of the edge pixels
X_s: the starting points of filling
X_e: the ending points of filling
N_e=0?
No
Yes
$X_s = X_l,\ X_e$ = left pixel of X_r
N_e=1?
No
Yes
$X_s = X_l,\ X_e$ = edge point
$X'_s = X_r,\ X'_e$ = right pixel of X_e
$X_s = X_l,\ X_e$ = leftmost edge point
$X'_s = X_r,\ X'_e$ = rightmost edge point
Go to filling of hole
(b)

FIGURE 6. **Flowcharts of hole-filling procedure: (a) flowchart of the hole-filling procedure and (b) procedure of identifying pairs of starting and ending points with or without edge elements.**

edges located in the background interfere with the hole-filling procedure, we use a strong edge map E_s in which edges in the background have been suppressed and the strong edges around the nearby object have been emphasized (as described in Section II-B). The near-object hole is filled from the hole border toward the strong edge points inside this hole. The near-object hole is filled through several hole-filling passes until the hole is filled completely.

Each pass of the filling procedure is illustrated with a flowchart in Fig. 6. The near-object holes are filled horizontally with the depth values of the hole border line-by- line because the near-object holes typically appear on the sides of an object close to the camera. When the hole region is first found during the raster scan, we identify the left and right hole border pixels (X_l, X_r) and set the current line to be filled with suitable depth values between X_l and X_r. The hole border pixels belong to the outer boundary surrounding a hole but the depth regions occasionally appear inside a hole. This depth region breaks the current line into two or more lines—segmented current lines—each being filled separately. No depth region exists on the segmented line.

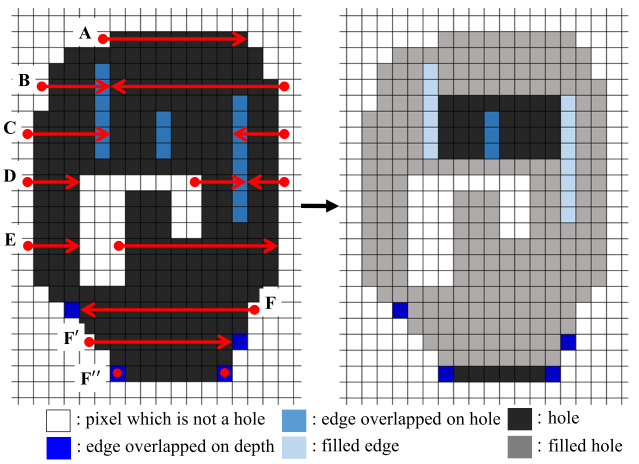
A
B
C
D
E
F
F′
F′′
: pixel which is not a hole
: edge overlapped on hole
: hole
: edge overlapped on depth
: filled edge
: filled hole

FIGURE 7. **Example of first pass in hole-filling procedure: the left image represents starting and ending points and their filling directions during the first pass, and the right image illustrates that only part of the hole is filled during the first pass.**

First, we consider a case with no depth region between X_l and X_r. The next step is to fill the current hole line with a depth value, which requires us to identify starting (X_s) and ending (X_e) points. This situation is complicated because there can be one or more edge lines between X_l and X_r. We impose one important criterion that there should be no starting points on the edge line. The procedure of identifying pairs of starting and ending points with or without the presence of edge lines is detailed in Fig. 6(b). The starting point is located on a border surrounding the hole and the ending point on a hole; the depth value of the sta-rting point is then copied and filled up to the ending point.

Consider the cases in which edge lines possibly exist (Fig. 6(b)). The first case is that no edge line exists. X_s is then X_l and X_e becomes the left pixel of X_r, as depicted in Case A of Fig. 7. The second case is that one edge line exists between X_l and X_r, and $N_e = 1$ as in Fig. 6(b). We have two pairs of starting and ending points, as detailed in Fig. 6(b), which corresponds to Case B of Fig. 7. Filling converges to the edge element from both border pixels. The remaining case is that $N_e > 1$, (two or more edge elements exist). We define two pairs of starting and ending points including X_r and X_l. This corresponds to Case C of Fig. 7, in which we have unfilled holes to be filled later.

Next, consider the situation where depth regions exist on a current line. We segment the current line into two or more lines—segmented current lines—in which no depth region exists. Each segmented current line is fed to the procedure of identifying starting and ending points of filling. Cases D and E in Fig. 7 illustrate this situation. Case E refers to a filling direction of left to right. A near-object hole typically has a greater chance of appearing in the left background than in the right background of objects close to the camera. Case D is the complicated case where an edge element exists between the inner depth region and hole border. Finally, we have a situation omitted from Fig. 6(b) because of its complexity. This situation can be illustrated in Case F of Fig. 7, in which

