Creating See-through Image Using Two RGB-D Sensors for Remote Control Robot

Tatsuya Kittaka, Hiromitsu Fujii, Atsushi Yamashita and Hajime Asama
Department of Precision Engineering, Faculty of Engineering, The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan
Email: {kittaka, fujii, yamashita, asama}@robot.t.u-tokyo.ac.jp

***Abstract*—This paper presents a system which generates images to see through obstacles. For remote operation of robots in dangerous situations such as disaster sites, visual information affect the operational efficiency to a great extent. Diminished reality, which has been proposed in the field of computer vision, is a technique which generates images as if users saw through obstacles. The proposed method uses RGB-D sensors (RGB-D sensors can get not only RGB information as ordinary cameras can get, but also information of distance between the sensor and each point in the image) attached in front of and on the arm of the robot and enables to see through obstacles, letting the sensor move along with the arm. Using distance information from RGB-D sensors, coordinate transform is performed to produce images from arbitrary viewpoints. Since the method is simplified as much as possible, it is suitable for more general situation and requires smaller calculation cost than previous works. The experiments on a manipulator robot show the ability to see through obstacles from arbitrary viewpoints in real time.**

I. Introduction

Nowadays, there is increasing demand for remote control robots to conduct dangerous tasks in construction sites, disaster sites and so on. For instance, after a tsunami caused by the earthquake in the East Japan Pacific Ocean hit Fukushima Daiichi Nuclear Power Station in 2011, many kinds of robots have been used for unmanned remote operation to make reconnaissance, to clean up of rubble and so on [1].

However, it has been pointed out that operational efficiency of remote operation is significantly lower than that of manned operation. Providing operators with appropriate images plays an important role in improving the operational efficiency [2]. If target work objects are occluded by obstacles, operators have difficulty confirming the shape or the position of the objects, which leads to a drop of operational efficiency. Although multiple cameras are mounted on the robot to expand the field of view in conventional remote operation, operators have to conduct tasks while comparing multiple images, which requires highly skilled technique and the ability to concentrate [3]. Thus, it is effective to integrate multiple images into one image which enables operators to see the background (in this paper, we define "background" as an area which can not be seen in images because of obstacles).

In the field of computer vision, there is a technique called Diminished Reality (DR), which enables to eliminate real objects in images or see objects through obstacles [4]–[7]. This technique is also called See-through [8]. It is widely studied for entertainment [5] and services such as video completion [9] and pedestrians removal from Google Street View images [10]. We propose a system in which users can see the background through obstacles by applying DR.

II. Related Research

In DR, acquiring information of background is realized by various methods like using prior knowledge of the shape or the position of the background [11] or interpolating the background by using information of the surroundings (this technique is called Image Inpainting) [4]. However, it is not always possible to get accurate information of the background in unknown environment like disaster sites by these methods. In case of static background and moving obstacles, it is effective to use images in the past to acquire information of the background [9], although using multiple cameras and acquiring images from different viewpoints is suitable for more general use [5]–[8], [10]. In paticular, systems which allow to move cameras [5], [8], [11], produce less dead angle than those with fixed cameras.

Robots work near the background (the target objects), hence three-dimensional information of background is desired. However, most studies mentioned above approximate the background as a two-dimensional plane because they assume that the background is faraway enough from cameras. Actually, very few studies deal with three-dimensional information of the background [6], [7]. In paticular, Jarusirisawad et al. proposed a system in which the viewpoint of the output images can move (they placed a virtual camera in space to represent the viewpoint) [6]. This helps users to confirm the shape or the position of objects behind obstacles.

However, the system of Jarusirisawad et al. is not suitable for real-time remote operation of robots. They use an algorithm with high calculation cost called plane-sweep algorithm, which requires many iteration to acquire three-dimensional information. Sugimoto et al. used RGB-D sensors to acquire three-dimensional information much faster [7]. RGB-D sensors can get not only RGB information as ordinary cameras can get, but also information of distance between the sensor and each point in the image. Since the system of Sugimoto et al. does not allow any RGB-D sensors to move, using movable RGB-D sensors may be better approach to acquire three-dimensional information of background at a low calculation cost.

Hence, our approach is as follows: using multiple and movable RGB-D sensors, processing three-dimensional envi-

ronmental information, and defining an output view (a virtual sensor) in space to see the environment from arbitrary viewpoints.

