Regular Paper

Contour Detection Method by CycleGAN Using CG Images as Ground Truth

Tsukasa Kudo†

†Faculty of Informatics, Shizuoka Institute of Science and Technology, Japan
kudo.tsukasa@sist.ac.jp

Abstract - Recently, various studies on object recognition have been conducted, with remarkable development, especially using deep learning. Here when the target objects are relatively small and arranged in high density in the image, its contour detection is often required as a preprocessing. Several methods have also been proposed for contour detection using deep learning, including generative adversarial networks (GANs) to perform pixel-by-pixel contour detection. However, it is necessary to prepare ground truth images that have a one-to-one correspondence with the target images for these methods. Thus, their application is often challenging. Cycle-consistent adversarial networks (CycleGAN) translates images between two domains and can be trained without preparing one-to-one ground truth images corresponding to the target images. Therefore, this study proposes a contour detection method through CycleGAN with efficiently generated ground truth images using computer graphics. Furthermore, through the comparative evaluations with the case of using ground truth images with a one-to-one correspondence with the target images, it is shown that the accuracy of the proposed method can be achieved close to this case.

Keywords: CycleGAN, Contour detection, Image recognition, Computer graphics, CG, Image-to-image translation

1 INTRODUCTION

Recently, object recognition in images has been used in various fields, and necessary information can be automatically extracted. When the target object is relatively small in the image, e.g., as in the face recognition, the object is first detected as a bounding box; then, the object recognition is performed for this area [26]. However, when objects are arranged in high density, a more precise method, such as contour detection, is often required.

As a research field, the author has been dealing with inventory management of parts in machine assembly factories stored in bulk containers (hereinafter, container). Since a factory often has thousands or more containers, efficient data collection and automatic information extraction are required. Thus, the author has been studying how workers wear wearable cameras and automatically collect images of containers. However, since the containers are densely arranged, contour detection is required to determine the target container.

Several methods using deep learning (DL) are proposed for contour detection [15], [27]. It has been shown that the target area or contour can be detected in a pixel-by-pixel manner using these methods. However, since these methods require a one-to-one correspondence with the original and its ground truth images as training data, it is often challenging to actual application.

Meanwhile, Cycle-consistent adversarial networks (CycleGAN) have been proposed as a kind of generative adversarial networks (GAN) [5], [31]. Moreover, it has been shown that mutual image translation between two types of images can be performed using CycleGAN, such as between horses and zebras. Although both images must be prepared for the training data, it is not necessary to make the above-mentioned correspondence of each data in both images. This suggests that the training data of CycleGAN for translating an image, including target objects into one with detected contours of the objects, can be prepared efficiently.

Therefore, this study demonstrates the feasibility of the method using the ground truth images of the contour efficiently created by computer graphics (CG) for training the CycleGAN model. For this purpose, two types of ground truth images are prepared by targeting photos of the books arranged on the bookshelf. One is the above CG images for the proposed method, and another is the contour images being a one-to-one correspondence with the targeting photos. Furthermore, the comparative evaluations of accuracy between both cases are performed. Consequently, It is shown that the accuracy of the former using the proposed method, can be close to the latter case.

The remainder of this paper is organized as follows. Section 2 describes this study's motivation and related works. Section 3 presents the proposed contour detection method using CG images. Section 4 shows the implementation of the experimental system. Section 5 presents the evaluations of the proposed method. Section 6 discusses the evaluation results. Finally, Sec. 7 presents the conclution.

2 MIOTIVATION AND RELATED WORKS

2.1 Motivation of This Study

Currently, a huge amount of data is input as big data from various sensors with the progress of the Internet of things (IoT). Additionally, several attempts have been made to automatically extract useful information from these data by using DL to achieve high discrimination accuracy [16], [25]. For example, to monitor targets by images, a large number of cameras, such as surveillance, in-vehicle, river, and wearable cameras, have been used, and necessary information is automatically extracted from several videos and images.

Regarding such IoT applications, the author has been study-

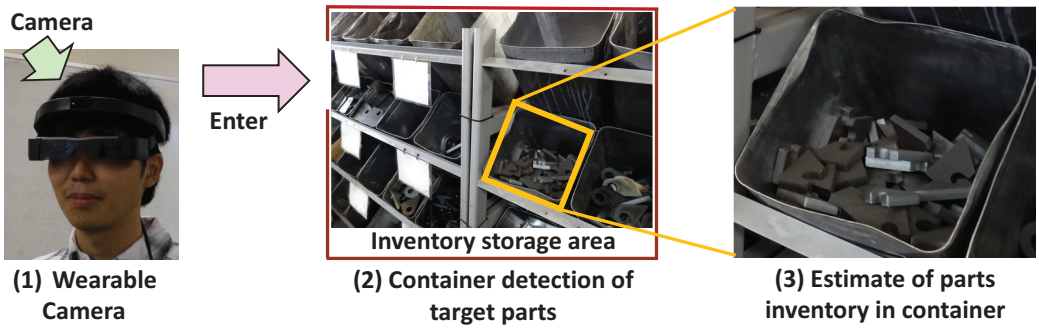


Figure 1: Automatic inventory management using videos

ing automated inventory management in machine assembly factories. Various parts are stored in containers (Fig. 1 (3)), and they cannot be counted visually from the outside. The number of these containers often reaches several thousand. For this issue, this study focused on the fact that inventory fluctuates only with inventory operations, such as the worker's replenishing or shipping of parts. Additionally, this study extracts the necessary inventory information automatically from the videos shot through the wearable camera worn by the worker (Fig. 1 (1)). Workers can take the video of the target objects without extra load by shooting video continuously while working.

Furthermore, this study showed that some necessary information for inventory management could be extracted automatically by applying DL to these videos in the previous study, such as determining workers' positions and estimating inventory of parts in containers [9], [10]. When the target is relatively small in the image, object detection is required before object recognition. Additionally, when the target is moving against the background, such as the worker picking up parts during the work, it could be detectd using the optical flow [11]. However, each container does not move against its background, also containers are densely arranged, as shown in Fig. 1 (2). Thus, container detection remains an issue.

The motivation for this study is to develop a method to efficiently detect each object's area from such dense objects including the preparation of training data for DL.