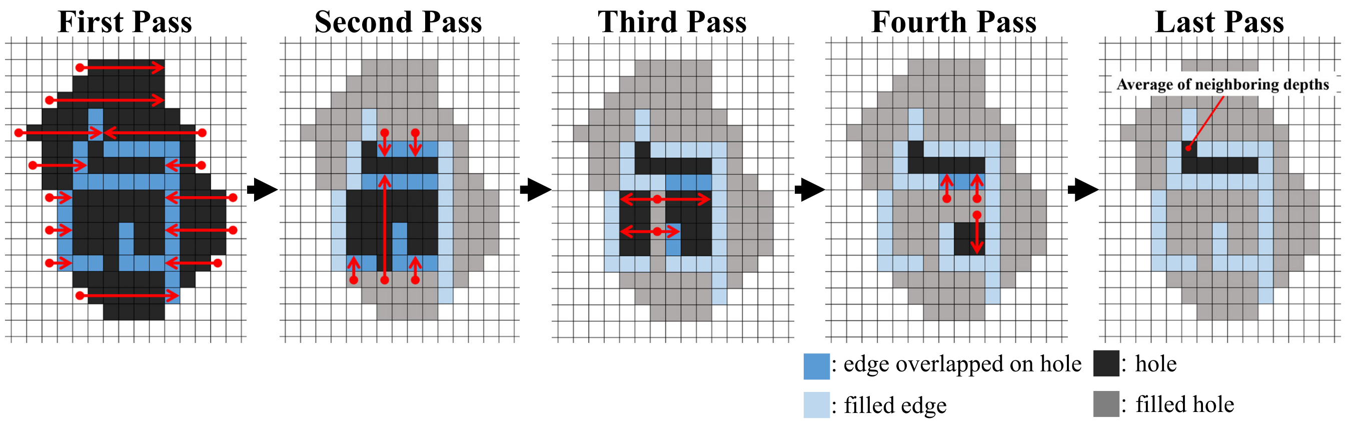

FIGURE 8. Several passes of the hole-filling procedure. The arrow represents the direction of filling, with filling starting from the hole border and proceeding to the edge depicted in blue.

edge elements are exactly on the border pixels because the border pixel on an edge element cannot be chosen as a starting point, resulting in the other border pixel being chosen instead (Cases F and F'). If two edge elements are on the two border points that correspond to Case F" in Fig. 7, edge filling does not occur.

The right image in Fig. 7 illustrates that the hole region of the left image has been filled by this procedure, but some part of hole is not filled because the holes are protected by the outermost edge elements. We then change the filling direction from horizontal to vertical. The hole-filling of the second pass is the same as the first pass except for the filling direction. We repeat the filling procedure until the starting point is no longer found. The filling direction alternates between the horizontal and vertical directions, as depicted in Fig. 8. When the hole is completely surrounded by the edge elements, we cannot select the starting point even though the filling procedure is still unfinished. Therefore, we fill the remaining hole region in the last pass, as depicted in Fig. 8. Because it is difficult to determine the optimal depth value, we employ the 5 × 5 mask over each pixel of the unfilled hole region. The depth pixels covered by the mask are averaged and used to fill the hole. Their mask position moves along the hole border and moves toward the center of an unfilled hole region.

III. EXPERIMENTAL RESULTS AND DISCUSSION

We use the depth image and color image for our hole-filling procedure. The RealSense camera offers a depth image with a resolution of 1280 × 720 and a color image with a resolution of 1920 × 1080. However, because specifications such as the field of view (FOV) and resolution of color and depth cameras are different, we must align the coordinates between depth and color images using the RealSense SDK. The resolutions of the aligned color and depth images are changed to 640 × 480, because the common part of two aligned images is extracted. Although reduced, the extracted depth image's resolution is at least as high as the Kinect V2 (512 × 424), Xtion (640 × 480), and Orbbe Pro (640 × 480).

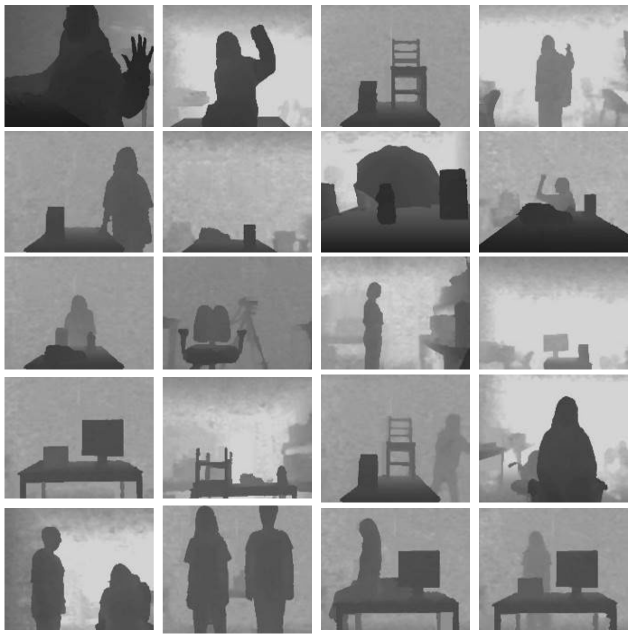

FIGURE 9. Examples of depth images belonging to the ground truth data set.

To evaluate the corrected depth images with the Criminisi, Telea, and proposed methods, we have constructed the image data set for the indoor environment. The image data set consists of 113 pairs of depth and color images obtained from the RealSense camera. For the quantitative evaluation of the experimental results, we filled the holes manually and constructed ground truth images from 55 randomly-selected depth images. Fig. 9 illustrates examples of the ground truth images. We performed the experiment using the C++ programing language and Visual Studio 2017. We used an Intel Core^{TM} i7-7700 (3.6 GHz) processor with 16GB of RAM. First, as described in Section III-A, we fill the holes of the input depth images and compare the result with the ground truth images in terms of the root-mean-square error (RMSE), peak signal-to-noise ratio (PSNR), structural similarity index