III. PROPOSED METHOD

The outline of the proposed method is shown in Fig. 1. In this research, we suppose a robot with an arm, which is often used in disaster sites, and propose a system in which RGB-D sensors are attached in front of the robot and on the arm, as illustrated in Fig. 1 (a).

Figure 1 (b) shows the flow of the process. In order to reduce calculation cost for real-time see-through, the method is simplified as much as possible. First, images from the sensors attached in front (we call it "front sensor") and on the arm (we call it "arm sensor") are converted to three-dimensional point cloud data. Second, point clouds are coordinate-transformed to the coordinate system of "virtual sensor", which is set in arbitrary position in space. Then, point clouds are projected on a two-dimensional output image. Finally, images generated from the front sensor and the arm sensor are integrated by alpha blending to present see-through-obstacles images.

Detailed explanation of the components of this method, conversion into three-dimensional data, coordinate transformation, projection and integration is provided in following sections.

A. Conversion from Two-dimensional Images into Three-dimensional Data

By using internal parameters of the sensor and distance information from the RGB-D sensor, three-dimensional point cloud data are created.

Supposing that a certain point $\boldsymbol{p}$ is seen in an image at the position of ${}^{\mathrm{M}}\boldsymbol{p} = (u, v, 1)^{\mathrm{T}}$. Σ_{M} is a two-dimensional homogeneous coordinate system which unit is of pixels. The aim of this section is to determine the coordinate ${}^{\mathrm{S}}\boldsymbol{p} = (x, y, z, 1)^{\mathrm{T}}$, which is the coordinate $\boldsymbol{p}$ represented in a three-dimensional homogeneous coordinate system which unit is of millimeters. The origin of this coordinate system is the position of the sensor. By using the internal parameters of the sensor calibrated in advance and the distance information z from the RGB-D sensor, the relation between ${}^{\mathrm{M}}\boldsymbol{p}$ and ${}^{\mathrm{S}}\boldsymbol{p}$ can be described, and the unknown values x and y, components of ${}^{\mathrm{S}}\boldsymbol{p}$, are determined.

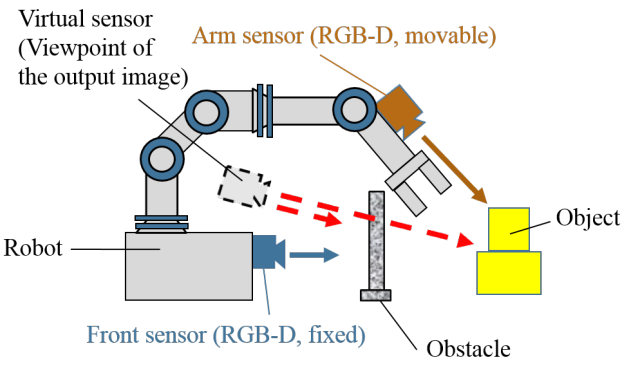
B. Coordinate Transformation to Imaginary Sensor Coordinate System

By multiplying rotation matrixes and translation vectors, the front-sensor coordinate system and the arm-sensor coordinate system are translated into the virtual-sensor coordinate system.

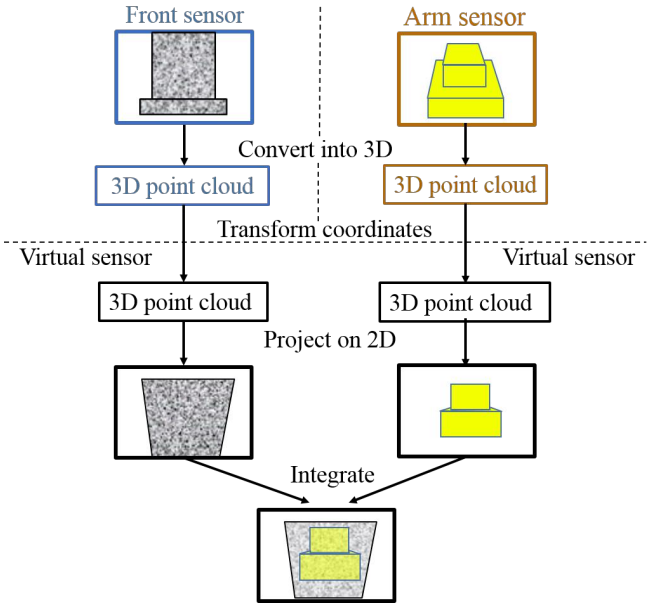
To transform coordinate system from Σ_{S} to $\Sigma_{\mathrm{S}'}$, a matrix ${}^{\mathrm{S}'}\mathbf{R}_{\mathrm{S}}$ and a vector ${}^{\mathrm{S}'}\mathbf{t}_{\mathrm{S}\to\mathrm{S}'}$ are needed as (1).

