A Recognition System of AR Markers Attached to Carts in a Factory

Takumi Kitsukawa[1], Masahiro Takahashi[1], Alessandro Moro[2], Yoshihiro Harada[3], Hideo Nishikawa[3], Minori Noguchi[3], Akifumi Hamaya[3], and Kazunori Umeda[1]

***Abstract*— In this paper, we propose a system for recognizing carts by attaching AR markers to them. In recent years, the Internet of Things (IoT) has been introduced into factories in the manufacturing industry. However, factories that produce a wide variety of products in small quantities still use carts for their operations, and automation has not progressed. Therefore, a method of attaching a low-cost AR marker to the cart and using a fixed-point camera to recognize the ID is considered. When using this method, it is necessary to improve the recognition performance of the marker by using image processing because the marker attached to the cart is small. In the proposed system, markers are detected using an object-detection method based on deep learning in images acquired by a fixed-point camera and recognized by a combination of cropping, preprocessing, and deblurring. As a result, the distance from which AR markers can be recognized increased from 2.9 m to 3.9 m. The recognition rate was improved from 12% to 81% with a distance of 1 m and a speed of 0.25 m/s. It has also been confirmed that the system can be processed online. We verified the practicality of the system by conducting an experiment using a cart in an actual factory.**

I. Introduction

In recent years, the IoT has been introduced into factories in the manufacturing industry. As a result, the automation of various operations is being promoted. However, factories that produce a wide variety of products in small quantities use carts like the one shown in Fig. 1 to take parts one by one from the shelves and transport them to the processing location for processing and assembly. It is difficult to automate these operations, and many of the processes are done manually. In other words, factories using carts are not able to visualize the production line, and there are two main issues that need to be solved.

The first issue is understanding the work process and progress of each worker. Without IoT, we are not able to understand the work content and work hours of workers in the factory. It is also difficult to know whether the number of workers for a task is appropriate. In addition, since individual work hours are not quantified, it is difficult to know if the work is being done correctly.

The second issue is identifying and locating carts and parts. Carts are scattered throughout the factory and warehouse, and it takes time and effort to find the desired carts. In many cases, the design of the carts is uniform throughout the factory, and since the parts mounted on the carts are similar, there is a possibility that the carts may be misplaced.

To solve these problems, it is thought that integrated circuit (IC) chips or other devices that emit radio waves can be attached to the carts to keep track of their location and those of workers [2]. However, this solution is costly and time consuming to implement. A less costly solution is to attach AR markers to the carts, which are then photographed and read by a fixed-point camera in the factory to determine the position of the carts. However, assuming that AR markers are actually attached to the carts, they must be small so that they do not interfere with the work. Since using a small marker would reduce the recognition performance, it is necessary to improve the performance of the markers for practical use.

Fig. 1 Proposed system illustration [1]

Conventional studies have been conducted to improve the recognition performance of AR markers by improving the design of existing markers or proposing new marker designs [3] [4]. These markers have shown improved recognition accuracy but have hardly increased the distance for recognition. Also, problems such as increased marker size and decreased number of marker IDs can be created as compared to conventional markers.

Therefore, in this paper, we aim to improve recognition performance by using existing AR markers and combining them with object detection and deblurring methods. For the existing AR marker, we use the ArUco marker [5]. An overview of the proposed system is shown in Fig. 1. In this paper, we describe the proposed system and verify its effectiveness through experiments.

II. Proposed System

A. Overview of the Proposed System

The flow of the proposed system is shown in Fig. 2. From the captured image, we detect a marker using a deep learning method and crop the area around the marker. We preprocess the cropped image and recognize the markers by using the ArUco library [6]. We found that by cropping the area around a detected marker, the rate of recognizing distant markers increases.