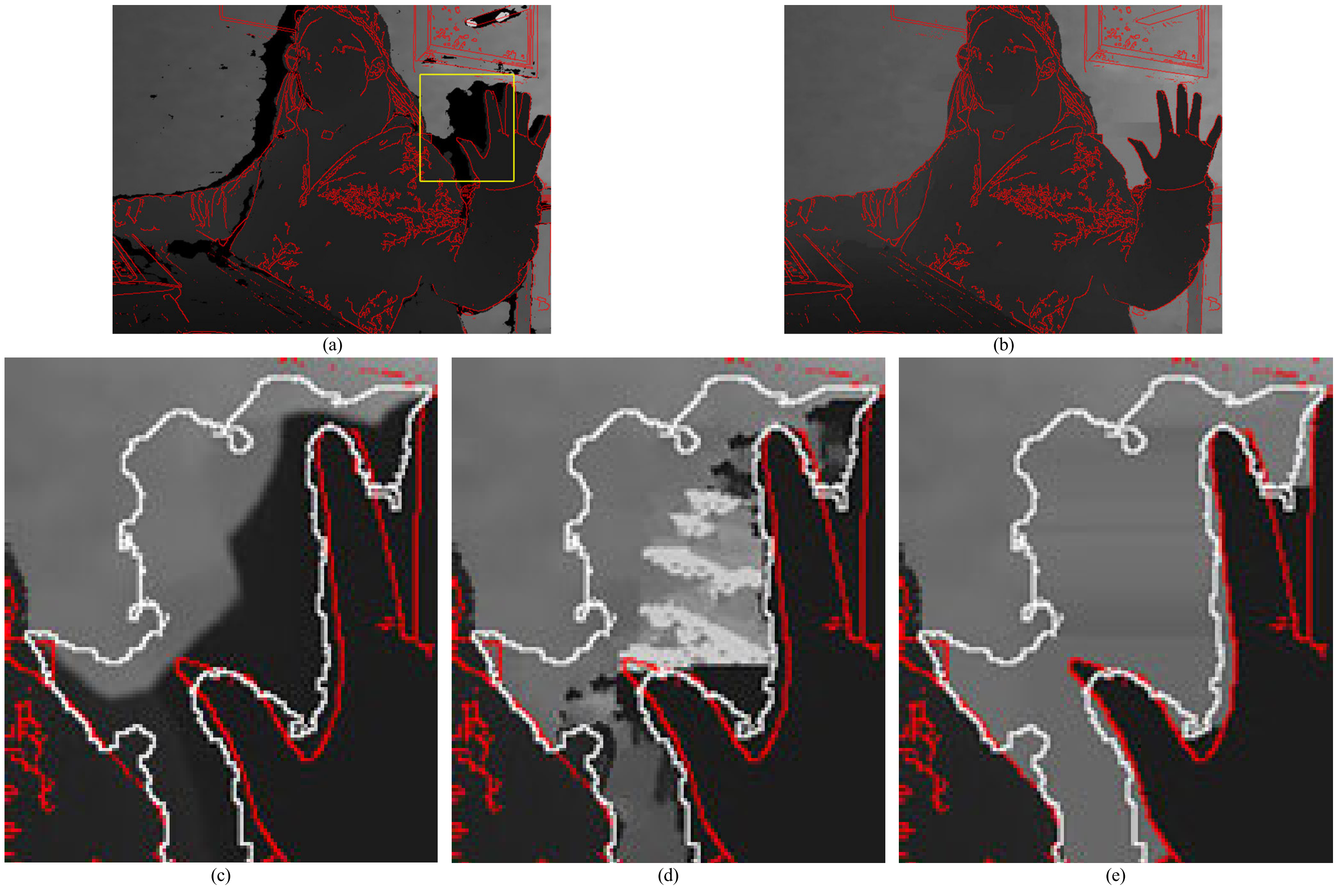

FIGURE 10. **Corrected depth images: (a) original depth image, (b) ground truth depth image, and hole-filled depth images with (c) Telea method, (d) Criminisi method, and (e) the proposed method. Hole borders are in white.**

map (SSIM), and correlation coefficient (CC). As described in Section III-B, we generate a second-view color image with the ground truth image and the hole-filled depth image to construct a 3D stereo image pair, which can be observed on a 3D monitor. This application is not only interesting in itself but also useful for testing the visual quality of the hole-filled depth image.

A. COMPARISON OF CORRECTED DEPTH IMAGES

To evaluate the performance of the proposed hole-filling FIGURE 9. Examples of depth images belonging to the ground truth data set First PassSecond PassThird PassFourth PassLast Pass: edge overlapped on hole: hole: filled hole: filled edge method, we compare the proposed method with the Liu *et al.* [16] and Chen *et al.* [19] methods. We can identify which hole-filling method demonstrates superior performance by visually comparing the ground truth depth image with hole-filled depth images from the three methods. Moreover, we can numerically evaluate the quality of hole-filled depth images in terms of RMSE, PSNR, SSIM, and CC [28]–[30].

We visually evaluate the hole-filled depth images from the three methods. Figs. 10 and 11 are examples of the hole-filled depth images which the Canny edges obtained from the color image overlap (depicted in red). In each figure, we include the depth image with holes filled with 0, the ground truth depth image, and magnified hole-filled depth images from the three methods. First, we analyze an example image in which a person is very close to the camera, as depicted in Fig. 10. The large holes appear on the left side of the face and the left side of the left hand as depicted in Fig. 10(a). We focus on the area highlighted by the yellow rectangle. The hole inside the yellow rectangle is located in the background between the left shoulder and hand. Telea calculates the depth value to fill the hole pixel one at a time. Telea selects the pixels located within a certain radius of the hole pixel to estimate the depth value. Furthermore, the depth value is determined by the average of the depths weighted based on the distance from the hole pixel to the depth pixel under consideration. Because Telea fills the hole from the hole border toward the center of the hole, the border depths spread toward the center. If we observe the area enclosed by the hole border marked in white in Fig. 10(c), a serious problem is that the depth values of an object (boundary represented by a red line) has been smeared into the hole region, moving the resultant object boundary toward the center of the hole.

