Reconstruction of 3D Models Consisting of Line Segments

Naoto Ienaga(✉) and Hideo Saito

Department of Information and Computer Science, Keio University, Yokohama, Japan
{ienaga,saito}@hvrl.ics.keio.ac.jp

Abstract. Reconstruction of three-dimensional (3D) models from images is currently one of the most important areas of research in computer vision. In this paper, we propose a method to recover 3D models using the minimum number of line segments. By using structure-from-motion, the proposed method first recovers a 3D model of line segments detected from an input image sequence. We then detect overlapping 3D line segments that redundantly represent a single line structure so that the number of 3D line segments representing the target scene can be reduced without losing the detailed geometry of the structure. We apply matching and depth information to remove redundant line segments from the model while keeping the necessary segments. In experiments, we confirm that the proposed method can greatly reduce the number of line segments. We also demonstrate that the accuracy and computational time for camera pose estimation can be significantly improved with the 3D line segment model recovered by the proposed method. Moreover, we have applied the proposed method to see through occluded areas.

1 Introduction

Reconstruction of three-dimensional (3D) models is one of the most studied topics in computer vision. For 3D model reconstruction from images, image features that provide the input data of structure-from-motion need to be matched between different viewpoints [1,2]. To achieve the matching of feature points, techniques, such as scale-invariant feature transform [3] or speeded up robust features [4], are often used. However, only a few feature points are detected in artificial situations consisting mainly of texture-less objects. For adapting such artificial environments, line segments can be used as image features.

For 3D reconstruction, line segments should also be matched between images from different viewpoints. In comparison with feature points, a line segment feature has a less distinctive appearance, which makes it difficult to correctly match line segments. Many existing methods [5–7], therefore, do not use appearance-based matching of line segments. However, some studies about descriptors for line segment features have been reported. The mean standard-deviation line descriptor [8] is a representative example and a line-based eight-directional histogram feature (LEHF) has also been introduced [9]. To make the LEHF more robust, directed LEHF has been proposed [10] and 3D reconstruction has been done with

C.-S. Chen et al. (Eds.): ACCV 2016 Workshops, Part II, LNCS 10117, pp. 100–113, 2017.
DOI: 10.1007/978-3-319-54427-4_8

this method [11]. Reconstructed 3D models are used for various purposes, including estimating camera pose for augmented reality [12]. In [12], the accuracy of camera poses has been improved by optimization through image sequences and plane segmentation. The reconstructed 3D models have a problem.

The method used in [12] is problematic because it retains all detected line segments of all frames. The final 3D model, therefore, has many redundant line segments. As shown in Fig. 2(a), many redundant line segments overlap at each position, although they should be represented by only one line segment. This redundancy leads to longer computation times and lower accuracy when we use the model to estimate camera poses or leads to worse visual information on the 3D model.

Line segment-based 3D models are superior to point cloud-based models in terms of visual information because a line segment indicates outlines. Some studies [13,14] have used line segments to enhance models, while other studies have suggested that using all of the available depth information (e.g., [15]) is better than using only line segments as visual information. In terms of data size, a line segment is superior [16]. We therefore conclude that line segment-based models are very useful, especially for improving the visual appearance of a model.

In this paper, we propose a method to reconstruct 3D models represented by line segments that can be adapted to texture-less situations. In this method, we first recover a 3D model based on the method in [12]. We then remove redundant line segments that overlap the same 3D line segments. Our contributions are as follow:

- We remove many redundant line segments of a 3D model that was reconstructed using the method in [12] without losing the detailed geometry of the structure.
- We experimentally demonstrate that the 3D model reconstructed by the proposed method provides improved accuracy of camera poses and requires less computation time.
- We improve the visual appearance of the model by removing line segments that do not actually exist.