[1]The Course of Precision Engineering, School of Science and Engineering, Chuo University, 1-13-27 Kasuga, Bunkyo-ku, Tokyo, Japan (Corresponding author: kitsukawa@sensor.mech.chuo-u.ac.jp)

[2]RITECS Inc., 3-5-11 Shibasaki, Tachikawa-shi, Tokyo, Japan

[3]Hitachi High-Tech Solutions Corporation, 1-17-1 Toranomon, Minato-ku, Tokyo, Japan

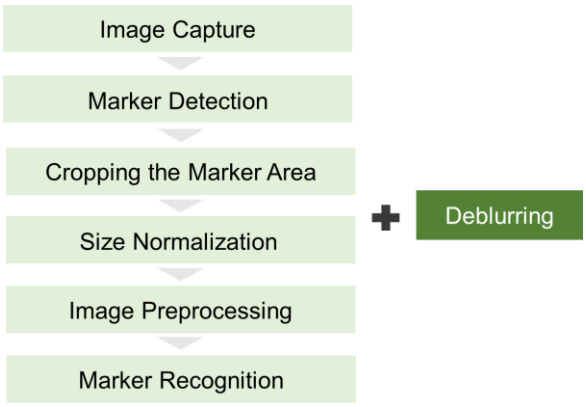

Fig. 2 Flow of the proposed system

In addition, attaching the AR marker to a cart in the factory means that the marker itself moves. As a result, there is a possibility that the acquired image will be blurred. Therefore, by adding a deblurring process using deep learning, we will construct a system that can deal with blurred images. Section III.C explains where to add the deblurring process.

The ArUco library, which is used for marker recognition, does some simple preprocessing such as binarization and distortion correction for angled markers. Therefore, to add cropping and deblurring processes to the system seems to be effective because they are not processed at the library. In this paper, the method of directly recognizing a marker from a captured image by the ArUco library is described as a previous method.

B. Marker Detection Using Deep Learning

For marker detection using deep learning, we use the object-detection method Faster R-CNN [7]. It has a model structure of identifying whether the content of a rectangle is an object or background and classifying the detected region. Unlike conventional object-detection methods, such as R-CNN [8] and Fast R-CNN [9], this method uses a CNN (Convolutional Neural Network) structure called RPN (Region Proposal Network) [10] to extract object candidate regions; consequently, the processing time is greatly improved. Marker detection is enabled by fine-tuning the trained model of this Faster R-CNN with a custom dataset created for markers.

To create a custom dataset for markers, we use Siam Mask [11], which is a mask-based object-tracking method that efficiently creates a large amount of training data. Given the position of an object to be tracked in the first frame of a video, it estimates the position of the object in all subsequent frames. It is faster than other deep learning–based object-tracking methods and can be operated in real time.

C. Preprocessing of Cropped Images

After detecting the marker and obtaining the coordinates of the four points as described in Section II.B, we crop the area around the marker from the input image. The next step is to normalize the size of the cropped image. In this way, it is possible to generate a cropped image of the same size regardless of the distance between the camera and the marker.

It is thought that the accuracy of marker recognition can be improved by sharpening the cropped image. Figure 3 (upper) shows the sharpening process applied to the cropped image. It can be seen that the contrast of the markers is clearer than in the original image, making it easier to recognize the shapes. However, since the noise in the original image is also sharpened, the boundary line between black and white is not clear in some areas. Therefore, the cropped image is first subjected to smoothing and then to sharpening, as shown in Fig. 3 (lower). As compared to the image with only the sharpening process, the noise is removed and the light and dark areas are clearer, making it easier to recognize the shape of the marker.

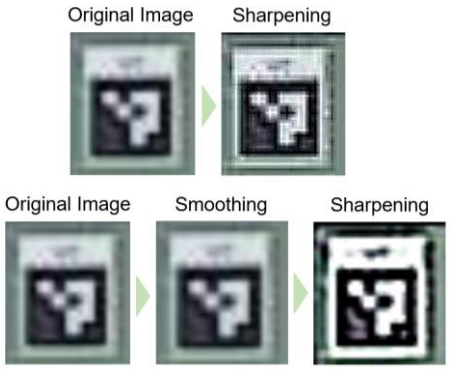

Fig. 3 Upper: sharpening only,
Lower: sharpening after smoothing

Based on these results, we use a combination of smoothing and sharpening to preprocess the cropped image.

D. Deblurring Using Deep Learning

For deblurring using deep learning, we use DeblurGAN-v2 [13], a method that adapts Generative Adversarial Networks (GAN) [12].

The model structure of DeblurGAN-v2 is as follows: in Generator, create an image by adding the upsampled high-dimensional feature map and the low-dimensional feature map, while in Discriminator, introduce PatchGAN to divide the generated image into patches to determine whether it is real or fake. In this way, the model focuses on an even small blur in the image. It is also capable of removing the blur from blurred images that are not in the training data.