Because the hole-filling order has a strong influence on the results of the hole-filling procedure, Criminisi introduced

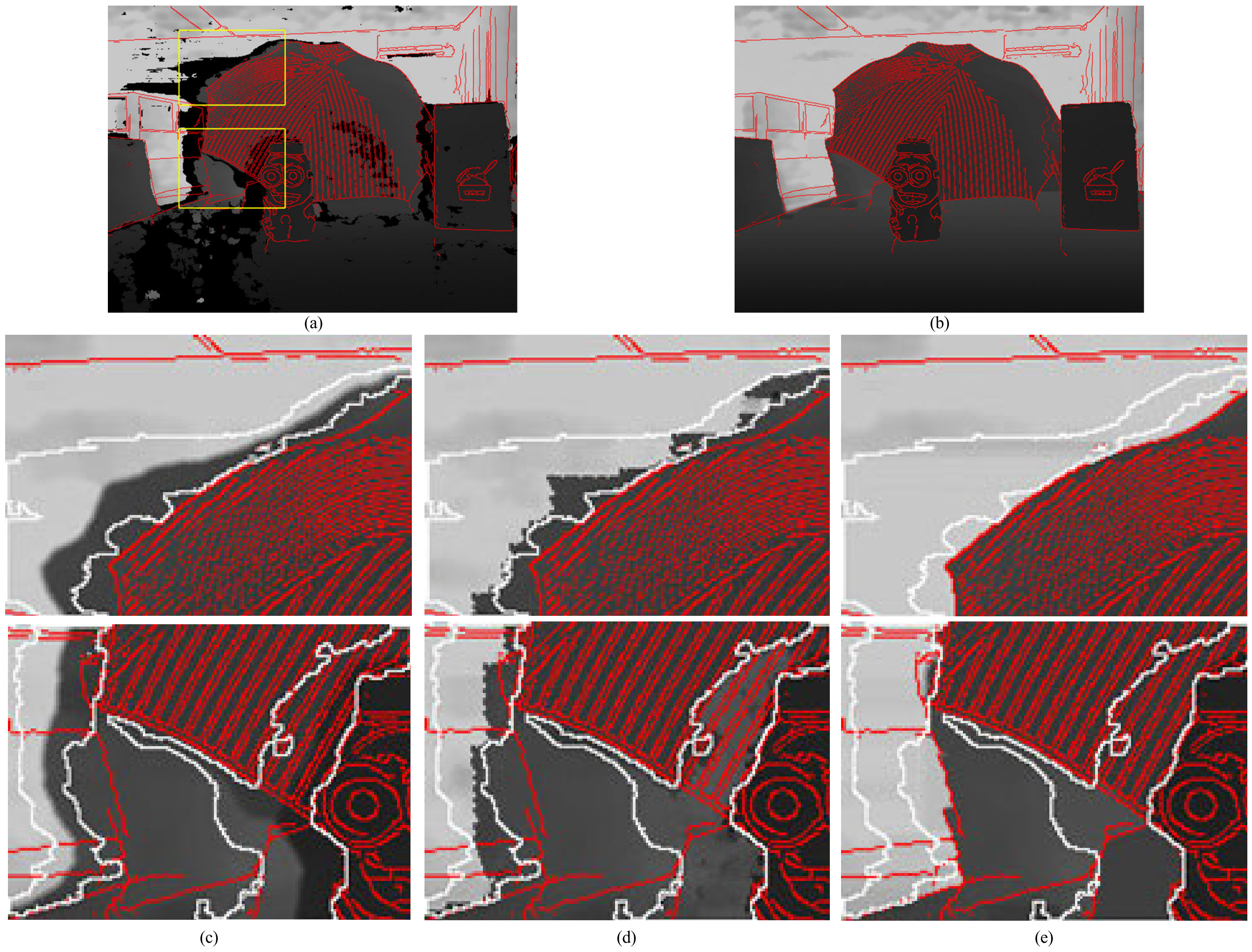

FIGURE 11. Corrected depth images: (a) original depth image of a doll in front of the umbrella, (b) ground truth depth image, and hole-filled depth images with (c) Telea method, (d) Criminisi method, and (e) the proposed method. Hole borders are in white.

order to determine the priority of hole-filling. Criminisi first calculates the two terms—confidence and data terms—to decide which hole pixel is filled first. Order is determined based on the product of two terms; hole-filling starts around the hole border and the pixel of which the edge direction is the most similar to the normal direction of the hole boundary. Criminisi selects the pixel with the highest priority. After the pixel is selected and a rectangle patch around it is copied, the patch is slid over the entire image and a region most similar to the patch is found. This procedure is repeated until the entire hole is filled completely. This method demonstrates that depth values around an object boundary (represented by a red line) are less smeared toward the center of the hole region than in Telea method. However, in Fig. 10(d), the hole on the left side of the index finger is filled with an incorrect depth value because the white part inside the picture frame at the corner of the right upper side of Fig. 10(a) is determined as the most similar patch. This incorrect patch can lead to erroneous results because it is used to identify a subsequent patch, depicted by white patterns in Fig. 10(d). Identifying the correct order of hole-filling is difficult. In contrast, as depicted in Fig. 10(e), the hole-filled depth image from the proposed method displays the hole filled with suitable depth values with the depths of the object boundary not smeared toward the background.

We then focus on the example where more than one object is located in the middle or far distance. As depicted in the first row of Fig. 11(c), Telea fills the part of the hole located at the left side of the umbrella with the incorrect depth value from the umbrella. When we observe the second row of Fig. 11(c), the hole of the umbrella located at the left of the doll is filled with depth diffused from the doll. Hole-filling by Criminisi method, as depicted in Fig. 11(d), is not suitable because an incorrect patch after being attached has a strong influence on patch-pasting at the later stage. However, the proposed method fills the hole more effectively, with the resultant image comparable to the ground truth depth image.