Numerous studies investigating line segments detect them with a line segment detector (LSD) [17], which was also used in [12]. Line segments can also be considered as edges of a plane. In some studies (e.g., [18]), planes rather than line segments were used to create a 3D model. When we consider line segments to be edges *of* a plane, we cannot detect the line segments *on* a plane, such as textures, small depressions, and edges of thin objects like paper. For example, in Fig. 1(c), papers are attached on the whiteboard, and we can detect their edges, as shown in Fig. 1(f). They cannot be detected if we consider line segments to be edges of a plane.

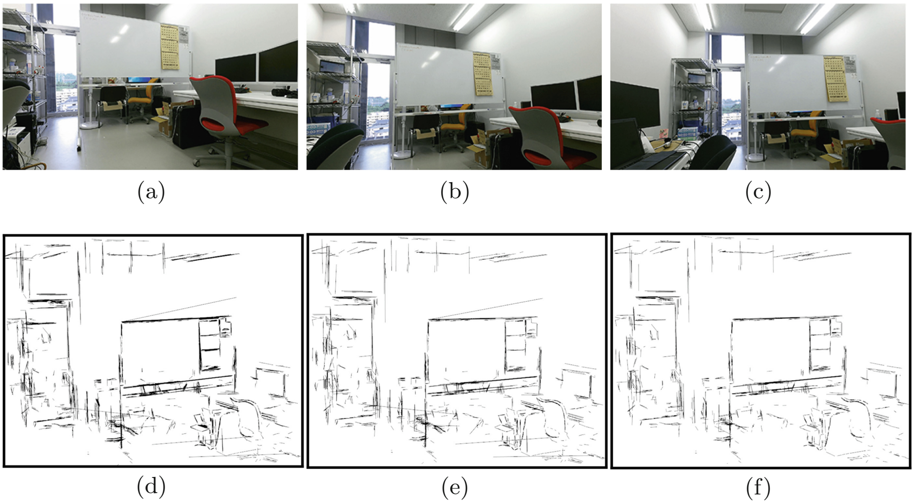


Fig. 1. Input images and 3D models of a whiteboard scene. (a) Input image 1 of 5. (b) Input image 3. (c) Input image 5. (d) 3D model reconstructed by [12]. (e) Reconstructed model with matching results. (f) Our final reconstructed model with matching results and depth information. Model (d) has 2266 line segments; model (f) has 1266 line segments.

2 Removing Redundant Line Segments

Our proposed method has two ways of removing redundant line segments from a 3D model; one utilizes matching information and the other utilizes depth information. Figure 1 shows three of five input images and the reconstructed models. Figure 1(d) is reconstructed by [12], and Fig. 1(f) is our final output.

2.1 Removing Redundant Line Segments with a Matching Result

When we consider the removal of redundant line segments, the information obtained by matching the line segments is very useful. Zhang and Faugeras [19] proposed a method to merge matched line segments to make a new line segment. In our case, the longest line segment of each matching line segment group is the most appropriate to keep because the method we are using [12] carefully removes outliers many times; this results in good line segment matching. We therefore remove redundant line segments by keeping only the longest ones.

Figure 2(b) is part of the model shown in Fig. 1(e). We removed all line segments except for the longest one of each group. As can be seen, we successfully removed redundant line segments while keeping the outlines.

2.2 Removing Redundant Line Segments with Depth Information

The model reconstructed by the previous process is not yet satisfactory as some line segments do not actually exist. Figure 3(a) shows such line segments.