$$ {}^{\mathrm{S}'}\boldsymbol{p} = \begin{bmatrix} {}^{\mathrm{S}'}\mathbf{R}_{\mathrm{S}} & {}^{\mathrm{S}'}\mathbf{t}_{\mathrm{S}\to\mathrm{S}'} \\ \mathbf{0}^{\mathrm{T}} & 1 \end{bmatrix} {}^{\mathrm{S}}\boldsymbol{p} \tag{1} $$

${}^{\mathrm{S}'}\mathbf{R}_{\mathrm{S}}$ is a 3×3 rotation matrix which represents the rotation from Σ_{S} to $\Sigma_{\mathrm{S}'}$. ${}^{\mathrm{S}'}\mathbf{t}_{\mathrm{S}\to\mathrm{S}'}$ is a 3×1 translation vector which represents the position of the origin of Σ_{S} represented in the coordinate system $\Sigma_{\mathrm{S}'}$. We suppose that a position of any coordinate system Σ_{S} is represented by a rotation matrix ${}^{\mathrm{S}}\mathbf{R}_{\mathrm{W}}$ and a translation vector ${}^{\mathrm{S}}\mathbf{t}_{\mathrm{W}\to\mathrm{S}}$, where Σ_{W} is a world coordinate system fixed on the robot. Therefore, our main idea is to use ${}^{\mathrm{S}}\mathbf{R}_{\mathrm{W}}, {}^{\mathrm{S}}\mathbf{t}_{\mathrm{W}\to\mathrm{S}}$ and ${}^{\mathrm{S}'}\mathbf{R}_{\mathrm{W}}, {}^{\mathrm{S}'}\mathbf{t}_{\mathrm{W}\to\mathrm{S}'}$ to calculate ${}^{\mathrm{S}'}\mathbf{R}_{\mathrm{S}}$ and ${}^{\mathrm{S}'}\mathbf{t}_{\mathrm{S}\to\mathrm{S}'}$. In this method, since the destination coordinate system $\Sigma_{\mathrm{S}'}$ represents the virtual-sensor coordinate system, whose position can be set arbitrarily, ${}^{\mathrm{S}'}\mathbf{R}_{\mathrm{W}}$ and ${}^{\mathrm{S}'}\mathbf{t}_{\mathrm{W}\to\mathrm{S}'}$ can also be set arbitrarily. Accordingly, the aim of this section is to determine ${}^{\mathrm{S}}\mathbf{R}_{\mathrm{W}}$ and ${}^{\mathrm{S}}\mathbf{t}_{\mathrm{W}\to\mathrm{S}}$. There are two RGB-D sensors in our system, so ${}^{\mathrm{S}}\mathbf{R}_{\mathrm{W}}, {}^{\mathrm{S}}\mathbf{t}_{\mathrm{W}\to\mathrm{S}}$ for each sensor are needed.

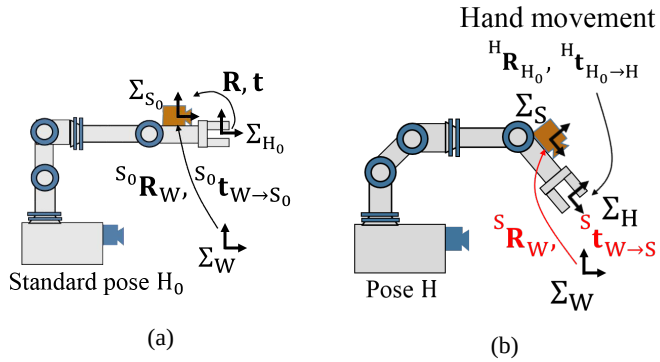


(a) The schema of the system. The RGB-D sensor on the arm moves along with the arm. The virtual sensor, which is set in arbitrary position in space, represents the position of the viewpoint of output images.