To numerically evaluate hole-filling, we calculate the RMSE, PSNR, SSIM, and CC from the ground truth image and hole-filled depth images. Because the hole-filled depth

TABLE 1. Method comparison based on ground truth image with RMSE, PSNR, SSIM, and CC.

Example image	Telea method [18]				Criminisi method [21]				Proposed method			
	RMSE	PSNR	SSIM	CC	RMSE	PSNR	SSIM	CC	RMSE	PSNR	SSIM	CC
Fig. 10	33.187	17.712	0.766	0.658	30.147	18.546	0.762	0.741	12.808	25.982	0.920	0.939
Fig. 11	48.450	14.425	0.715	0.742	68.304	11.442	0.724	0.751	22.060	21.259	0.833	0.885
Average of 55 images	37.340	17.156	0.731	0.750	36.112	17.286	0.745	0.786	23.892	21.106	0.845	0.897

image is modified only for the hole pixels, calculating the RMSE, PSNR, SSIM, and CC for the entire depth image is not suitable. Accordingly, the newly-filled hole areas are mapped to the ground truth image and the common area is cropped and defined as a set H to be used for our comparison. The RMSE is defined as:

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum\nolimits_{(x,y)\in H}\left(d_G(x,y)-d_i(x,y)\right)^2}, \tag{2}$$

where n is the number of pixels included in H and (x, y) are the pixel coordinates. $d_G(x, y)$ is the depth value of ground truth image and $d_i(x, y)$ is that of the hole-filled depth image. The PSNR is defined as:

$$\mathrm{PSNR} = 10\log_{10}\left(\frac{\max(d_i)^2}{\mathrm{MSE}}\right), \tag{3}$$

where $\max(d_i)$ is the maximum depth value of a hole-filled depth image and MSE is the mean square error between a hole-filled depth image and ground truth image. The SSIM measures the structural similarity of the image quality between two images and is defined as:

$$\mathrm{SSIM} = \frac{\left(2\mu_x\mu_y + C_1\right)\left(2\sigma_{xy} + C_2\right)}{\left(\mu_x^2 + \mu_y^2 + C_1\right)\left(\sigma_x^2 + \sigma_y^2 + C_2\right)}, \tag{4}$$

where μ_x and μ_y are the means of depth values d_G and d_i, σ_x^2 and σ_y^2 are the variances of d_G and d_i, and σ_{xy} is the covariance between d_G and d_i.

Table 1 shows the RMSE, PSNR, SSIM, and CC of the two hole-filled depth images for Telea, Criminisi, and proposed methods. Here, the average of RMSEs, PSNRs, SSIMs, and CCs is included for 55 hole-filled depth images with respect to the ground truth images. If the filled region of a hole more closely resembles the region of the ground truth depth image, the RMSE decreases, PSNR increases, and SSIM and CC converge to 1. When comparing the three methods, the proposed method demonstrates superior performance in terms of RMSE, PSNR, SSIM, and CC. We can conclude that the proposed method fills the hole most similarly with respect to the ground truth depth image.

We measure the CPU time of hole-filling respect to the 55 depth images to compare the speeds of the three methods. Table 2 reveals that Criminisi method is the slowest because it compares the large patch across the entire image. For average CPU time, Telea method is the fastest but it is not significantly faster than the proposed method. When we consider that the proposed method demonstrates superior hole-filling and

TABLE 2. CPU times of hole-filling procedure for three methods.

Example image	Telea method [18] (ms)	Criminisi method [21] (ms)	Proposed method (ms)
Fig. 10	363	2352	303
Fig. 11	397	6265	328
Average of 55 images	261.700	2522.927	294.318

the CPU time of hole-filling is comparable to the fast Telea method, the proposed method is a suitable choice for hole-filling.

B. EVALUATION OF GENERATED MULTIVIEW IMAGES

To construct 3D stereo imaging, we require two color images of different viewpoints. The RealSense camera provides a color image and a depth image, but we can generate a second-view image with the DIBR algorithm [15]. DIBR moves the point (x, y) of the original image to the point (x', y') of the

second-view image using the following equation:

$$x' = x + s\frac{t}{2}\left(\frac{Z_{\max}-Z_{\min}}{Z_{\min}}\right)\left(\frac{255-I_d}{255}\right), \quad y' = y, \tag{5}$$

where s is 1 when the second-view image is left-view and -1 when the second-view image is right-view, t is the disparity of two viewpoints, and $Z_{\max}$, $Z_{\min}$ are the maximum and minimum value of the distance from the camera to the object. I_d is the normalized value of the depth value of a current pixel, which is adjusted to a range of 0 to 255. Because the IR camera and the RGB camera are arranged side-by-side in a RealSense camera, the y-axis is the same. In this experiment, DIBR generates the left-view image with the color image and the hole-filled depth image. According to Equation 6, in generating a second-view image, the point near the camera corresponding to a smaller I_d moves farther to the right than the point in the background. Consequently, another undefined region—also referred to as a hole—appears between the object and the background. Recently, a method was proposed to fill in this type of hole [35] and is adopted to construct the final second-view image.