2.2 Related Works

To detect objects, as shown in Fig. 1 (2), some methods have been proposed in the conventional image processing fields. In contour tracking, the target's contour is detected to identify its area. In edge detection, the boundary of the target is specified using a filter or the like. In segmentation, mean-shift clustering, pixel value, or texture gradient determines the target area [28]. However, these contour detection methods using image processing are challenging when contours are incomplete or not closed [4].

With the progress of DL, several studies on object detection have been actively conducted, and various methods have been proposed. One is based on the region proposal that detects the bounding boxes where the objects exist. You Only Look Once (YOLO) have achieved high efficiency by detecting images using Convolutional Neural Network (CNN) processing only once; Single Shot Detector (SSD) has been used to detect objects of various sizes with one processing, and RatinaNet improved its efficiency [13]–[15], [17], [18]. Furthermore, some methods have been demonstrated for dense objects to detect the individual object's region, such as detecting new objects repeatedly (IterDet) and separating duplicate regions by post-processing [1], [19]. However, the exact area of targets cannot be specified since these region proposals use bounding boxes, namely rectangles.

Several methods have been proposed to detect object regions or contours in a pixel-by-pixel manner. A basic method has been proposed where each pixel examines an object's contour using CNN [22]. Semantic and instance segmentation methods have been proposed based on the region proposal, and they are segmentation for each object's class and object itself [7], [30]. Furthermore, after the method of image-to-image translation using conditional GANs (cGAN) between original and feature images was shown as pix2pix [8], methods using GAN have been investigated actively [12], [23], [27], [29]. However, since these methods require a one-to-one correspondence with the original and ground truth images as training data, their practical applications are often challenging.

CycleGAN has been proposed as a GAN for image-to-image translation. It can be used to translate images in a domain into the ones in another domain mutually. The previous study showed examples of mutual translation such as between photos and painter's drawings, and summer and winter photos [31]. Regarding the above challenges for practical use, CycleGAN has an essential feature that it is not necessary to make one-to-one correspondence with images between two domains as training data, i.e., it is easier to prepare training data.

Various applications of CycleGAN have been proposed. The first is the augmentation of training data used in the fields where it is difficult to generate sufficient training data for DL, such as medical treatment and detection of plant lesions [21], [24]. The second is to translate the original image into an image that is easier to detect objects as a preprocessing; methods that combine YOLO and RetinaNet have been proposed [3], [20]. However, the study targeting contour detection of densely arranged objects could not be found.

This study aims to develop a method to efficiently detect the contours of densely arranged objects, as shown in Fig. 1

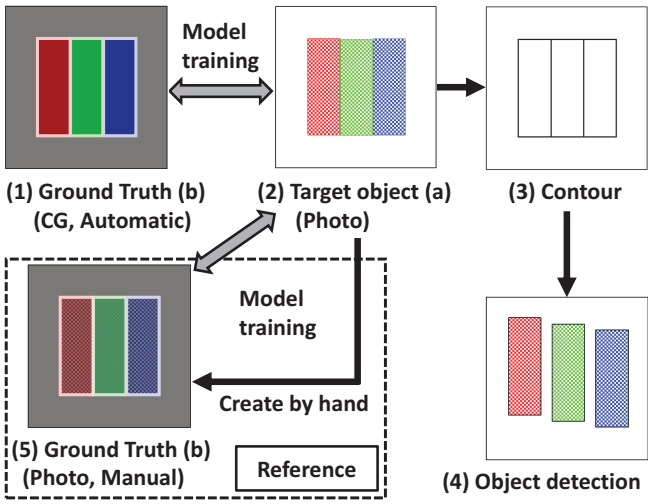


Figure 2: Contour detection method by CycleGAN using CG images as ground truth

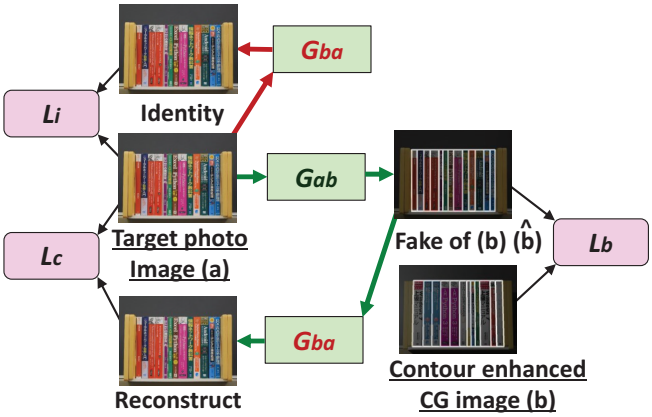


Figure 3: Configuration of CycleGAN model in this study

(2), by utilizing the above advantage of CycleGAN. Here, it is assumed that these contours are used for object recognition. Thus, this study employs ground truth images generated automatically using CG and not paired with the target images. The comparative evaluations of the following CycleGAN models are performed. One model is trained with the above CG images. Another model is trained with ground truth images that have a one-to-one correspondence with the target images. In other words, the goal of this study is to show that the former model can achieve accuracy close to the latter model.

3 CONTOUR DETECTION METHOD USING CG IMAGES

Figure 2 shows the proposed contour detection method from an image, including plural dense objects using CycleGAN with CG images as the ground truth contours. As shown in this figure, (1) shows the ground truth contour CG image, indicated by b below; (2) is the target photo, indicated by a below. The CycleGAN model is trained by images of the domains a and b. As mentioned in Sec. 2, since there is no need for a one-to-one correspondence between the ground truth and target images in CycleGAN training, b can be generated automatically using the CG tool.

Figure 3 shows the configuration of the CycleGAN model used in this study. To translate between a and b mutually, the CycleGAN model has two generators shown below.

$$\hat{a} = G_{ba}(b) \tag{1}$$
$$\hat{b} = G_{ab}(a) \tag{2}$$

Here, $\hat{a}$ and $\hat{b}$ are the fake images of the image a and b, respectively. For image a, the model is trained using discriminators that monitor the following three losses.

$$L_b = \| G_{ab}(a) - b \| \tag{3}$$
$$L_c = \| G_{ba}(G_{ab}(a)) - a \| \tag{4}$$
$$L_i = \| G_{ba}(a) - a \| \tag{5}$$

Here, $\|\|$ indicates an error, and similar losses are monitored for b. L_b evaluates the error between the fake image $\hat{b}$ and b; L_c evaluates the reconstruction image generated by applying these two generators sequentially, namely, between the fake image $\hat{a}$ and a; L_i evaluates the identity image generated from itself through the generator that generating its fake image. In training, these losses are added with a specified weight to make the total loss.