(b) The flow chart of the proposed method. Conversion from two-dimensional images into three-dimensional point cloud data, coordinate transformation, projection on the output image, and integration of two images are performed sequentially in real time.

Fig. 1: The concept of the proposed method



(a) (b)

Fig. 2: Updating the pose of the arm sensor. (a) The standard pose. The position of the arm sensor $^{S_0}\mathbf{R}_W, ^{S_0}\mathbf{t}_{W\to S_0}$ and the relation between the end-effector coordinate system and the arm-sensor coordinate system $\mathbf{R}, \mathbf{t}$ are calibrated in advance for coordinate transformation. (b) After a movement from the standard pose, the movement of the end effector $^{H}\mathbf{R}_{H_0}, ^{H}\mathbf{t}_{H_0\to H}$ are calculated to determine the new position of the arm sensor $^{S}\mathbf{R}_W, ^{S}\mathbf{t}_{W\to S}$.

If Σ_S represents the front-sensor coordinate system, $^{S}\mathbf{R}_W$ and $^{S}\mathbf{t}_{W\to S}$ are constant, because the front sensor is fixed on the robot. Therefore, transforming the front-sensor coordinate system into the virtual-sensor coordinate system is realized by calibrating $^{S}\mathbf{R}_W$ and $^{S}\mathbf{t}_{W\to S}$ in advance.

However, if Σ_S represents the arm-sensor coordinate system, $^{S}\mathbf{R}_W$ and $^{S}\mathbf{t}_{W\to S}$ should always be updated, because the arm sensor moves along with the arm. In the proposed method, we define a standard pose illustrated in Fig. 2 (a), the pose of the arm serving as a base, and calibrate $^{S_0}\mathbf{R}_W, ^{S_0}\mathbf{t}_{W\to S_0}$ at the standard pose in advance. Since the arm sensor is fixed on the arm, it is also possible to calculate constant matrixes $\mathbf{R}$ and $\mathbf{t}$ in advance, which are the rotation matrix and the translation vector from the end-effector coordinate system Σ_H to the arm-sensor coordinate system Σ_S. The end-effector coordinate system is represented as Σ_{H_0} when the arm is at the standard pose, and Σ_H when the arm has moved to a new position. A rotation matrix $^{H}\mathbf{R}_{H_0}$ and a translation vector $^{H}\mathbf{t}_{H_0\to H}$, which represent the movement of the end effector, can be calculated by solving a forward kinematics problem of the robot arm. Now, updating $^{S}\mathbf{R}_W, ^{S}\mathbf{t}_{W\to S}$ is realized using $^{S_0}\mathbf{R}_W, ^{S_0}\mathbf{t}_{W\to S_0}, ^{H}\mathbf{R}_{H_0}, ^{H}\mathbf{t}_{H_0\to H}, \mathbf{R}$ and $\mathbf{t}$.

C. Projection on the Output Image

This section explains how to transform point cloud data represented in the virtual-sensor coordinate system into a two-dimensional image seen from the position of the virtual sensor.

Supposing that the coordinate of a certain point $\boldsymbol{p}$ is $^{S'}\boldsymbol{p} = (x', y', z', 1)^{T}$. If $\boldsymbol{p}$ is projected on the output image at the position of $^{M'}\boldsymbol{p} = (u', v', 1)^{T}$, the relation between $^{M'}\boldsymbol{p}$ and $^{S'}\boldsymbol{p}$ is similar to that between $^{M}\boldsymbol{p}$ and $^{S}\boldsymbol{p}$ mentioned in Section III-A (in this case, internal parameters of the virtual sensor, which can be set arbitrarily, is needed).

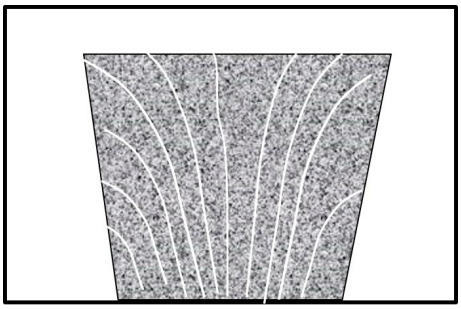


Fig. 3: A concept image of the missing pixels. The images generated by this process often lack pixels partially because point clouds exist discretely in space. For smoother images, interpolation is needed.