III. EXPERIMENTS

In this section, we evaluate the proposed system. We used I-O Data Device's Qwatch network camera TS-WRLP [15] to acquire images. We used an image resolution of 1280×720. All AR markers were 3[cm]×3[cm] in size, and the IDs were displayed on the markers so that they could be seen at a glance. The detectors were created by fine-tuning a Faster R-CNN model that had been pre-trained for the COCO dataset [16]. To create them, we used Detectron2 [17], a deep-learning library that implements object-detection algorithms.

A. Experiment 1: Creation of Marker Detectors

When creating marker detectors as described in Section II.B, it is difficult to create detectors that can learn all of the ID shapes used in a factory. Therefore, in this experiment, we created multiple detectors with different types of markers to be learned. We then evaluated how many types of markers needed to be learned.

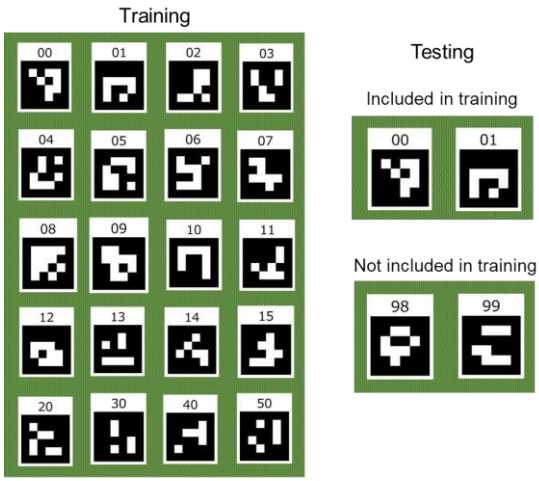

Fig. 4 Relationship between number of trained marker types and detection rate

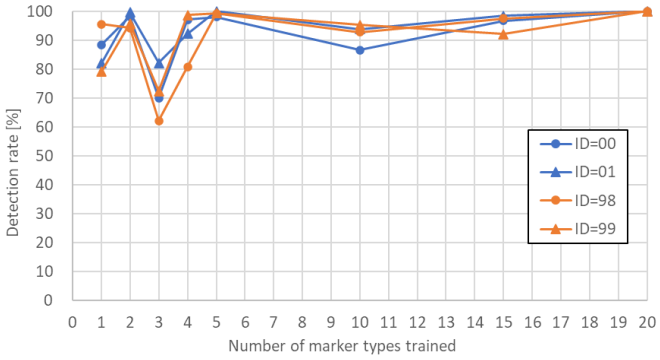

Fig. 5 Relationship between number of trained marker types and detection rate

Experimental conditions. Markers with 20 different IDs were used for training. For testing, we used four types of IDs: 00, 01, 98, and 99. 00 and 01 are included in the training data, while 98 and 99 are not. Figure 4 shows them. The number of marker types (IDs) to be trained was increased to 1, 2, 3... and detectors were created for each marker type. We used Siam Mask, as described in Section II.B, to create the training data. Videos of markers moving back and forth over a distance of approximately 0.3 m to 3 m were used for training and testing. They were captured in a room from the same camera position. The video used for training contained approximately 5200 images in 3 minutes, and the video used for testing contained about 900 images in 30 seconds. The hyperparameters for training the detectors were fixed at 0.005 for the learning rate, 300 for the number of epochs, and 64 for the batch size.

Figure 5 shows the detection rate according to the number of marker types learned. It can be seen that even if the number of marker types to be trained is small, it is sometimes possible to create detectors with high accuracy. Additionally, if the number of types is increased to 15 or 20, the detection rate increases, and the detection of unlearned markers is more stable.

Even though markers with ID=98, 99 were not included in the training data, the detection rate was as high as that of ID=00, 01. This is probably because the pre-trained model of Faster R-CNN can create highly accurate detectors even with a small amount of biased training data.

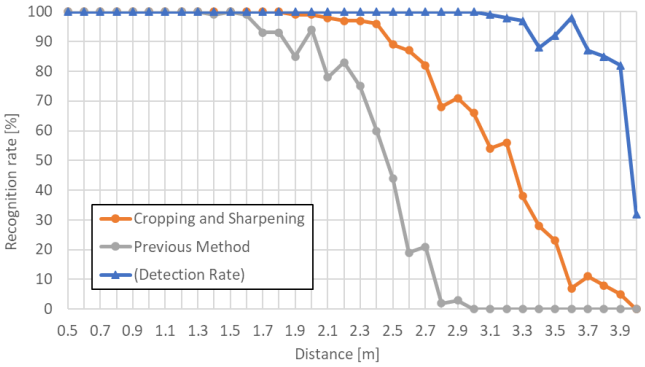

Fig. 6 Relationship between distance and recognition rate

B. *Experiment 2: Evaluation of Distance for Recognition*

In this experiment, we verify that the proposed method, which is shown on the left of Fig. 2, increases the distance at which markers can be recognized by preprocessing the cropped area after detecting a marker.