In other words, in the CycleGAN model, the image a is regenerated via the fake image of b, and the regenerated image is distinguished from original a by discriminator L_c. By using this configuration, it is not necessary to prepare the ground truth image that has a one-to-one correspondence with a, namely the contour enhanced image of a itself. For reference, Fig. 2 (5) shows a case of creating a contour-enhanced image from the photo shown in (2) instead of generating it using CG tool. Here, it is necessary to specify the contour of each object manually.

It is necessary that a and b are translated to each other by G_{ab} and G_{ba} in the training. However, since a contour image shown in (3) of Fig. 2 cannot be translated to a, a contour-enhanced image is used for b. That is, the contours are brighter than the threshold, and the rest are darker in b, so b can be translated to a. In addition, this also applies to (5) of Fig. 2.

Using the trained CycleGAN model, a is translated to a corresponding fake contour-enhanced image. Then, the contour image shown in Fig. 2 (3) is obtained by extracting the region where the brightness is above the threshold value from this image. And, using this contour, each target object can be detected, as shown in (4), for their recognition.

4 IMPLEMENTATION OF EXPERIMENTAL SYSTEM

4.1 Contour-enhanced Image Generation Using CG

Figure 4 shows the photo and contour-enhanced CG image of books arranged on the bookshelf, which are the target of this study. They correspond to (2) and (1) of Fig. 2, respectively.

(a) Target photo image (b) Contour enhanced CG image

Figure 4: Target image of experiment: books on bookshelf

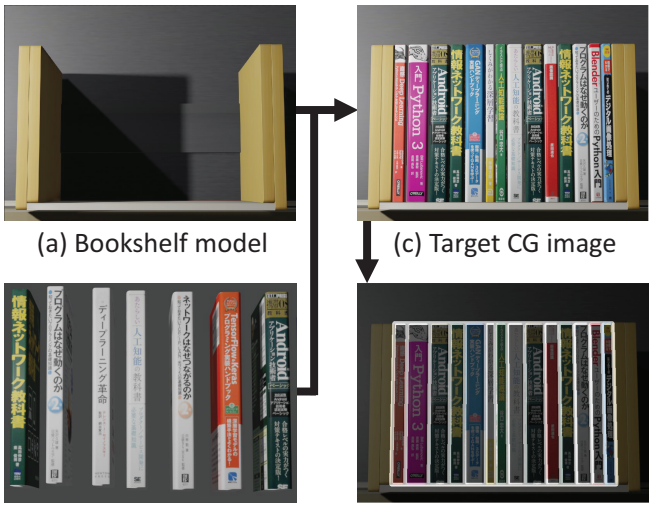


Figure 5: Contour-enhanced CG image generation procedure

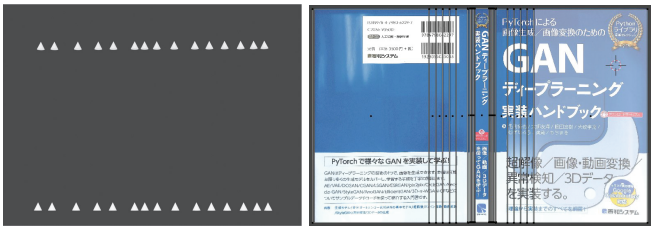
Figure 5 shows the contour-enhanced image generation procedure using a CG tool. First, a bookshelf model (a) and book models (b) are created. Next, books are randomly selected from (b) and arranged on (a) from the left. At the end of this process, books are selected according to the remaining space so that the width of the bookshelf and arranged books match. Finally, the CG image (c) of books on the bookshelf (hereinafter, bookshelf) is generated. Since the placement position and width of each book are grasped when the book is arranged, the contour information of all books can be obtained.

And, using this contour information and the image in (c), a contour-enhanced CG image (d) is generated by converting the brightness as follows.

$$f(x) = \begin{cases} cx & x \notin contour \\ cx + t & x \in contour \end{cases} \tag{6}$$

Here, t is the threshold value mentioned in Sec. 3, and it was set to 156 in anticipation of errors; c is the coefficient of conversion, and it was set to $100/255$. Consequently, since the brightness in the range of $[0, 255]$ is converted to $[0, 100]$ and $[156, 255]$ respectively, the contour area is divided from the other areas.

The CG images were generated using Blender Ver. 2.93.0, and OpenCV Ver. 3.4.2 with Python Ver. 3.7.10 for the image



(a) Vertex models of books' contour (b) Cover image and bevels of book

Figure 6: Contour vertices and book modeling

processing required after generation. First, the bookshelf and books models (hereinafter, bookshelf models) were created (Figs. 5 (a) and (b)). Then, using Python script in Blender's Scripting workspace, the target CG images and their contour-enhanced CG images shown in (c) and (d) were generated automatically and repeatedly. The CG images of (c) and (d) were rendered and saved after each time generation.

The position of each contour's vertex of the rendered image is accompanied by camera-induced distortion. As shown in Fig. 5 (c), the center of the camera's field of view, which is set at the bottom of the central book, is large, and the periphery is small. Therefore, as shown in Fig. 6 (a), the rendering is performed on a model where cones are placed at the position of the vertices at first. Then, the position information of vertices is obtained from this rendered image to create a contour-enhanced CG image.

It was presumed that the information on the boundaries of books is important for contour detection using CycleGAN shown in Sec. 3. As shown in Fig. 4 (a), there are black boundary lines generated from the gap between the books. In the CG book model (b), the left and right corners of the back cover are rounded, and shadow is generated by illuminating it from the side. Consequently, the book boundaries become clear when books are arranged (Fig. 4 (c)). Its concrete method is shown as follows.

In book modeling, the cover image of the book was created using a scanned image of its book jacket. First, a rectangular model to fit the size of the book is created. Then, made its cover image corresponds to the scanned jacket image using the UV Editing workspace of Blender. After that, the above corners were rounded using the bevel function of Blender. These results are shown in Fig. 6 (b). The vertical lines near the center are the bevel positions converted to smooth roundness at rendering. For the bevel function, the width was set to 0.5; the number of segments was set to 10; the shape was set to 0.7.