This type of multiview generation based on depth and color images is not only an interesting application but also a suitable method of evaluating the quality of our hole-filling results because we can inspect visually stereo images displayed on 3D monitors. In this experiment, a left-view image is newly generated as a second-view image. We observe the 3D stereo images with an LG 3D monitor W2363D (120 Hz)

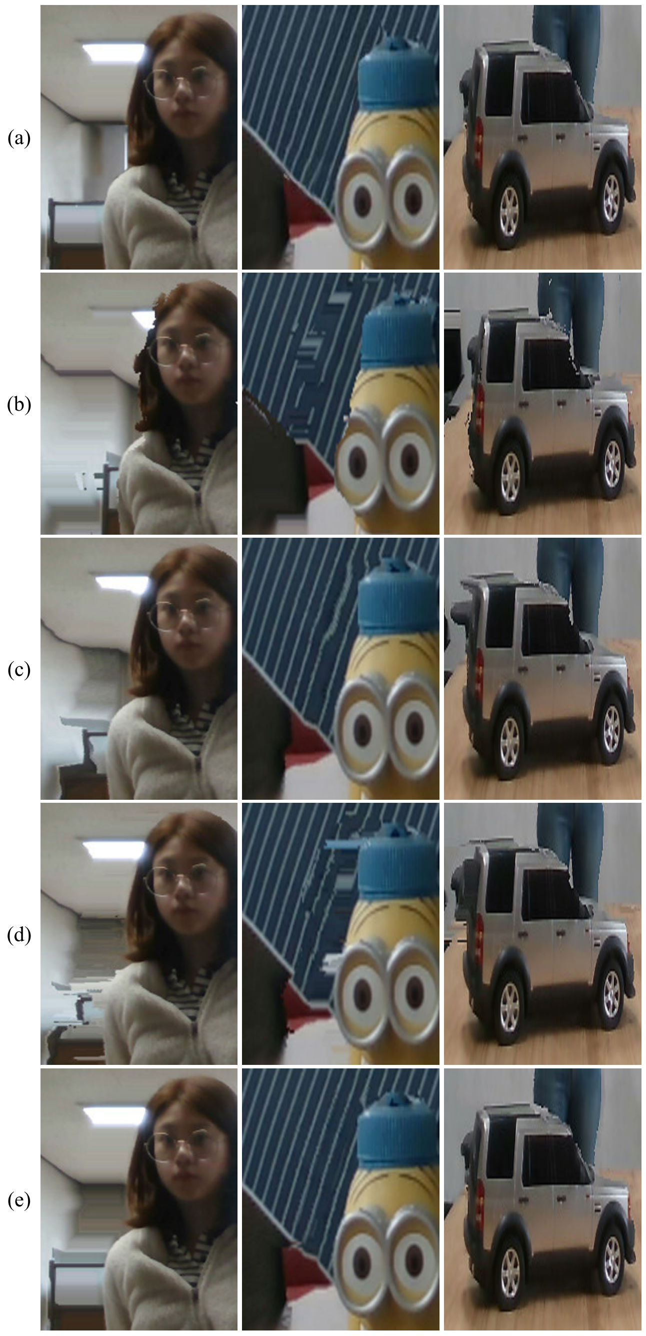

FIGURE 12. Generated left-view images with (a) ground truth depth, (b) original depth image with hole, and corrected depth images with (c) Telea method, (d) Criminisi method, and (e) the proposed method.

and NVIDIA 3D vision glasses to test the quality of the results of hole-filling by various methods.

For display clarity, the parts of the generated left-view images are enlarged in Fig. 12. Fig. 12(a) illustrates the result of applying DIBR with the ground truth image and functions as a reference to evaluate the three hole-filling algorithms. Fig. 12(b) illustrates the results when unfilled depth images are used. Originally, the depth value of such a hole pixel is given as 0. Because the depths of these holes are incorrectly assigned, these holes cause erroneous coordinate shifts and seriously degrade the left-view images, as depicted in Fig. 12(b). The first column of Fig. 12 illustrates the doll in front of the blue striped umbrella. In Telea's left-view image, the stripes of the umbrella are more effectively restored than Criminisi's left-view image, but several stripes of the umbrella still bend toward the doll because the depth value of the doll has spread through the umbrella. For Criminisi's left-view image, the stripes of the umbrella are much distorted and the part of the left side of the doll is stretched to the left side. Because the image of the first column of Fig. 12 is part of Fig. 11, these incorrect assignments of depth values are depicted in Fig. 11(c) and (d). In contrast, the proposed method restores the stripes of the umbrella similar to the DIBR result of ground truth. We can consider this case in terms of numerical error of three filling methods with respect to the ground truth depth image, as summarized in Table 1. As expected, the RMSE of the proposed method is smallest and its PSNR is largest among the three hole-filling methods.

Consider the images of the second column of Fig. 12. When the DIBR result of the ground truth image is compared with that of Telea's depth image, Telea's image reveals that the fluorescent lamp in the ceiling is shifted excessively to the right. Especially for the partition located in the bottom-left corner of the figure, the DIBR results demonstrate serious errors for the cases of Telea and Criminisi methods because these methods could not restore the object boundary in the depth image. Furthermore, because the depth values of the object are smeared out to the background, the partition moves to the right along with the objects. In contrast, our method successfully performs hole-filling, and the partition itself remains nearly intact. The proposed method reconstructs this part accurately as depicted in the last row of this column. In the third column, Telea's and Criminisi's left-view images reveal that the shape of the convex protruding part on the upper-left side of the car is completely different from that of the ground truth's left-view car. In this case, when holes are filled, the rear part of the car is assigned to the background and remains in that place, while the remaining part of the car moves to the right. The proposed method demonstrates superior performance and produces similar results as the ground truth case.