By using the process stated from Section III-A to this section, it is possible to calculate the position projected on the output image, $^{M'}\boldsymbol{p} = (u', v', 1)^{T}$, from the position achieved from the front sensor image and/or the arm sensor image, $^{M}\boldsymbol{p} = (u, v, 1)^{T}$. However, the images generated by this process often lack pixels partially, as illustrated in Fig. 3. The cause of this problem can be explained as follows. This process projects three-dimensional point clouds on two-dimensional images. Since the points exist discretely in space, the gap between points may appear in the image when seen from a different position. A typical solution to such problems is interpolation, which usually uses backward transformation to calculate the RGB information of the gap areas. However, as the proposed method does not use backward transformation but forward transformation, the dimension of the output information (two dimensional images) is smaller than that of the input information (three dimensional point clouds), hence it is impossible to determine a unique backward transformation. Therefore interpolation is performed by assigning the same RGB value to neighboring pixels in the output image for forward transformation. This simple interpolation enables to produce a certain level of smooth images at low calculation cost. The effect of this interpolation is evaluated quantitatively in Section IV-C.

D. Integration of the Output Images from the Front Sensor and the Arm Sensor

The output images from the front sensor and the arm sensor are made transparent and piled up to produce the final output image. The proposed method uses alpha blending to calculate the RGB values of each pixel. The opacity of the images from the front sensor and the arm sensor are α and $1-\alpha$ respectively, where $0 \leq \alpha \leq 1$. A pixel is filled in black if information about the pixel is obtained from neither the front sensor nor the arm sensor.

IV. EXPERIMENTS AND EVALUATION

A. Experimental Setup

We conducted experiments on an actual machine to verify the effectiveness of the proposed method. Figure 4 shows the

experimental setup. In this experiments, two RGB-D sensors (ASUS: Xtion Pro Live) and a manipulator robot (YASKAWA: MOTOMAN-HP3J) are used as illustrated in Fig. 4 (a). There are an obstacle in front of the front sensor and three balls with a diameter of 0.05 m which are the objects to be observed (Fig. 4 (b)). The distance between the front sensor and the obstacle is about 0.3 m and the distance between the front sensor and the farthest wall is about 1.0 m. The information of joint angles of the manipulator robot is acquired at a rate of 30 Hz. The RGB-D sensors acquire 640×480 pixel images at a rate of 30 fps. The movement of the end effector is set in advance. The coordinate of the end effector X, Y, Z is defined in Fig. 4 (a). As Fig. 5 (a) shows, the end effector moves about 0.15 m in the X, Y, and Z direction and returns to the initial position. Figure 5 (b) shows the time series data of the angle of the end effector. The angle is represented by Z-Y-X Euler angles, which means that any angle is represented by a first rotation about Z axis by an angle α, a second rotation about Y axis by an angle β, and a last rotation about X axis by an angle γ (if $\alpha = \beta = \gamma = 0$, the end effector is parallel to X axis). The internal parameters of the virtual sensor are the same as those of the front sensor. The parameter for alpha blending α is set to 0.5.

B. Resulting Output Images

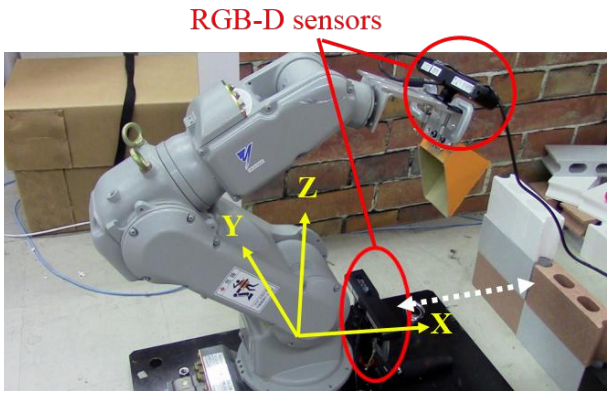
Images acquired by the experiments are shown in Fig. 6. The two images in Fig. 6 (a) are input images from the front sensor and the arm sensor at a certain moment. In the image from the front sensor, the balls behind the obstacle can not be seen. Figure 6 (b) shows the output image from the virtual sensor at this moment. The position of the virtual sensor is the same as that of the front sensor. Figure 6 (c) shows an image of the front sensor when there are no obstacles. Comparing Fig. 6 (b) and Fig. 6 (c), it turns out that the balls in the background can be seen properly.