Figure 7 shows the rendering environment. A point light was placed on the left side to create shadows between books, and a spotlight was placed to prevent the right side from being dark. The position of the camera and black plane, standing behind the bookshelf, was adjusted to match the shooting environment of the photo shown in Fig. 4 (a). The positions of the point light, spotlight, and camera are $(10.4, -10.4, 4.0)$, $(9.9, -10.1, 10.9)$, $(20.0, 0.0, 0.0)$ when the origin is at the bottom of the back of the central book.

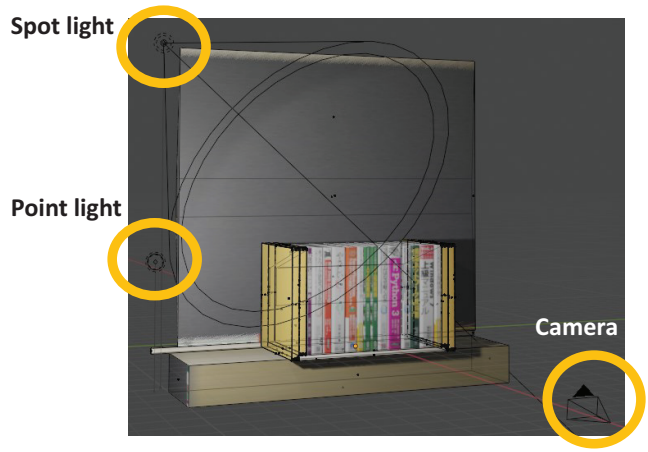


Figure 7: Rendering environment for CG images

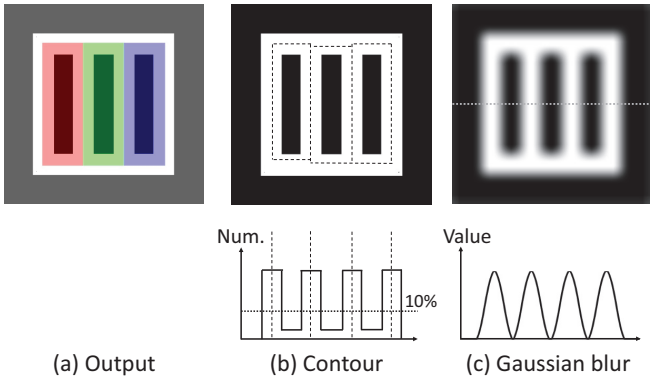


Figure 8: Contour detection and blur for metrics

4.2 Implementation of CycleGAN Model

To evaluate the effectiveness of the proposed method, an experimental system with the necessary function was implemented. This system was constructed on a personal computer, a CPU with i9-10850K (3.6 GHz), and 64 GB memory. GPU was GeForce RTX 3090 with 24 GB memory, and OS was Windows 10. Tools and programming languages were Keras Ver. 2.4.3, Tensorflow-GPU Ver. 2.4.1, and the above Python. The code of the CycleGAN model was created based on the published program code [2], along with the necessary functions for the experiment, such as the early stop control, and saving and displaying results.

For the CycleGAN model training, the above bookshelf images were resized to 128×128, and the batch size was 32. For the configuration of this model, residual networks (ResNet) were used for the shortcut connections skipping one or more layers to increase the depth of the network [6]. Additionally, the weights of the losses in Eqs. (3) to (5) were set to 4, 10, and 2. Their weighted summation was reflected for the training as the total loss.

Figure 8 shows the implementation of contour detection. Figure 8 (a) shows the output image of the CycleGAN model, the fake contour-enhanced image. It is converted to grayscale, then the contour and another area are separated based on the threshold, which was set to 111 considering the grayscale error. Consequently, the contour area is white, and another area is black as shown in Fig. 8 (b). The line segment of the contour area shows the detected contour in the next step.

Contour detection was performed on the image in Fig. 8 (b), and the vertical contour was first detected. Firstly, the number of pixels in the contour area in the vertical direction for each horizontal position was calculated as shown in the lower graph of Fig. 8 (b). Then, the horizontal area with 10% or more contour pixels was selected as a contour candidate area. For the case of Fig. 8 (b), there were four candidate areas. The position of the largest number of pixels was selected as the vertical part of the contour for each area, which is indicated by the broken line.

Next, the number of pixels in the contour area of the horizontal direction was counted between each adjacent vertical part of the above contours. The vertical area with 50% or more numbers was selected as the horizontal part candidates of the contour. Similar to the vertical part, the horizontal part of the contour was selected. Then, the contour of each object was detected, as shown by the broken line in Fig. 8 (b). They were rectangles, in this experiment.

In the training of the CycleGAN model, the following contour differences were used as metrics to determine the completion of the training. One is the contour created from b shown in Fig. 2 (1), which is the ground truth contour. The other is created from $\hat{b}$ shown in Eq. (2) called fake b. To illustrate the distance between the contour lines, Gaussian blur with a kernel size 5×5 and a standard deviation of 0.3 is performed, as shown in Fig. 8 (c). The mean absolute error (MAE) between both is used as the metric.

When the brightness of the pixel of the coordinate (i, j) is shown by g_{ij} for b and f_{eij} for $\hat{b}$ at epoch e respectively, the metric is shown as follows.

$$m_e = \sum_{i=1}^{n} \sum_{j=1}^{n} |f_{eij} - g_{ij}| / n^2 \tag{7}$$

Here, n is the number of pixels in each coordinate axis. In the training, m_e was calculated for each epoch. Every time m_e became the minimum compared before, the model weight was saved as the best weight. Then, when m_e did not improve the specified number of times, the training was completed. In this experiment, this number was set to 15 epochs.

5 EVALUATIONS

5.1 Evaluations of Contour Detection

Contour detection was conducted for both cases: in one case, the contour-enhanced images generated by CG were used; in another case, the contour-enhanced images manually created from the target photo images were used.