Finally, we observe a pair of stereo images: the left-view color image and its original color image with the 3D monitor and 3D vision glasses. For the Telea and Criminisi methods, part of the background near the object boundary is attached to the object and has moved with the object, which frequently causes unpleasantness to our eyes. Because the quality of the object boundary of the proposed method is well-preserved, we experience the stereoscopic effect much more vividly on a 3D monitor.

IV. CONCLUSION

In this paper, we proposed a method of filling holes in a RealSense depth image with the help of edge information from the color image. Based on observations, the holes are typically straddling an object boundary between the object and the background. We designed the algorithm to fill the hole in a depth image such that the hole is filled from two border pixels to the object boundary inside the hole. Because the uncompleted boundaries of the object cause filling errors in the proposed hole-filling procedure, we fill the gap between two endpoints of the broken edge. Our filling method is also designed to address the case in which there are more than two edge pixels between two border pixels. Accordingly, we could prevent the depths of the background or object from being smeared beyond the object boundary in a hole-filled depth image.

By filling the hole from the hole border toward the object boundary, the hole-filled depth image of our method exhibits the most similar results to the ground truth image in a visual test. Comparing the RMSE, PSNR, SSIM, and CC of the three methods, the proposed method demonstrates visually and numerically superior performance. Moreover, we generate a second-view color image based on the hole-filled depth image using the DIBR method. When we observe the 3D stereo image pair constructed with the original color and our second-view color images, the proposed method has the highest performance; we feel significantly more comfortable in 3D stereoscopic observation and experience more vivid stereoscopic effects.

REFERENCES

[1] G. Halmetschlager-Funek, M. Suchi, M. Kampel, and M. Vincze, "An empirical evaluation of ten depth cameras: Bias, precision, lateral noise, different lighting conditions and materials, and multiple sensor setups in indoor environments," *IEEE Robot. Autom. Mag.*, vol. 26, no. 1, pp. 67–77, Mar. 2019, doi: 10.1109/MRA.2018.2852795.

[2] Structure by Occipital. (2013). *Precise 3D Vision for Embedded Applications*. [Online]. Available: https://structure.io/embedded

[3] ASUS. (2011).*Specifications of Xtion Pro Live: ASUS*. [Online]. Available: https://www.asus.com/3D-Sensor/Xtion_PRO_LIVE

[4] Microsoft. (2014). *Kinect for Windows*. [Online]. Available: https://developer.microsoft.com/en-us/windows/kinect/hardware

[5] ORBBEC. (2015). *Astra Series: Product Introduction*. [Online]. Available: https://orbbec3d.com/product-astra-pro

[6] Intel RealSense. (Oct. 2019). *Intel RealSense D400 series Product Family Datasheet*. [Online]. Available: https://dev.intelrealsense.com/docs/intel-realsense-d400-series-product-family-datasheet

[7] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping (SLAM): Part II," *IEEE Robot. Autom. Mag.*, vol. 13, no. 2, pp. 99–110, Jun. 2006, doi: 10.1109/MRA.2006.1678144.

[8] R. A. Newcombe, A. Fitzgibbon, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, and S. Hodges, "KinectFusion: Real-time dense surface mapping and tracking," in *Proc. 10th IEEE Int. Symp. Mixed Augmented Reality*, Basel, Switzerland, Oct. 2011, pp. 127–136, doi: 10.1109/ISMAR.2011.6092378.

[9] S. H. Khan, M. Bennamoun, F. Sohel, R. Togneri, and I. Naseem, "Integrating geometrical context for semantic labeling of indoor scenes using RGBD images," *Int. J. Comput. Vis.*, vol. 117, no. 1, pp. 1–20, Mar. 2016, doi: 10.1007/s11263-015-0843-8.

[10] P. Henry, "RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments," in *Proc. Int. Symp. Express Robot.*, 2014, pp. 477–491, doi: 10.1007/978-3-642-28572-1_33.

[11] X. Wang and J. Tanaka, "GesID: 3D gesture authentication based on depth camera and one-class classification," *Sensors*, vol. 18, no. 10, p. 3265, Sep. 2018, doi: 10.3390/s18103265.

[12] C. Häne, L. Heng, G. H. Lee, F. Fraundorfer, P. Furgale, T. Sattler, and M. Pollefeys, "3D visual perception for self-driving cars using a multi-camera system: Calibration, mapping, localization, and obstacle detection," *Image Vis. Comput.*, vol. 68, pp. 14–27, Dec. 2017, doi: 10.1016/j.imavis.2017.07.003.

[13] W. Zhou, Y. Lv, J. Lei, and L. Yu, "Global and local-contrast guides content-aware fusion for RGB-D saliency prediction," *IEEE Trans. Syst., Man, Cybern. Syst.*, to be published, doi: 10.1109/TSMC.2019.2957386.

[14] J. Yuan, W. Zhou, and T. Luo, "DMFNet: Deep multi-modal fusion network for RGB-D indoor scene segmentation," *IEEE Access*, vol. 7, pp. 169350–169358, 2019, doi: 10.1109/access.2019.2955101.

[15] C. Fehn, "Depth-image-based rendering (DIBR), compression, and transmission for a new approach on 3D-TV," *Proc. SPIE Stereoscopic Displays Virtual Reality Syst.*, vol. 5291, pp. 93–104, May 2004, doi: 10.1117/12.524762.

[16] S. Liu, D. Gao, P. Wang, X. Guo, J. Xu, and D.-X. Liu, "A depth-based weighted point cloud registration for indoor scene," *Sensors*, vol. 18, no. 11, p. 3608, 2018.

[17] C. Guillemot and O. Le Meur, "Image inpainting : Overview and recent advances," *IEEE Signal Process. Mag.*, vol. 31, no. 1, pp. 127–144, Jan. 2014, doi: 10.1109/MSP.2013.2273004.