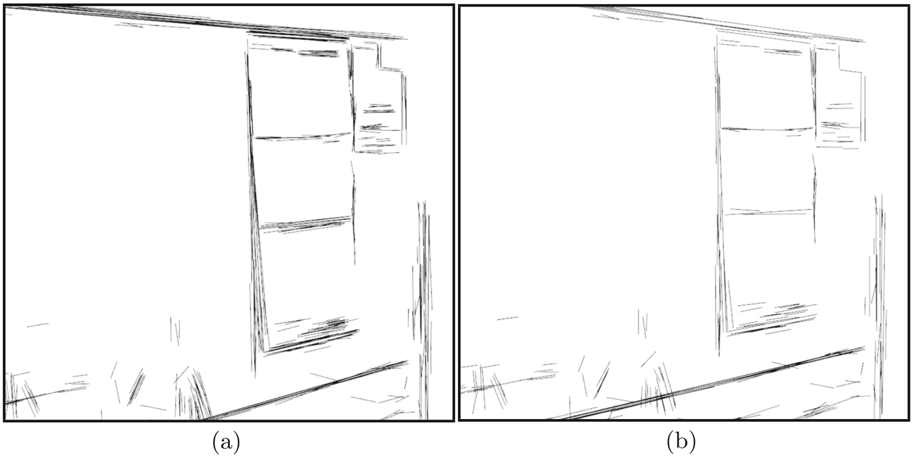


Fig. 2. Result of removing redundant line segments by matching. (a) Part of Fig. 1(d). Many redundant line segments are at each actual line structure position. (b) Part of Fig. 1(e). Compared with (a), many redundant line segments have clearly been removed.

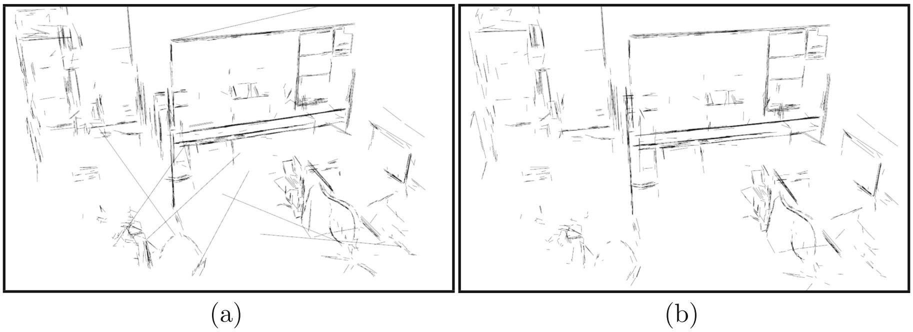


Fig. 3. Result of removing redundant line segments using the depth information. (a) Part of Fig. 1(e). Some line segments that are unnaturally projected do not actually exist. (b) Part of Fig. 1(f). Some line segments that do not actually exist in Fig. 3(a) are completely removed.

We utilize LSD [17] to detect the line segments from 2D images and use the method in [12] to make lattice points of line segments in 3D space from the start and end points detected by LSD in 2D space. We then remove redundant line segments. This method is displayed in Algorithm 1.

Figure 4 illustrates why nonexisting line segments are drawn. The depth values on a line segment detected on the boundary of an object in a 2D image are sometimes mixed depth values of the foreground and the background because of depth noise. Some line segments in Fig. 3(a) are detected on the boundary of objects. To remove these line segments, we check the depth values of the lattice points of the line segment and check whether the depth values change rapidly or smoothly. The proposed algorithm for removing redundant line segments related to depth is Algorithm 2. When the depth value between two lattice points exceeds the average change of depth values times 90% of the number of lattice points, a line segment is removed.

```
Algorithm 1. Making 3D Lattice Points of 2D Line Segment
   var
     SP: start point of a 2D line segment (input)
     EP: end point of a 2D line segment (input)
     P3: lattice points of a 3D line segment (output)
     STEP, dY, dX
   begin
     dY := EP.Y - SP.Y
     dX := EP.X - SP.X
     if | dY / dX | < 1
       STEP := dX / (dY^2 + dX^2)^0.5
       while (STEP > 0 and SP.X < EP.X) or
                                   (STEP < 0 and SP.X > EP.X)
         convert SP to a 3D point and add it to P3
         SP.X := SP.X + STEP
         SP.Y := Y value of a 2D line segment when X == SP.X
       endwhile
     else
       STEP := dY / (dY^2 + dX^2)^0.5
       while (STEP > 0 and SP.Y < EP.Y) or
                                   (STEP < 0 and SP.Y > EP.Y)
         convert SP to a 3D point and add it to P3
         SP.Y := SP.Y + STEP
         SP.X := X value of a 2D line segment when Y == SP.Y
       endwhile
     endif
end.
```