For the training and test data, 128 bookshelf photo images shown in Fig. 4 (a) were prepared and inflated to 256 pairs by horizontal flip. Contour-enhanced photo images were manually created from each image through the procedure shown in Eq. (6) and Fig. 8. Images of each group were randomly divided into 204 training data and 52 test data. Since CycleGAN was used, the division of the training and test data in each group did not correspond to each other. On the other

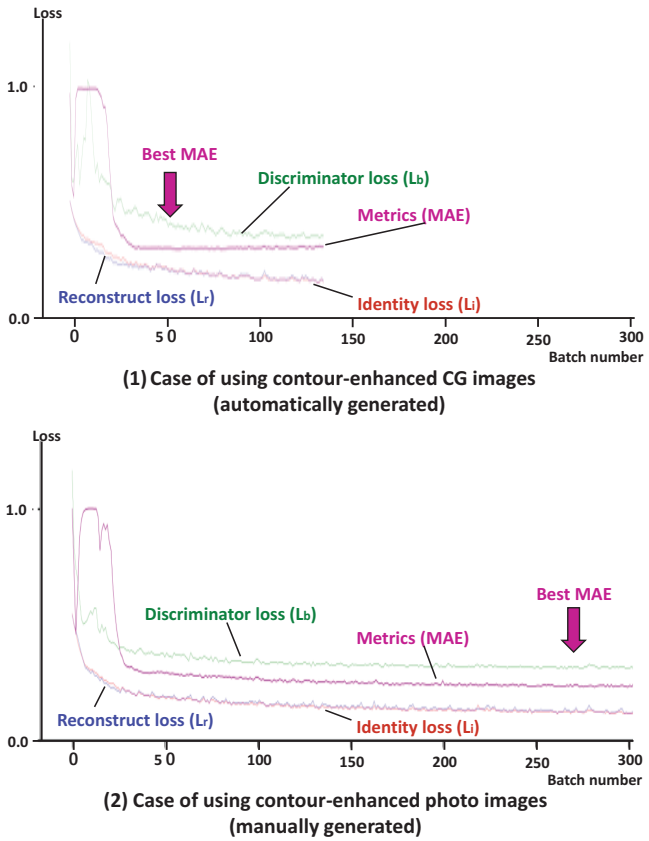


Figure 9: Transition of losses and metrics with training

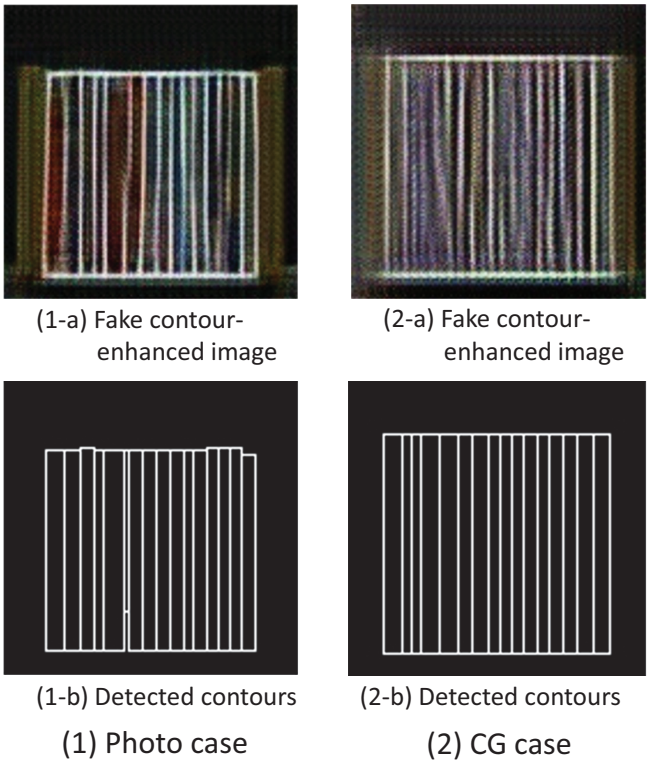


Figure 10: Fake contour-enhanced image and detected contours

hand, 204 contour-enhanced CG images were generated for the training data by the procedure shown in Sec. 4.1.

For the case of the contour-enhanced CG images (hereinafter, CG case), the CycleGAN model was trained with these CG images and the above training data of the photo without contours. The metrics were monitored using the test data simultaneously. Then, the contours were detected from the test data. For the case of the contour-enhanced photo images (hereinafter, photo case), the above contour-enhanced photo images and training data of photo were used. Here, the metrics monitoring and contour detection were performed by the same data with the CG case.

Figure 9 shows the transition of losses and metrics with training shown in Eqs. (3) to (5) and (7). The horizontal axis of the figure shows the batch number, and one epoch corresponds to five batches. During the period immediately after the start of training, each loss and metrics showed individual tendencies. Then, they showed the same tendency. The arrow of "best MAE" indicates the epoch number when the metrics become the best. In the CG case, the best MAE 0.303 was obtained at epoch number 10. Similarly, in the photo case, 0.2278 was obtained at epoch 54.

Figures 10 (1-a) and (2-a) show fake contour-enhanced images generated from the bookshelf image shown in Fig. 4 (a) using the trained model for the above two cases, respectively. Their detected contours using the procedure shown in Fig. 8 (b) are shown in Figs. 10 (1-b) and (2-b), respectively. As shown in Fig. 10 (2), the affine transformation of the above bookshelf image is performed so that the book area of the photo matches the area of the CG image shown in Fig. 5 (b). So, there is a difference in image size between Figs. 10 (1) and (2).

As shown in the upper row of Figs. 10 ((1-a) and (2-a)), the fake contour-enhanced images of the target images were generated in each case, although there are some distorted and blurred parts. Additionally, contours were detected from these images, as shown in the lower row of Figs. 10 ((1-b) and (2-b)). However, some contours are too narrow in width or inconsistent in height compared to books.

5.2 Evaluations of Object Recognition

To evaluate the accuracy of the contour detection, object recognition accuracy was evaluated by template matching of the back cover images between the target photo books, as shown in Fig. 4 (a), and the extracted books using the detected contours. The former back cover images were created using the contours set manually, as mentioned in Sec. 3. The latter books were extracted from the target photo books using the detected contours of each case. Additionally, test data excluded inflated images by horizontal flip were used for target photo books.