Experimental conditions. We used 10 kinds of marker IDs from 0 to 9. The distances were 0.5 m to 4.0 m from the camera. Ten images were taken at each distance, and the recognition rate was calculated. We used the marker detectors described in Section II.B, which were fine-tuned and created with a total of about 12000 custom datasets using the Siam Mask annotation tool. To preprocess the cropped images, we used a method of smoothing followed by sharpening, as described in Section II.C. We used a 3×3 Gaussian filter as described in Section II.C for smoothing and an 8-neighborhood sharpening filter for sharpening after cropping around the detected marker.

Figure 6 shows the experimental results. The success rate of marker recognition at each distance is shown in the figure: the recognition rate is over 80% up to 2.7 m, and the recognition rate is improved at each distance. In addition, the distance at which recognition became impossible increased from 3.0 m to 4.0 m.

From the above results, we can confirm that the distance for recognition of AR markers can be increased by combining marker detection using deep learning, neighborhood cropping, and preprocessing.

C. *Experiment 3: Evaluation of Deblurring*

In this experiment, we made the AR marker blur by sliding it horizontally at a certain speed. We verified that the proposed method with deblurring improves the recognition performance of the obtained blurred images. The purpose of the experiment was twofold: first, to evaluate the effectiveness of the deblurring process by applying deep learning to the blurred images, and second, to evaluate the recognition performance when the cropping process and the preprocessing (smoothing and sharpening) are combined with the deblurring process. There are two possible locations for the deblurring process: one for the entire input image and the other for the cropped image after the marker is detected. Based on the above, the flow of the methods to be compared in this experiment is

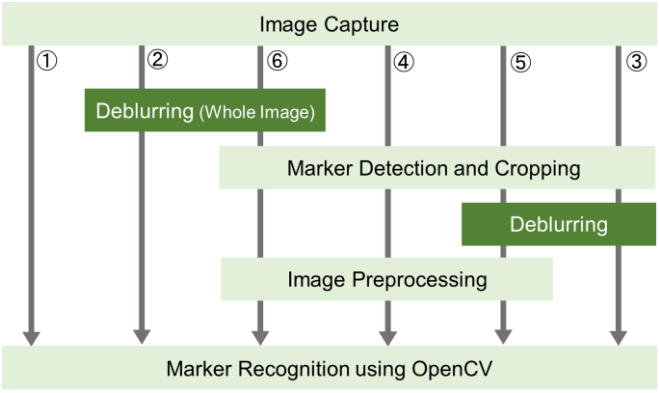

Fig. 7 Flow of methods to be compared

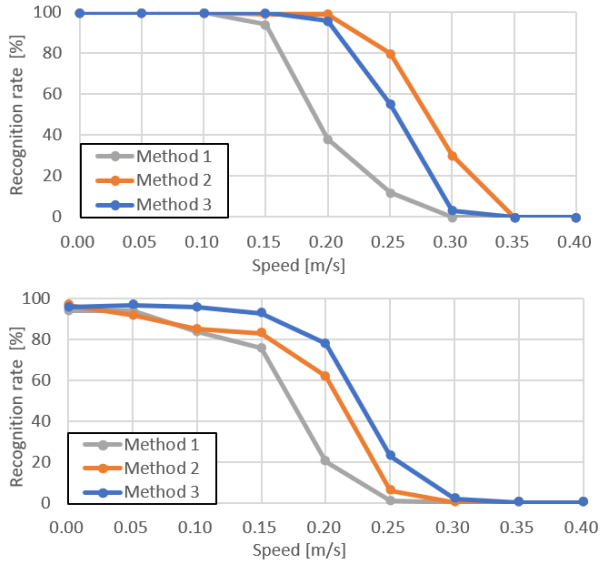

Fig. 8 Relationship between speed and recognition rate
Upper: distance 1 m, Lower: distance 2 m

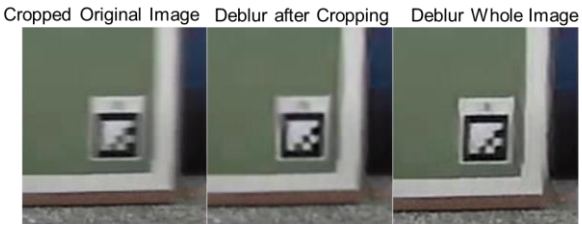

Fig. 9 Actual deblurred image

shown in Fig. 7. The numbers in the figure indicate the number of each method; we call them Methods 1–6.