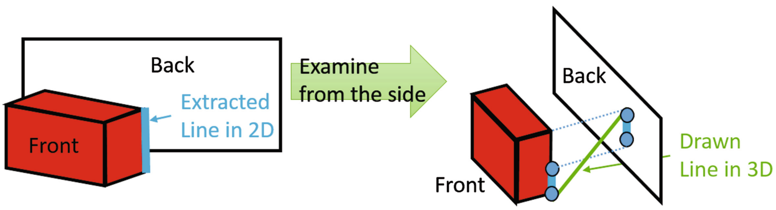


Fig. 4. Sometimes the depth values of the front of an object and its background on a line segment extracted from the boundary of an object in 2D are mixed. We draw a line segment in 3D space using the depth of both end points of the line segment extracted in 2D space. If one of the end points is located in front of the object and the other is located in the background, the line segment drawn in 3D becomes an incorrect line segment. To remove such a line segment, we check the depth values of the lattice points of each line segment.

saito@hvrl.ics.keio.ac.jp

```
Algorithm 2. Removing Redundant Line Segments with Depth
   output is TRUE or FALSE. FALSE means removing a line segment.
   var
     Z: Z values of lattice points of a 3D line segment (input)
     N: length of ARR
     MAX, MARGIN, ITER
   begin
     MAX := floor( N * 0.9 + 0.5 ) * | Z[0] - Z[N-1] | / N
     for ITER := 1 to N-1
       MARGIN := | Z[ITER-1] - Z[ITER] |
       if MARGIN > MAX
         return FALSE
       endif
     endfor
     return TRUE
end.
```

Figure 3(b) shows the result of Algorithm 2. Some line segments that do not actually exist are completely removed.

2.3 Summary of Algorithm for Removing Redundant Line Segments

We have explained the method for removing redundant line segments from a 3D model in two ways: first, with a matching result and second, with depth information. In practice, we first remove line segments with depth and then we keep the longest line segments of each matching group. The result of [12] is shown in Fig. 1(d) and that of our proposed method is shown in Figs. 1(f) and 5. The model in (d) [12] has 2266 line segments. In contrast, the model in (f) of the proposed method has 1266 line segments and successfully retains the outlines of the model. Moreover, we improve the visual appearance of the model by removing line segments that do not actually exist.

3 Experiments

We conducted experiments to demonstrate that we can remove only redundant line segments while keeping the necessary segments. In the experiments, we estimated camera poses by using both the 3D models reconstructed by [12] and those reconstructed by the proposed method.

We used input images of two scenes, shown in Fig. 6. The desk scene and the door scene consist of 12 and 60 frames, respectively. Table 1 shows the numberof line segments of the 3D models reconstructed by [12] and by the proposed method. The proposed method greatly reduced the number of segments used.

Table 2 shows that the proposed method leaves only the necessary line segments. We used 11 images to estimate the camera pose. The desk scene consisted of five images, and the door scene consisted of six images. In the desk