Figure 7 shows output images after moving the position of the virtual sensor (The virtual sensor moved 0.15 m in the X direction and -0.15 m in the Y direction from the position of the front sensor, and rotated 30 deg about Z axis). Figure 7 (a) is an output image without interpolation which was mentioned in Section III-C. Figure 7 (b) is the output image after interpolation, which turns out to be smoother than that without interpolation. The position of the virtual sensor can be moved arbitrarily even while operating the manipulator to see the environment from various viewpoints.

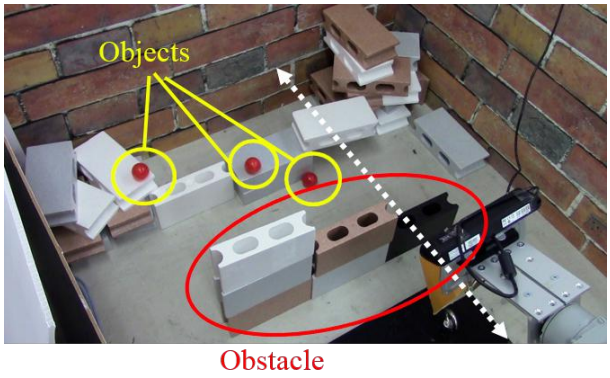
C. Evaluation and Discussion

We evaluated the error in the integrated image based on the disparity of the brick patterns printed on the farthest wall between the images from the front sensor and those from the arm sensor (illustrated in Fig. 7 (b) as an example). Throughout the movement in the experiments, it was observed that the disparity was smaller than 0.05 m at a distance of about 1.0 m from the front sensor.

The results of the other quantitative evaluations are as follows. Using a 3.40 GHz Intel Core i7 CPU, the frame rate of the output images was 9.2 fps on average, with a standard deviation of 0.6 fps. This processing speed was acceptable within the experiments; the system worked in real time.

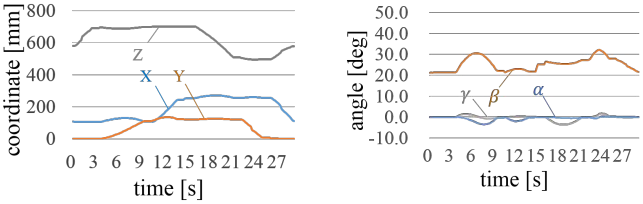


(a) The manipulator and the RGB-D sensors. The distance between the front sensor and the obstacle (dotted arrow) is about 0.3 m.



(b) An obstacle in front of the front sensor and three objects to be observed. The distance between the front sensor and the farthest wall (dotted arrow, although the front sensor is not in this picture) is about 1.0 m.

Fig. 4: The experimental environment



(a) (b)

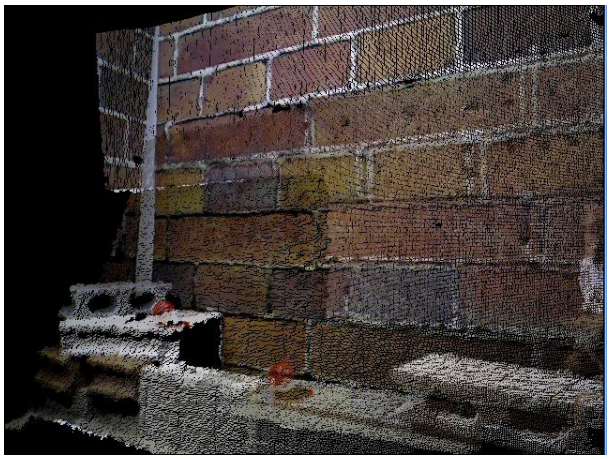
Fig. 5: The trajectory of the end effector. (a) The time series data of the coordinates X, Y, Z of the end effector during the experiments. The coordinate system is defined in Fig. 4 (a). (b) The time series data of the angles α, β, γ of the end effector during the experiments. The angle is represented by Z-Y-X Euler angles.

(a) Input images from the front sensor (left) and the arm sensor (right) at a certain moment. In the image from the front sensor, the balls behind the obstacle can not be seen.