Experimental conditions. The experiments were conducted at distances of 1 m and 2 m. The speed was increased by 0.05 m/s, 0.10 m/s, 0.15 m/s, and 0.20 m/s; 10 images were acquired at each speed to obtain the recognition success rate. The marker detectors and the preprocessing, smoothing, and sharpening were the same conditions as in the previous experiments. For the deblurring process, we used the trained model of DeblurGAN-v2, as described in Section II.D.

First, we compared Methods 1, 2, and 3 to evaluate the effectiveness of the deblurring process. Figure 8 shows the relationship between speed and the recognition success rates at distances of 1 m and 2 m. Both at 1 m and 2 m, the recognition rate is improved by using the deblurring process.

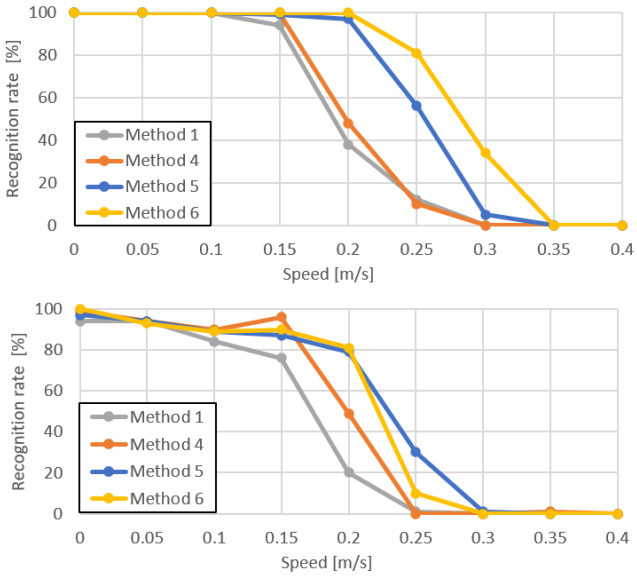

Fig. 10 Relationship between speed and recognition rate
Upper: distance 1 m, Lower: distance 2 m

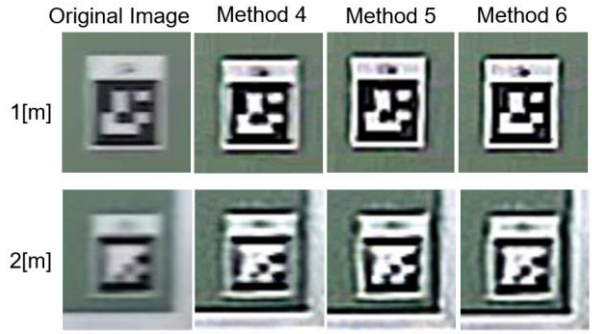

Fig. 11 Actual image for comparison for Methods 4-6
Upper: distance 1 m, Lower: distance 2 m

At a distance of 1 m, the recognition performance of Method 2, which performs deblurring on the entire input image, is higher than that of Method 3, which performs deblurring after cropping. On the other hand, at a distance of 2 m, the recognition performance of Method 3 was higher than that of Method 4. This may be due to the characteristics of DeblurGAN-v2, which is used in the deblurring process. Figure 9 shows the actual deblurred images. The two images on the left in the figure are the results of deblurring after cropping using Method 3. The rightmost image in the figure is the result of deblurring the whole image using Method 2; it is enlarged for comparison with the result of Method 3. These results confirm the effectiveness of the process for deblurring images.