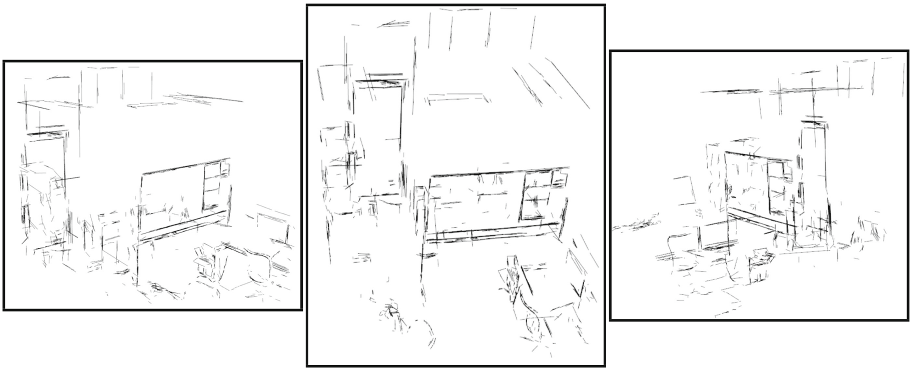
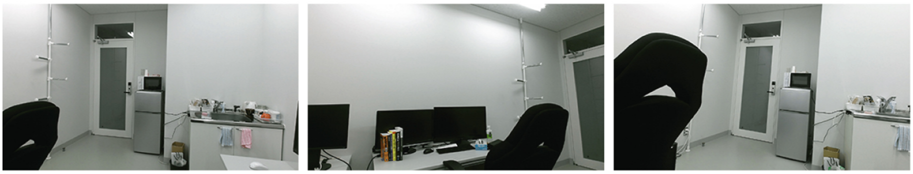


Fig. 5. Result of 3D reconstruction of the whiteboard scene. Three views for the model shown in Fig. 1(f) are presented.

frame 1 frame 6 frame 12

(a) Desk scene



frame 1 frame 31 frame 60

(b) Door scene

Fig. 6. Input images of the desk scene (a) and the door scene (b). The desk scene and the door scene consist of 12 and 60 frames, respectively.

Table 1. Number of line segments by the method in [12] and the proposed methods.

Scene	[12]	Proposed
Desk	3083	**1301**
Door	14339	**4775**

scene (Fig. 7), we tracked the purple book on the right side of the desk, and in the door scene (Fig. 8), we tracked the inside frame of the door on the left side. Each blue rectangle in both figures is the result of reprojecting four 3D points

Table 2. Average of reprojection errors and the computational times to estimate camera poses. The bold numbers indicate the lowest number of errors (pixels) and lowest times (s).

Image	Comparison	[12]	Proposed
Desk, image 1	Error	**4.70**	4.72
	Time	4.47	**3.93**
Desk, image 2	Error	1300.71	**294.01**
	Time	4.97	**4.44**
Desk, image 3	Error	37.13	**8.87**
	Time	4.74	**3.94**
Desk, image 4	Error	76.97	**56.46**
	Time	5.42	**4.68**
Desk, image 5	Error	**2.79**	2.96
	Time	4.48	**4.03**
Door, image 1	Error	49.51	**39.89**
	Time	6.47	**4.42**
Door, image 2	Error	**32.17**	35.31
	Time	9.04	**5.69**
Door, image 3	Error	38.01	**36.88**
	Time	6.64	**4.73**
Door, image 4	Error	**26.09**	37.21
	Time	6.28	**4.37**
Door, image 5	Error	32.57	**24.10**
	Time	7.97	**5.22**
Door, image 6	Error	**17.78**	24.26
	Time	6.74	**4.50**

for tracking, which were calculated by using the estimated camera poses. The average reprojection errors of the four points (in pixels) and the computational times (in seconds) to estimate the camera poses are shown in Table 2. The computational times might seem long compared with the ones reported in [12], but it should be noted that the experimental environments, such as the machine and the scenes, are different. We can clearly say that our models provided better accuracy of camera poses than those of the method in [12] in less computational time. When the camera positions of reconstructing a model and of tracking differ significantly, the reprojection error is large. Models reconstructed by the proposed method removed the incorrect line segments with depth, and the reprojection errors were therefore constrained considerably, as shown in images 2, 3, and 4 in the desk scene in Fig. 7. Although the proposed method sometimes has a large error, the difference is not so large compared with the difference when the proposed method has a smaller error.

image 1 image 2 image 3

image 4 image 5

(a) The method in [12]

image 1 image 2 image 3

image 4 image 5

(b) The proposed method

Fig. 7. Tracking results of the desk scene. (a) The method in [12]. (b) The proposed method. The reprojection errors and computational times are listed in Table 2. (Color figure online)