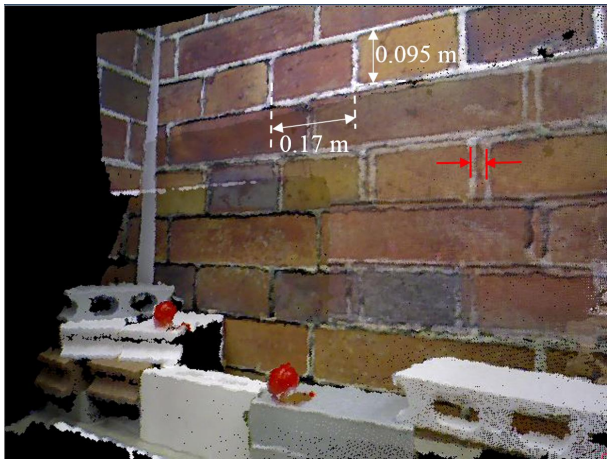
(b) The output image at this moment. The balls can be seen through the obstacle.

(c) The ground truth data. Comparing Fig. 6 (b) and this picture, it turns out that the balls in the background can be seen properly.

Fig. 6: The images acquired by the experiments



(a) An output image after moving the position of the virtual sensor without interpolation which was mentioned in Section III-C. The virtual sensor moved 0.15 m in the X direction and -0.15 m in the Y direction from the position of the front sensor, and rotated 30 deg about Z axis. In this picture, information of about 40 % of the field of view is missing.



(b) After the interpolation, the missing pixels improved by about 20 % making the output image smoother. This process can be performed at low calculation cost, which allows the whole system to work in real time. Throughout the movement in the experiments, it was observed that the disparity (illustrated in the picture as an example) was smaller than 0.05 m at a distance of about 1.0 m from the sensor.

Fig. 7: Acquired images after moving the viewpoint

Next, we evaluated how precisely the output images reproduced the true images. We focused on not disparity of positions but disparity of texture and used NCC (Normalized Cross-Correlation) as a measure. The evaluation was performed under a condition without the obstacle and the position of the virtual sensor was the same as that of the front sensor. The NCC between the images of the front sensor and those of the virtual sensor resulted in 0.94 on average, with a standard deviation of 9.1×10^{-3}, which means that there is high similarity between the output images and the real images.

We also evaluated the missing pixels which was mentioned in Section III-C. The percentage of the number of the missing

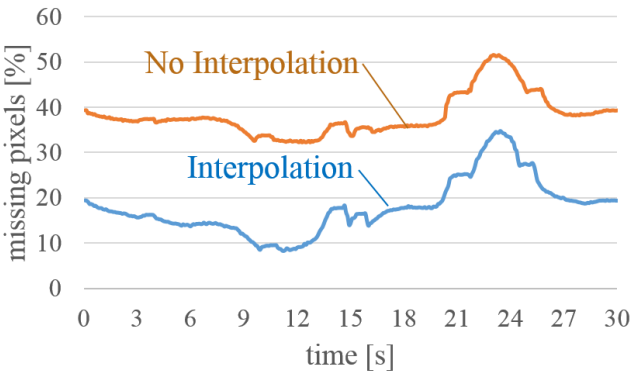


Fig. 8: The percentage of the missing pixels during the experiments. The evaluation was performed when the virtual sensor was located at the position in Fig. 7. The percentage improved after the simple interpolation explained in Section III-C.

pixels to the total number of pixels in the output images was calculated. The evaluation was performed when the virtual sensor was located at the position in Fig. 7. Then a comparison was made between the percentage before and after the interpolation explained in Section III-C. Figure 8 shows the result. The percentage of the missing pixels varied during the movement. Especially, there was a big missing pixels during the period of time between 20 s and 27 s. The cause of this rise in the percentage can be explained as follows. As Fig. 5 (a) shows, the X coordinate is big and the Z coordinate is small during this period, which means that the arm sensor is close to the environment and that the field of view is limited in narrow area. In order to remove the effect of such kind of outliers, we evaluated the percentage by calculating median. The median was 37 % before the interpolation, and 17 % after the interpolation. It turned out that information of about 83 % of the field of view can be acquired, in spite of not small distance between the virtual position of the viewpoint and the actual positions of the sensors. It also turned out that the interpolation improved the missing pixels by about 20 % making the output image smoother.

