International Journal of Control, Automation and Systems 18(9) (2020) 2306-2314
http://dx.doi.org/10.1007/s12555-019-9618-z

ISSN:1598-6446 eISSN:2005-4092
http://www.springer.com/12555

Image Preprocessing-based Generalization and Transfer of Learning for Grasping in Cluttered Environments

Kuk-Hyun Ahn and Jae-Bok Song*

Abstract: In a cluttered environment in which objects are lying very closely to each other, the arranging motion is required before the robot attempts to grasp the target object. Thus, a robot must determine which motion to perform based on a given situation. This study presents an approach to learning a decision-making ability for the robot to grasp the target object after rearranging the surrounding objects obstructing the target object. The learning is performed in the virtual environment, and the image, which is an input of the deep Q-network, is preprocessed to directly apply the results of the learning to the real environment. That is, the difference between the two environments is minimized by making the states obtained from the virtual and real environments similar to each other. In addition, image preprocessing can be used to generalize the results of learning so that the robot can determine the appropriate actions to take when objects that were not used for learning are given. A hierarchical structure, which consists of high-level and low-level motion selectors, is used for the learning: the former determines the grasping or pushing actions while the latter determines how to perform such selected actions. The results of various experiments show that the proposed scheme is effective in grasping the target object in a cluttered environment without the need for any additional learning in the real world.

Keywords: AI-based application, grasping, reinforcement learning, sim-to-real transfer.

1. INTRODUCTION

Substantial research has been conducted on grasping in consideration of the fact that grasping is the most basic task that a robot needs to perform. However, in most cases, the focus is on grasping itself, such as the cases estimating the grasp pose without considering the surrounding environment [1–3]. In studies on grasping in cluttered environments, robots usually learn to grasp any object that is expected to be most likely to be grasped without a specified target object [4–6]. This approach has certain limitations if a robot performs a task for which a target object is specified, such as fetching an object specified by a user in a home environment. Some studies have also investigated a method of performing the pushing motion to arrange the adjacent objects for grasping in a cluttered environment. In [7], the scattering motion was carried out when the pixels of the dilated image of the target object overlapped those of the surrounding objects. However, as the scattering motion was performed despite the fact that the adjacent objects did not interfere with grasping, it is not an appropriate method for decision-making. In [8], in order to grasp a target object in a cluttered environment, a task plan was scheduled for a sequence of several given motions, and the goal was accomplished by conducting these motions in sequence. There existed a limitation in that the sequence of motions was not determined by recognizing a given situation, but instead determined from the result of performing the actions in the simulator.

Deep reinforcement learning [9], which combines the reinforcement learning scheme with a deep neural network (DNN), can serve as a solution to acquiring the ability of determining which action to perform. Through the use of the DNN, the information necessary for the decision is extracted from the situation given as an image, and the robot can thus learn the optimal way by repeating the task using reinforcement learning. In [10], the action value functions for pushing and grasping motions were approximated by DNNs, and a robot learned how to grasp an object after scattering the closely-placed objects. However, in this case, the target object was not specified, and the robot grasped one of the scattered objects as a result of the pushing action. In this study, the Q-learning based reinforcement learning, which can be applied to robots for motion planning [11,12], is used to train the robot to learn the decision-making ability necessary to grasp the target object after arranging the surrounding objects, if they are interfering with the grasp.

Manuscript received August 2, 2019; revised November 29, 2019; accepted February 8, 2020. Recommended by Editor Fumitoshi Matsuno. This work was supported by IITP grant funded by the Korea Government MSIT (No. 2018-0-00622).

Kuk-Hyun Ahn and Jae-Bok Song are with the School of Mechanical Engineering, Korea University, 145 Anam-ro, Seongbuk-gu, Seoul 02841, Korea (e-mails: {gookrice, jbsong}@korea.ac.kr).
* Corresponding author.