4 Application to Diminished Reality

As an example application of camera pose estimation using a reconstructed 3D model by the proposed method, we implemented a system for visualizing an area occluded by some obstacles, which is called diminished reality (DR). In this section, we show some results of the DR system based on the proposed method.

image 1 image 2 image 3

image 4 image 5 image 6

(a) The method in [12]

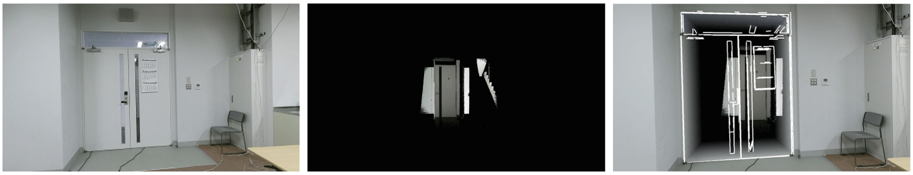
image 1 image 2 image 3

image 4 image 5 image 6

(b) The proposed method

Fig. 8. Tracking results of the door scene. (a) The method in [12]. (b) The proposed method. The reprojection errors and computational times are listed in Table 2. (Color figure online)

Using this DR system, we visualized a scene behind a wall of the room. Here, we reconstructed a line segment-based 3D model of the room by using the proposed method. The occluded area by the wall of the room was captured by an RGB-D camera, Kinect V2 and reconstructed using the method in [20]. In the performance of the DR stage, camera pose was estimated by the proposed method for each frame. The predefined area was tracked, and the occluded area



(a) (b) (c)

Fig. 9. (a) An example of input images. (b) Background image captured by Kinect V2 and reconstructed by [20]. The viewpoint of Kinect V2 is converted to that of a foreground camera, which has a viewpoint that is estimated by the proposed method. (c) DR result.

was superimposed on it. However, simple superimposition was poor at expressing the occluded area appearing on top of the wall rather than behind it. We therefore used the two approaches proposed in [21] to provide depth cues when viewing the occluded area. One approach used a gradation along the border between the wall and the occluded area, and the other superimposed line segments to the predefined area. The DR result is shown in Fig. 9(c).

Figure 10 shows a person walking outside a room. To capture a wide background, two Kinect V2s were used. We reconstructed the background in each frame in order to handle dynamic scenes. Because we can estimate the camera pose even in a scene with white walls, we could achieve DR by converting the viewpoint of the camera from behind the wall to the foreground.

5 Conclusion

We proposed a method to reconstruct a 3D model. We first recovered the 3D model using the method in [12] and removed the redundant line segments. In the method proposed in this paper, we used matching information to keep the minimum number of necessary line segments, and we removed line segments that were unnaturally projected due to depth noise. By removing such line segments, we improved the visual appearance.

In the experiments, we demonstrated that the proposed method could greatly reduce the number of line segments while keeping the detailed geometry of the structure. By using the proposed method to reconstruct a 3D model, the accuracy of estimated camera poses was better than that with the method in [12], and there were fewer line segments. By reducing the number of line segments, we can estimate camera poses with less computational time compared to [12]. Moreover, our proposed method enables seeing through walls.