The whole experimental results suggest that our system helps users to acquire information of objects behind obstacles by moving the arm sensor and/or the virtual sensor while operating the manipulator.

V. Conclusion

We proposed a system in which operators of remote control robots can see the background through obstacles by applying DR. The proposed method uses movable RGB-D sensors, which enable to see the background according to positions of obstacles. Information from RGB-D sensors, coordinate transformation and other processes enable to produce images from arbitrary viewpoints. Since the method is simplified as much as possible, it is suitable for more general situation and requires smaller calculation cost than previous works. The experiments on the actual machine verified the ability to see the background in real time and to acquire information of the great part of the field of view, even if the viewpoint moved.

For future works, there are three issues:

1) **Reducing errors in image integration**
2) **Reducing the missing pixels when the viewpoint moves**
3) **Expanding the method for any numbers of sensors**

For expanding the method, it is essential to reduce the errors. Expanding the method may in turn contribute to reducing the missing pixels.

Acknowledgments

This work was funded by Tough Robotics Challenge, ImPACT Program of the Council for Science, Technology and Innovation (Cabinet Office, Government of Japan).

The manipulator robot was offered from YASKAWA Electric Corporation.

References

[1] Shinji Kawatsuma, Mineo Fukushima and Takashi Okada: “Emergency Response by Robots to Fukushima-Daiichi Accident: Summary and Lessons Learned,” Industrial Robot: An International Journal, Vol. 39, No. 5, pp. 428–435, 2012.

[2] Masaharu Moteki, Kenichi Fujino, Takashi Ohtsuki, and Tsuyoshi Hashimoto: “Research on Visual Point of Operator in Remote Control of Construction Machinery,” Proceedings of the 28th International Symposium on Automation and Robotics in Construction, pp. 532–537, 2010.

[3] Akihiko Nishiyama, Masaharu Moteki, Kenichi Fujino and Takeshi Hashimoto: “Research on the Comparison of Operator Viewpoints between Manned and Remote Control Operation in Unmanned Construction Systems,” Proceedings of the 30th International Symposium on Automation and Robotics in Construction, pp. 772–780, 2013.

[4] Jan Herling and Wolfgang Broll: “Pixmix: A Real-time Approach to High-quality Diminished Reality,” Proceedings of the IEEE International Symposium on Mixed and Augmented Reality 2012, pp. 141–150, 2012.

[5] Enomoto Akihito and Hideo Saito: “Diminished Reality Using Multiple Handheld Cameras,” Proceedings of the 8th Asian Conference on Computer Vision, Vol. 7, pp. 130–135, 2007.

[6] Songkran Jarusirisawad, Takahide Hosokawa and Hideo Saito: “Diminished Reality Using Plane-sweep Algorithm with Weakly-calibrated Cameras,” Progress in Informatics, No. 7, pp. 11–20, 2010.

[7] Kazuya Sugimoto, Hiromitsu Fujii, Atsushi Yamashita and Hajime Asama: “Half Diminished Reality Image Using Three RGB-D Sensors for Remote Control Robots,” Proceedings of the 12th IEEE International Symposium on Safety, Security, and Rescue Robotics, 43, pp. 1–6, 2014.

[8] Peter Barnum, Yaser Sheikh, Ankur Datta and Takeo Kanade: “Dynamic Seethroughs: Synthesizing Hidden Views of Moving Objects,” Proceedings of the IEEE International Symposium on Mixed and Augmented Reality 2009, pp. 111–114, 2009.

[9] Yuping Shen, Fei Lu, Xiaochun Cao and Hassan Foroosh: “Video Completion for Perspective Camera under Constrained Motion,” Proceedings of the 18th International Conference on Pattern Recognition, Vol. 3, pp. 63–66, 2006.

[10] Arturo Flores and Serge Belongie: “Removing Pedestrians from Google Street View Images,” Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops 2010, pp. 53–58, 2010.

[11] Francesco Cosco, Carlos Garre, Fabio Bruno, Maurizio Muzzupappa and Miguel A. Otaduy: “Augmented Touch without Visual Obtrusion,” Proceedings of the IEEE International Symposium on Mixed and Augmented Reality 2009, pp. 99–102, 2009.