Figure 11 shows the template matching results of the CG case. Template matching was performed using matchTemplete function of OpenCV, where the normalized correlation coefficient matching was used. Fig. 11, (1) shows the back covers of target photo books; (3) shows one created using detected contours. (2) shows the template matching results between the two, and the back cover images are the same as (1), and the image of (3) is shown by a white rectangle. Similarly,

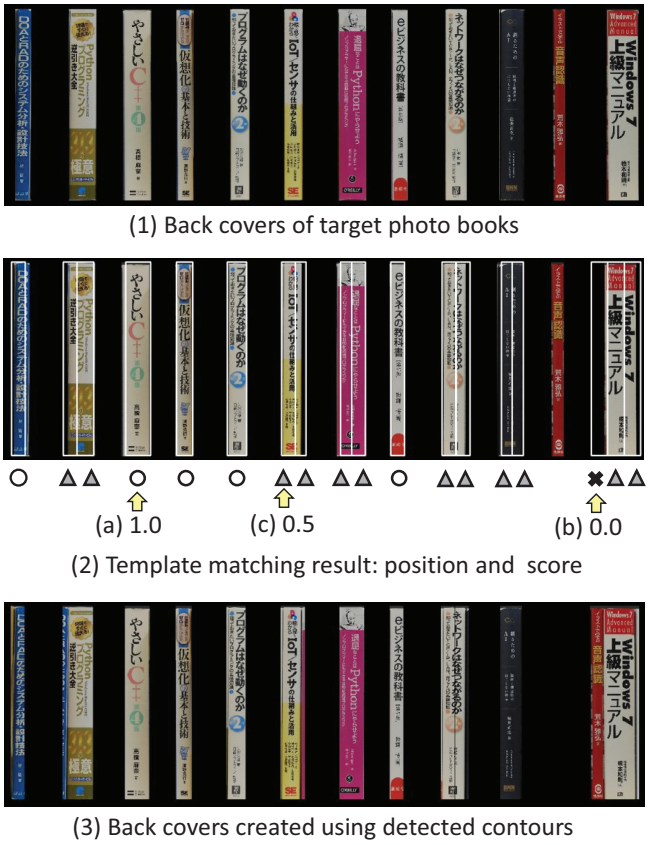


Figure 11: Book recognition results using detected contours of CG case

Fig. 12 shows the results of the photo case.

The following three scores were introduced according to the matching level to evaluate the accuracy of object recognition, as shown in Fig. 11 (2). (a) shows the case of almost the same indicated by ◯, which score is 1.0; (b) shows the case of not almost overlap indicated by ×, which score is 0.0; (c) shows the case in between both of the above indicated by △, that is, there is an extra area or only a part of both is overlap. Here, (c) is the case when 50% or more of the detected contour area is included in one of the target photo books; (b) is the case of less than 50%.

Figure 11 (2) and Fig. 12 show that there are differences in the results of each book recognition. For example, the second back cover from the left is detected by dividing into two in Fig. 11 (2), but it is detected as one in Fig. 12. Additionally, the third back cover from the right is recognized as score 0.5, and the second is not recognized in Fig. 11 (2). For these back covers, the recognition result was reversed in Fig. 12.

Figure 13 shows the percentage of back covers recognized with scores of 1.0 and 0.5 out of a total of 237 back covers. Here, even if the back cover is recognized twice with a score of 0.5, such as the book on the right end of Fig. 12, it is counted as one book. When only the score of 1.0 was targeted, the recognition rate was 55.7% for the CG case, lower than 59.9% for the photo case. However, including the score of 0.5, it was 84.5% and 81.5%, and the CG case was higher than the photo case.

To analyze the status of the score, the average score in each

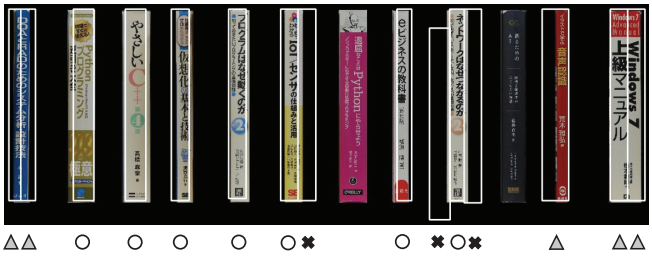


Figure 12: Book recognition results using detected contours of photo case

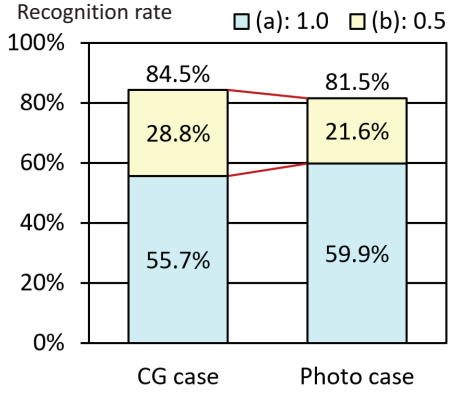


Figure 13: Book recognition rate using detected contour

target book image is shown in Fig. 14, and the total number of each score is shown in Fig. 15. As shown in Fig. 14, the line graph shows the result arranged in descending order to grasp the score's tendency between each case. Note that even if the image number in this figure is the same, both results are not for the same target book image. The score was from 0.87 to 0.39 in the CG case and 1.0 to 0.43 in the photo case, and the averages were 0.63 and 0.69, respectively. That is, the average of the CG case was 9.5% lower than the photo case.

Figure 15 shows the total number of the target photo books is 237, and the number of detected contours exceeds this in both cases. There are 379 detected contours, which is more than the 328 in the photo case. The number of the score of 1.0 is 162. Similarly, the number of the score of 0.5 is 150.