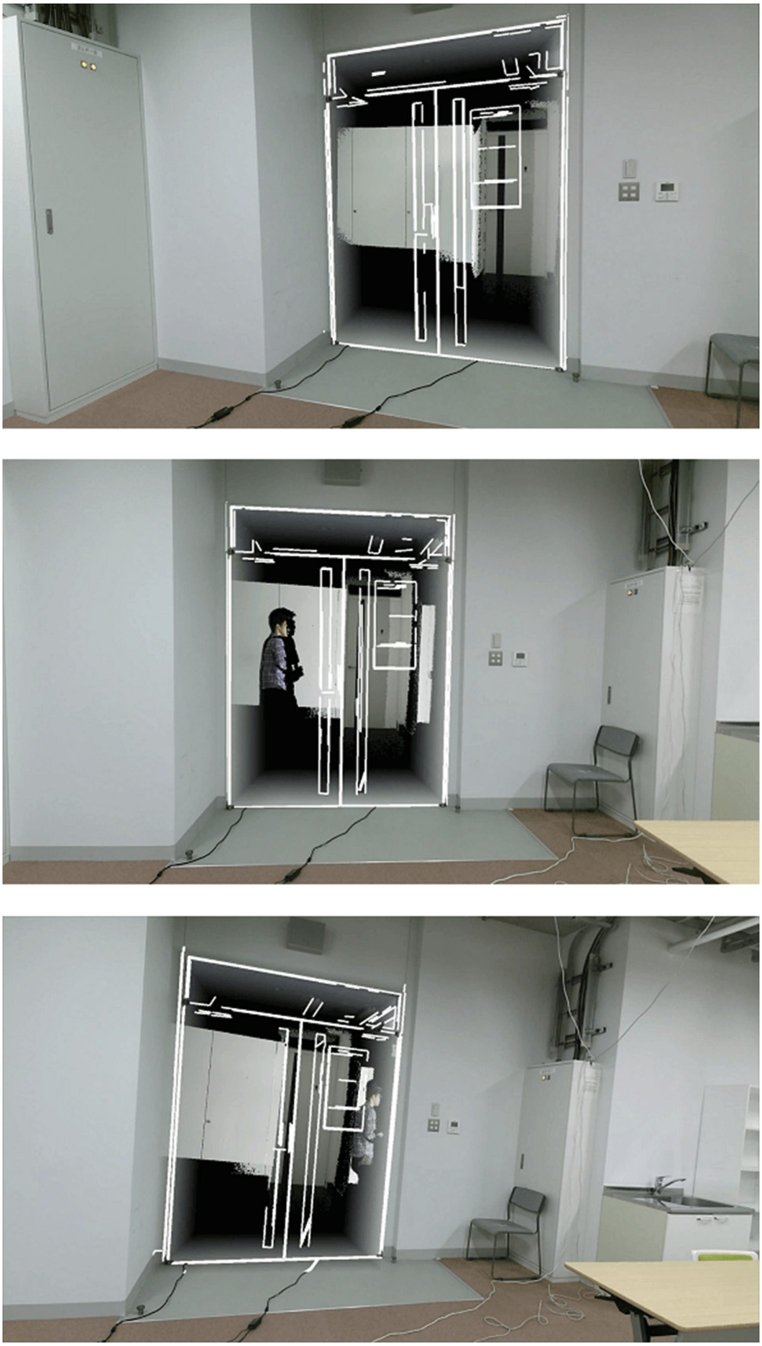


Fig. 10. DR results with a walking human.

Acknowledgement. This work was partially supported by MEXT/JSPS Grant-in-Aid for Scientific Research(S) 24220004, and JST CREST "Intelligent Information Processing Systems Creating Co-Experience Knowledge and Wisdom with Human-Machine Harmonious Collaboration".