Second, we compared Methods 4, 5, and 6 and evaluated their recognition performance when they were combined with the cropping, smoothing, and sharpening processes. Figure 10 shows the relationship between speed and the recognition rate at distances of 1 m and 2 m. As compared to Method 4, which performs smoothing and sharpening after cropping, Method 5, which adds deblurring after cropping, and Method 6, which adds deblurring to the entire image, improve the recognition

TABLE I. Processing speed of each method

	Method1	Method2	Method3	Method4	Method5	Method6
Processing Speed	100fps	4.6fps	4.0fps 2.9fps(one) 2.3fps(two)	4.0fps	4.0fps 2.9fps(one) 2.3fps(two)	2.2fps

performance. At a distance of 1 m, the recognition performance of Method 6 is higher than that of Method 5. On the other hand, at a distance of 2 m, the recognition performance of Method 5 is higher than that of Method 6. Figure 11 shows the actual images at distances of 1 m and 2 m and a speed of 0.25 m/s. From left to right, the original image and the results after processing by Method 4, Method 5, and Method 6 are shown. Method 4 is hard to recognize because it sharpens the blurred areas. On the other hand, Method 5 and Method 6, which remove the blur before smoothing and sharpening, make it easier to read the markers. Comparing the images of Method 5 and Method 6, we can see that Method 6 is better than Method 5 at a distance of 1 m. For a distance of 2 m, the results are similar for both methods.

We measured the processing speed of Methods 1 to 6. The PC used for measuring had an Intel Core i7-6700 3.4GHz CPU and NVIDIA GeForce GTX 1080 Ti GPU. Table I shows the processing speed of each method. Method 3 and Method 5 perform deblurring after cropping, so the processing speed varies depending on the number of detected markers. The processing speed decreases as the number of detected markers increases: 4.0 fps for 0 markers, 2.9 fps for 1 marker, and 2.3 fps for 2 markers. The processing speed of Method 6 was 2.2 fps, although it is considered to take more time because marker detection is performed after the deblurring process or for the entire input image. Processing speeds of a few fps in Table I are thought to be sufficient for the management of carts online.

D. Experiment 4: Evaluation in an Actual Factory

In this experiment, the effectiveness of the proposed system was verified by attaching the AR marker to a cart actually used in the factory.

Experimental conditions. Figure 12 shows the plan view of the experimental environment and the attached IDs. The numbers in the figure indicate the IDs of the AR markers, which were attached to each of the two support posts of the cart. The camera was mounted on a pillar and pointed horizontally toward the ground. The height of the camera was the same as that of the AR marker attached to the cart. Figure 13 shows the actual installation status of the camera. The experiment was conducted by capturing two videos, and the cart was made to move in different ways. In Route 1, the cart entered the pit surrounded by blue in front of the camera from the aisle and approached the camera; afterward, the cart was pulled back to the aisle and passed through the aisle. In Route 2, when the cart passed through the aisle, it did not enter the pit and passed straight. After passing, it was turned around from the opposite direction and passed through the aisle in front of the camera again. These routes are shown in Fig. 12. The number of successfully recognized markers was calculated by using the AR marker recognition method alone (Method 1) and the proposed systems (Methods 5 and 6) for

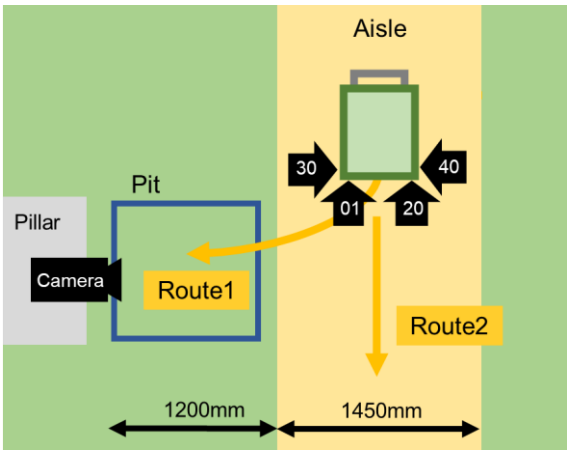

Fig. 12 Outline drawing of the experimental environment

Fig. 13 Actual installation status of the camera

Fig. 14 Image of frame that has succeeded in recognizing markers

TABLE II. Number of successes of each route

	Method1	Method5	Method6
Route1	608	689	708
Route2	0	0	0

videos of Routes 1 and 2. This experiment was conducted offline.

Figure 14 shows an image of a frame for which marker recognition has been successful. The yellow boundary box indicates a successful detection, and the blue text indicates the ID of the recognized marker.