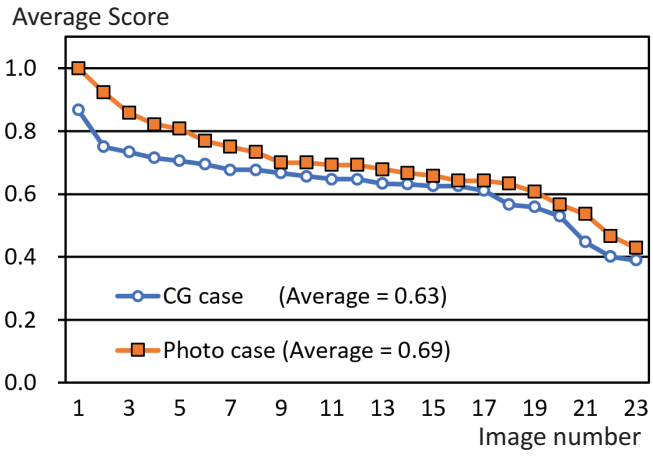


Figure 14: Average score of book recognition for each image

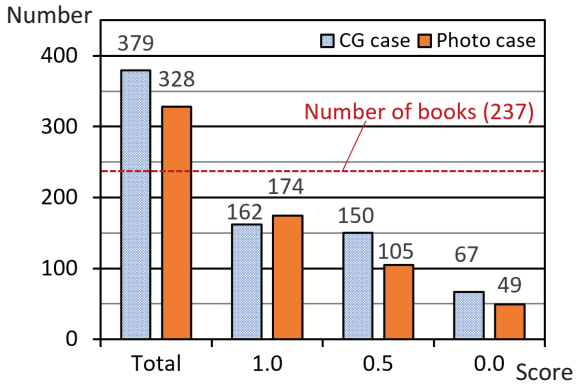


Figure 15: Number of each score in book recognition

Figure 16: Evaluation of contour-enhanced image using target images of different sizes

Comparing the photo case, though the former is less than 174 of the photo case, the latter is more than 105.

5.3 Evaluations of Influence with Contour Size Error

The influence of the error in the size of the photo image of the book and the contour-enhanced CG image was evaluated. There was also an error in their positions. Thus, this experiment was conducted without closely matching their sizes by the affine transformation mentioned in Sec. 5.1. Figure 16 shows the images used in this experiment. Figure 16 (a) shows a book photo image, and Fig. 16 (b) shows a book CG image from which a contour-enhanced image was created. The latter is a little larger, especially positions of the upper sides of the contours are high. Figure 16 (d) shows the fake contour-enhanced image generated using the model trained with these images. Here, Fig. 16 (c) is a photo image

Figure 17: Book recognition results using target images of different sise

of the book converted to the size of 128×128 pixels to input the model. In Fig. 16 (d), the position of the upper side is higher than in (c), similar to the CG image of the book shown in (b).

Contour detection was performed using this fake contour-enhanced image shown in Fig. 16 (d). Then, each book image was extracted by these contours from the bookshelf photo image shown in Fig. 16 (a). Figure 17 shows the results of book recognition using detected contours similar to Sec. 5.2.

As shown in Fig. 17, the contour is detected as large size, and most of them have an empty area at the top. In the matching result of the second book from the left, the bottom of the rectangle is omitted that extends beyond the figure.

6 DISCUSSIONS

This study shows that it is possible to perform contour detection using CycleGAN without preparing ground truth images with a one-to-one correspondence to the target images. For the books arranged on the bookshelf, the comparative evaluation was performed on the accuracy of the target book's back cover recognitions using the detected contour by the CG and photo cases (Figs. 11 and 12).

Consequently, the almost same accuracy was obtained in these two cases (Fig. 13). At a score of 1.0, the accuracies of the CG and photo cases were 55.7% and 59.9%, respectively, meaning that the photo case was higher than the CG case. However, at scores of 1.0 and 0.5, the accuracies of the CG and Photo cases were 84.5% and 81.5%, meaning that the CG case was higher than the photo case.

The score 1.0 ratio was lower in the CG case is considered because the ratio of divided contour for a back cover is more than in the photo case (Fig. 15). For example, the contour of one back cover was detected as two contours of score 0.5 as shown in the second back cover from the left in Fig. 11 (2). The score of 1.0 was less in the CG case than in the photo case, but that of 0.5 was more. This is a general tendency, as shown in Fig. 14. The average score of the CG case is lower than that of the photo case.

To improve the score 1.0 ratio in the CG case, it is conceivable to adjust the threshold for detecting vertical contours shown in Fig. 8, which is currently set to 10%, to an appropriate value. As shown in Fig. 10 (2-a), since the fake contour-enhanced image has distortion and blurring of the contour area, it is expected that these parts can be excluded by setting the threshold higher.

Furthermore, to improve the contour detection accuracy,

it is considered that the method to incorporate the metrics, which are used for only the early stop of training the model in this study, as a discriminator of the CycleGAN model to feedback error between the following. One is the detected contour created by a fake contour-enhanced CG image, as shown in Fig. 10 (2-b). Another is the ground truth contour created from the CG vertex model, as shown in Fig. 6 (a). These improvements are feature challenges.

The experiment showed that the ground truth images could be automatically generated by the CG tools using the prepared bookshelf and book models in advance, instead of manually creating each ground truth image. This method using CG images is considered effective in the field, where many objects with relatively few types are randomly arranged. It means that various application fields of contour detection are expected such as objects placed on desks and cars on roads.

As shown in Fig. 17, using the CycleGAN model, the difference of scale and position between the target and ground truth images is significantly affected on the detected contour. Therefore, depending on the application field, it may be effective to match the shapes of the target and ground truth image by preprocessing.

Evaluations of the effectiveness of the proposed method in environments with these various shapes and backgrounds are feature challenges.

7 CONCLUSIONS

To recognize one object in objects arranged in high density, contour detection is often required to preprocess. In this field, various methods have been proposed utilizing GAN to perform pixel-by-pixel detection. However, with these methods, it is necessary to prepare ground truth images that have a one-to-one correspondence with the target images. Thus, there is often an obstacle to their application. Meanwhile, the CycleGAN model can translate images from two different domains to each other without preparing such one-to-one correspondent ground truth images.

Therefore, this study proposed a method to use the ground truth images automatically generated by CG tool utilizing the advantages of CycleGAN. Furthermore, the comparative evaluations between this method and the approach of using the ground truth images that had a one-to-one correspondence with the target images were conducted. The accuracy of image recognition using template matching based on detected contours was compared. Consequently, the proposed metho's accuracy could be close to the case of using the conventional one-to-one correspondence ground truth images, even when the ground truth CG images were used.

Future challenges include the improvement of contour detection accuracy using the proposed method and the evaluation of the accuracy in the case of applying this method to other application fields.

ACKNOWLEDGMENTS

This work was supported by JSPS KAKENHI Grant Number 19K11985.

REFERENCES

[1] X. Chu, A. Zheng, X. Zhang, and J. Sun, "Detection in crowded scenes: One proposal, multiple predictions," Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition, pp. 12214–12223 (2020).