References

1. Agarwal, S., Snavely, N., Simon, I., Seitz, S.M., Szeliski, R.: Building Rome in a day. In: 2009 IEEE 12th International Conference on Computer Vision, pp. 72–79 (2009)
2. Sturm, P., Triggs, B.: A factorization based algorithm for multi-image projective structure and motion. In: Buxton, B., Cipolla, R. (eds.) ECCV 1996. LNCS, vol. 1065, pp. 709–720. Springer, Heidelberg (1996). doi:10.1007/3-540-61123-1_183
3. Lowe, D.G.: Object recognition from local scale-invariant features. In: 7th IEEE International Conference on Computer Vision, pp. 1150–1157 (1999)
4. Bay, H., Tuytelaars, T., Gool, L.: SURF: speeded up robust features. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3951, pp. 404–417. Springer, Heidelberg (2006). doi:10.1007/11744023_32
5. Hofer, M., Wendel, A., Bischof, H.: Line-based 3D reconstruction of wiry objects. In: 18th Computer Vision Winter Workshop (2013)
6. Hofer, M., Wendel, A., Bischof, H.: Incremental line-based 3D reconstruction using geometric constraints. In: 24th British Machine Vision Conference, pp. 92.1–92.11 (2013)
7. Jain, A., Kurz, C., Thormählen, T., Seidel, H.P.: Exploiting global connectivity constraints for reconstruction of 3D line segments from images. In: 2010 IEEE Computer Vision and Pattern Recognition, pp. 1586–1593 (2010)
8. Wang, Z., Wu, F., Hu, Z.: MSLD: a robust descriptor for line matching. Pattern Recogn. **42**, 941–953 (2009)
9. Hirose, K., Saito, H.: Fast line description for line-based slam. In: 23th British Machine Vision Conference, pp. 83.1–83.11 (2012)
10. Nakayama, Y., Honda, T., Saito, H., Shimizu, M., Yamaguchi, N.: Accurate camera pose estimation for KinectFusion based on line segment matching by LEHF. In: 22nd International Conference on Pattern Recognition. 2149–2154. (2014)
11. Nakayama, Y., Saito, H., Shimizu, M., Yamaguchi, N.: 3D line segment based model generation by RGB-D camera for camera pose estimation. In: Jawahar, C.V., Shan, S. (eds.) ACCV 2014. LNCS, vol. 9010, pp. 459–472. Springer, Heidelberg (2015). doi:10.1007/978-3-319-16634-6_34
12. Nakayama, Y., Saito, H., Shimizu, M., Yamaguchi, N.: Marker-less augmented reality framework using on-site 3D line-segment-based model generation. J. Imaging Sci. Technol. **60**, 20401-1–20401-24 (2016)
13. Hofer, M., Maurer, M., Bischof, H.: Improving sparse 3D models for man-made environments using line-based 3D reconstruction. In: 2014 2nd International Conference on 3D Vision, pp. 535–542 (2014)
14. Sugiura, T., Torii, A., Okutomi, M.: 3D surface reconstruction from point-and-line cloud. In: 3rd International Conference on 3D Vision, pp. 264–272 (2015)
15. Newcombe, R.A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A.J., Kohi, P., Shotton, J., Hodges, S., Fitzgibbon, A.: KinectFusion: real-time dense surface mapping and tracking. In: 2011 10th IEEE International Symposium on Mixed and Augmented Reality, pp. 127–136 (2011)

16. Yonezawa, T., Ueda, K.: Real-time 3D data reduction and reproduction of spatial model using line detection in RGB image. In: Joint 7th International Conference on Soft Computing and Intelligent Systems and 15th International Symposium on Advanced Intelligent Systems, pp. 727–730 (2014)
17. von Gioi, R.G., Jakubowicz, J., Morel, J.M., Randall, G.: LSD: a line segment detector. Image Process. On Line **2**, 35–55 (2012)
18. Nguyen, T., Reitmayr, G., Schmalstieg, D.: Structural modeling from depth images. IEEE Trans. Vis. Comput. Graph. **21**, 1230–1240 (2015)
19. Zhang, Z., Faugeras, O.: Building a 3D world model with a mobile robot: 3D line segment representation and integration. In: 10th International Conference on Pattern Recognition, pp. 38–42 (1990)
20. Holz, D., Behnke, S.: Fast range image segmentation and smoothing using approximate surface reconstruction and region growing. In: Lee, S., Cho, H., Yoon, K.-J., Lee, J. (eds.) Intelligent Autonomous Systems 12, vol. 194, pp. 61–73. Springer, Heidelberg (2013)
21. Avery, B., Sandor, C., Thomas, B.H.: Improving spatial perception for augmented reality X-Ray vision. In: 2009 IEEE Virtual Reality Conference, pp. 79–82 (2009)