[18] A. Telea, "An image inpainting technique based on the fast marching method," *J. Graph. Tools*, vol. 9, no. 1, pp. 23–34, Jan. 2004, doi: 10.1080/10867651.2004.10487596.

[19] L. Chen, H. Lin, and S. Li, "Depth image enhancement for Kinect using region growing and bilateral filter," in *Proc. 21st Int. Conf. Pattern Recognit. (ICPR)*, Tsukuda, Japan, Nov. 2012, pp. 3070–3073.

[20] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Proc. 6th Int. Conf. Comput. Vis.*, Bombay, India, Jan. 1998, pp. 839–846.

[21] A. Criminisi, P. Perez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *IEEE Trans. Image Process.*, vol. 13, no. 9, pp. 1200–1212, Sep. 2004, doi: 10.1109/TIP.2004.833105.

[22] J. Patel and T. K. Sarode, "Exemplar based image inpainting with reduced search region," *Int. J. Comput. Appl.*, vol. 92, no. 12, pp. 27–33, Apr. 2014, doi: 10.5120/16063-5295.

[23] S. Choi, B. Ham, and K. Sohn, "Space-time hole filling with random walks in view extrapolation for 3D video," *IEEE Trans. Image Process.*, vol. 22, no. 6, pp. 2429–2441, Jun. 2013, doi: 10.1109/TIP.2013.2251646.

[24] H. Guo, N. Ono, and S. Sagayama, "A structure-synthesis image inpainting algorithm based on morphological erosion operation," in *Proc. Congr. Image Signal Process. (CISP)*, Hainan, China, Nov. 2008, pp. 530–535.

[25] J. A. A. Salido and C. Ruiz, "Using morphological operators and inpainting for hair removal in dermoscopic images," in *Proc. Comput. Graph. Int. Conf. (CGI)*, Yokohama, Japan, 2017, pp. 1–6.

[26] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vols. PAMI–8, no. 6, pp. 679–698, Nov. 1986, doi: 10.1109/TPAMI.1986.4767851.

[27] I. Sobel, "An isotropic 3×3 gradient operator," in *Machine Vision for Three-Dimensional Scenes*, H. Freeman ed. New York, NY, USA: Academic, 1990, pp. 376–379.

[28] S. David, "Image compression," in *Data Comparison: The Complete Reference*, 4th ed. New York, NY, USA: Springer, 2007, pp. 270–283.

[29] Q. Huynh-Thu and M. Ghanbari, "The accuracy of PSNR in predicting video quality for different video scenes and frame rates," *Telecommun. Syst.*, vol. 49, no. 1, pp. 35–48, Jan. 2012, doi: 10.1007/s11235-010-9351-x.

[30] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004, doi: 10.1109/TIP.2003.819861.

[31] C. Grana, D. Borghesani, and R. Cucchiara, "Optimized block-based connected components labeling with decision trees," *IEEE Trans. Image Process.*, vol. 19, no. 6, pp. 1596–1609, Jun. 2010, doi: 10.1109/TIP.2010.2044963.

[32] M. Sonka, V. Hlavac, and R. Boyle, "Segmentation I," in *Image Processing, Analysis, and Machine Vision*, 3rd ed. London, U.K.: Chapman-Hall, 2007, pp. 191–195.

[33] A. Jevtic, I. Melgar, and D. Andina, "Ant based edge linking algorithm," in *Proc. 35th Annu. Conf. IEEE Ind. Electron.*, Porto, Portugal, Nov. 2009, pp. 3353–3358.

[34] H. Kimm, N. Abolhassani, and F. Lee, "Edge detection and linking pattern analysis using Markov chains," in *Proc. IEEE 16th Int. Conf. Comput. Sci. Eng.*, Sydney, NSW, Australia, Dec. 2013, pp. 1146–1152.

[35] S.-W. Nam, K.-H. Jang, Y.-J. Ban, H.-S. Kim, and S.-I. Chien, "Hole-filling methods using depth and color information for generating multiview images," *ETRI J.*, vol. 38, no. 5, pp. 996–1007, Oct. 2016, doi: 10.4218/etrij.16.0116.0062.

JI-MIN CHO received the B.S. degree from the Department of Electronics Engineering, Kyungil University, Kyungsan, South Korea, in 2015, and the M.S. degree from the School of Electronics Engineering, Kyungpook National University, Daegu, South Korea, in 2017, where she is currently pursuing the Ph.D. degree. Her research interests include image processing and 3-D vision.

SOON-YONG PARK received the B.S. and M.S. degrees from the School of Electronics Engineering, Kyungpook National University, Daegu, South Korea, in 1991 and 1993, respectively, and the Ph.D. degree from Stony Brook University, NY, USA, in 2003. He was with the Korea Atomic Energy Research Institute, from 1993 to 1999, and the Electronics and Telecommunications Research Institute, from 2004 to 2005. He was a Professor with the School of Computer Science and Engineering, Kyungpook National University, from 2005 to 2019. He is currently a Professor with the School of Electronics Engineering, Kyungpook National University. His research interests are 3-D scanning, 3-D registration, and robot vision.

SUNG-IL CHIEN received the B.S. degree from Seoul National University, Seoul, South Korea, in 1977, the M.S. degree from the Korea Advanced Institute of Science and Technology, Daejeon, South Korea, in 1981, and the Ph.D. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, USA, in 1988. Since 1981, he has been with the School of Electrical and Computer Engineering, Kyungpook National University, Daegu, South Korea, where he is currently a Professor. His research interests are computer vision, pattern recognition, and color image processing.