Table II shows the number of markers successfully recognized with Methods 1, 5, and 6 in Routes 1 and 2. In the video of Route 1, the number of successful recognitions increased for the proposed system Method 5 and Method 6, as compared to the AR marker recognition method alone, Method 1. As compared with Method 5, Method 6, which deblurs the entire acquired image, increased the number of successful

recognitions. In the video of Route 2, neither the AR marker recognition method alone nor the proposed system was able to recognize the marker. This is probably due to the fact that the cart was moving too fast for successful removal of the blur. However, there were several frames in which the proposed method succeeded in detecting markers, even though it failed to recognize IDs. Therefore, it can be said that the proposed system's motion deblurring process and marker detection were also effective for Route 2.

From the above results, it was verified that the proposed system is effective for videos with AR markers attached to actual carts in a factory.

IV. Conclusion

We proposed a recognition system for AR markers attached to carts in a factory. Specifically, we proposed an AR marker recognition method that includes marker detection, cropping, and a deblurring process using deep learning. In addition, we conducted experiments to evaluate the recognition performance of the AR marker–recognition method. We then showed the effectiveness of our system through experiments on the distance between the marker and the camera and experiments on the speed at which the marker is moved.

In the experiments on distance, the distance at which the AR marker was recognized increased from 2.9 m to 3.9 m. In the experiments related to speed, the recognition rate increased from 12% to 81% for a distance of 1 m and a speed of 0.25 m/s. The processing speed was 2 to 4 fps, which is sufficient for actual use in an online environment. Furthermore, we verified the effectiveness of the proposed system by conducting an experiment in a factory assuming actual use.

As for future work, the processing speed is already practical, but further improvement in processing speed is required. In our proposed system, separate learning models are used for deblurring and marker detection. Therefore, we believe that we can improve the processing speed by combining these two learning models into a single learning model.

References

[1] Sakae Co., Ltd.: "Sakae General Catalog 2021" (p334), https://skebook.com/sakae_webcatalog2021/book/#target/page_no=427, 2021-09-01.

[2] RICOH: "Indoor location information service (for manufacturing)," https://www.ricoh.co.jp/sensing/factory/, 2021-09-01.

[3] Y. Wang, Z. Zheng, Z. Su, G. Yang, Z. Wang, and Y. Luo, "An improved ArUco marker for monocular vision ranging," Chinese Control and Decision Conference (CCDC), 2020.

[4] H. Ishii, Z. Bian, H. Fujino, T Sekiyama, T. Nakai, A. Okamoto, H. Shimoda, M. Izumi, Y. Kanehira, and Y. Morishita, "Augmented reality applications for nuclear power plant maintenance work," International Symposium on Symbiotic Nuclear Power Systems for the 21st Century (ISSNP), 2007.

[5] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," Pattern Recognition, Vol.47, No.6, pp.2280–2292, 2014.

[6] OpenCV: "Detection of ArUco Markers," https://docs.opencv.org/4.5.2/d5/dae/tutorial_aruco_detection.html, 2021-09-01.

[7] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," Neural Information Processing Systems (NIPS), 2015.

[8] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," Computer Vision and Pattern Recognition (CVPR), 2014.

[9] R. Girshick, "Fast R-CNN," IEEE International Conference on Computer Vision (ICCV), 2015.

[10] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu. "High performance visual tracking with Siamese region proposal network," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[11] Q. Wang, L. Zhang, L. Bertinetto, W. Hu, and P. H. S. Torr, "Fast online object tracking and segmentation: A unifying approach," Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

[12] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," arXiv:1406.2661 [stat.ML], 2014.

[13] O. Kupyn, T. Martyniuk, J. Wu, and Z. Wang, "DeblurGAN-v2: Deblurring (orders-of-magnitude) faster and better," Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2019.

[14] O. Kupyn, V. Budzan, M. Mykhailych, D. Mishkin, and J. Matas, "DeblurGAN: Blind motion deblurring using conditional adversarial networks," Proceedings of the IEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[15] IO DATA Co., Ltd.: "Network Camera Qwatch TS-WRLP," https://www.iodata.jp/product/lancam/lancam/ts-wrlp/index.html, 2021-09-01.

[16] T. Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft COCO: Common objects in context," Computer Vision and Pattern Recognition (CVPR), 2015.

[17] Facebook AI: "Detectron2," https://ai.facebook.com/tools/detectron2/, 2021-09-01.