[2] D. Foster, "Generative deep learning: Teaching machines to paint, write, compose, and play," O'Reilly Media (2019), https://github.com/davidADSP/GDL_code (referred May 24, 2021).

[3] P. Gao, T. Tian, L. Li, J. Ma, and J. Tian, "DE-CycleGAN: An object enhancement network for weak vehicle detection in satellite images," IEEE J. Selected Topics in Applied Earth Observations and Remote Sensing, Vol. 14, pp. 3403–3414 (2021).

[4] X. Y. Gong, H. Su, D. Xu, Z. Zhang, F. Shen, and H. Yang ,"An overview of contour detection approaches," Int. J. Automation and Computing, Vol. 15, No. 6, pp. 656–672 (2018).

[5] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, and Y. Bengio, "Generative adversarial networks," Communications of the ACM, Vo. 63, No. 11, pp.139–144 (2014).

[6] K. He, X, Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," Proc. IEEE Conf. Computer Vision and Pattern Pecognition, pp. 770–778 (2016).

[7] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," Proc. IEEE Int. Conf. Computer Vision, pp. 2961–2969 (2017).

[8] P. Isola, J. Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," Proc. IEEE Conf. Computer Vision and Pattern Recognition, pp. 1125–1134 (2017).

[9] T. Kudo, and R. Takimoto, "CG utilization for creation of regression model training data in deep learning." Procedia Computer Science, Vol. 159, pp. 832–841 (2019).

[10] T. Kudo, "A proposal for article management method using wearable camera," Procedia Computer Science Vol. 178, pp. 1338–1347 (2020).

[11] T. Kudo, "Moving object detection method for moving cameras using frames subtraction corrected by optical flow," Int. J. Informatics Society, Vol. 13, No. 2, pp. 79–91 (2021).

[12] M. Li, Z. Lin, R. Mech, E. Yumer, and D. Ramanan, "Photo-sketching: Inferring contour drawings from images," 2019 IEEE Winter Conf. Applications of Computer Vision (WACV), pp. 1403–1412 (2019).

[13] T. Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," Proc. IEEE Conf. Computer Vision and Pattern Recognition, pp. 2117–2125 (2017).

[14] T. Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection." Proc. IEEE Int. Conf. Computer Vision, pp. 2980–2988 (2017).

[15] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen, "Deep learning for generic object detection: A survey," Int. J. Computer Vision, Vol. 128, No. 2, pp. 261–318 (2020).

[16] M. Mohammadi, A. Al-Fuqaha, S. Sorour, and M. Guizani, "Deep learning for IoT big data and streaming analytics: A survey," IEEE Communications Surveys and Tutorials, Vol. 20, No. 4, pp. 2923-2960 (2018).
[17] M. Rajiput, "YOLOv5 is here! elephant detector training using custom dataset & YOLOV5," https://towardsdatascience.com/yolo-v5-is-here-b668ce2a4908 (referred May 24, 2021).
[18] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," Proc. IEEE Conf. Computer Vision and Pattern Recognition, pp. 779–788 (2016).
[19] D. Rukhovich, K. Sofiiuk, D. Galeev,O. Barinova, and A. Konushin, "IterDet: Iterative scheme for object detection in crowded environments," arXiv preprint arXiv:2005.05708 (2020).
[20] K. Saleh, A. Abobakr, M. Attia, J. Iskander, D. Nahavandi, M. Hossny, and S. Nahvandi, "Domain adaptation for vehicle detection from bird's eye view LiDAR point cloud data," Proc. IEEE/CVF Int. Conf. Computer Vision Workshops, pp. 3235–3242 (2019).
[21] V. Sandfort, K. Yan, P. J. Pickhardt, and R. M. Summers, "Data augmentation using generative adversarial networks (CycleGAN) to improve generalizability in CT segmentation tasks," Scientific Reports, Vol. 9. No. 1, pp. 1–9 (2019).
[22] W. Shen, X. Wang, Y. Wang, X. Bai, and Z. Zhang, "Deepcontour: A deep convolutional feature learned by positive-sharing loss for contour detection," Proc. IEEE Conf. Computer Vision and Pattern Recognition, pp. 3982–3991 (2015).
[23] A. Sindel, A. Maier, and V. Christlein, "Art2Contour: Salient contour detection in artworks using generative adversarial networks," 2020 IEEE Int. Conf. Image Processing (ICIP), pp. 788–792 (2020).
[24] Y. Tian, G. Yang, Z. Wang, E. Li, and Z. Liang, "Detection of apple lesions in orchards based on deep learning methods of cyclegan and YOLOv3-dense," J. Sensors, Vol. 2019, Article 7630926 (2019).
[25] M. Verhelst, and B. Moons, "Embedded deep neural network processing: Algorithmic and processor techniques bring deep learning to IoT and edge devices," IEEE Solid-State Circuits Magazine, Vol. 9, No. 4, pp. 55–65 (2017).
[26] P. Viola, and M. Jones, "Rapid object detection using a boosted cascade of simple features," Proc. 2001 IEEE Computer Society Conf. Computer Vision and Pattern Recognition, Vol. 1, pp.511–518 (2001).
[27] H. Yang, Y. Li, X. Yan, and F. Cao, "ContourGAN: Image contour detection with generative adversarial network, Knowledge-Based Systems, Vol. 164, pp. 21–28 (2019).
[28] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," ACM Computing Surveys (CSUR), Vol. 38, No. 4, Article 13 (2006).
[29] Z. Zeng, Y. K. Yu, and K. H. Wong, "Adversarial network for edge detection," Int. Conf. Informatics, Electronics & Vision (ICIEV), pp. 19-23 (2018).
[30] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," Proc. IEEE Conf. Computer Vision and Pattern Recognition, pp. 2881–2890 (2017).
[31] J. Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," Proc. IEEE Int. Conf. Computer Vision, pp. 2223–2232 (2017).

(Received: October 31 , 2021)
(Accepted: January 11, 2022)

Tsukasa Kudo received the BSc and ME from Hokkaido University in 1978 and 1980, and the Dr. Eng. from Shizuoka University in 2008. In 1980, he joined Mitsubishi Electric Corp. He was a researcher of parallel computer architecture and engineer of business information systems. Since 2010, he is a professor of Shizuoka Institute of Science and Technology. Now, his research interests include deep learning and database application. He is a member of IEIEC and IPSJ.