Since a robot must repeat a large number of tasks to learn [6, 13], reinforcement learning is generally performed in the virtual environment, then the learning results are transferred to the real world. Learning can be performed more conveniently using a simulator because information about the environment can be obtained from the simulator without the need for an additional algorithm. However, it is difficult to apply the results of learning in the virtual environment to an actual robot due to the differences between the virtual environment and the real environment. Therefore, further learning should be performed in the real environment based on the learning result obtained in the virtual environment, or a scheme to reduce the difference between the two environments should be used while learning. In [14] and [15], for the agent to learn robustly against the environmental change, the color, pattern, and ambient illumination of the environment were continuously changed during the learning. In [16], the environmental difference was reduced by making the image used as an input state in the virtual environment similar to the real-world image using the DNN. In this study, a relatively simple image preprocessing method is proposed to overcome the difference between the two environments by making the image obtained from the real environment similar to the virtual environment.

Fig. 1 shows a flowchart of the proposed method for grasping in cluttered environments. For learning the decision-making ability, the robot repeatedly conducts a given task in the virtual environment where robot motion is selected from the preprocessed image through the DNN. After the training is completed, the DNN parameters are transferred to the real world, and the robot performs a motion selected from the image preprocessed to be similar to the virtual environment. The originality and contributions of this study are as follows:

- The vision-based method for learning the sequence of motions is proposed to grasp a target object in cluttered environments.

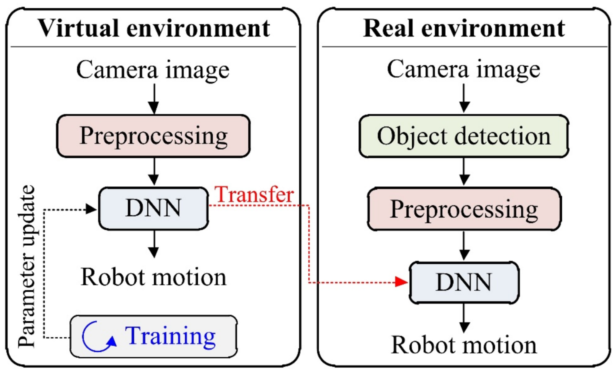

Fig. 1. Flowchart of the proposed learning method for grasping in cluttered envrionments.

- The image preprocessing method is proposed to increase the learning efficiency by excluding the elements from the image that are not necessary to determine the proper actions to grasp an object. With this method, the learning results also can be generalized to various objects that are not used in learning.
- The learning results in the virtual environment can be transferred directly to the real world using the proposed method by making the real-world image similar to the input state used in the learning process.

The rest of this paper is organized as follows: The learning structure is described in Section 2. Section 3 details the reinforcement learning method proposed in this study and the image preprocessing method to transfer the learning results to the real world, and Section 4 discusses the experimental results. Finally, a conclusion is drawn in Section 5.

2. LEARNING STRUCTURE

In this study, the deep Q-network (DQN) [17], which is an off-policy Q-learning method, is used to train the robot to learn the decision-making ability. First, the DNN approximates the action value function (Q-function), and the robot takes an action with the greatest action value in the given state s. Then, learning proceeds in the direction of minimizing the following temporal difference error δ under the greedy policy:

$$\delta = [r + \gamma \max_{a'} Q_{\theta'}(s', a') - Q_\theta(s,a)]^2, \tag{1}$$

where r is a reward obtained after an action a is performed, γ is a discount factor, $Q_\theta(s,a)$ is an action value which is estimated from the Q-network with the parameter set θ for the current state s and the selected action a, and $\max_{a'} Q_{\theta'}(s', a')$ represents the maximum action value in the next state, which is estimated using the target network with the parameter set θ'. Based on (1), the parameters of the Q-network are updated through the gradient descent method. As the Q-network comes to accurately estimate the action values through iterative learning, the robot can accomplish its goal by taking the appropriate series of actions.

In order to diversify robot motions for grasping or arranging and allow a robot to learn more efficiently, the hierarchical reinforcement learning structure [18] is implemented in this study by constructing the motion selector shown in Fig. 2. The high-level motion selector determines the motion that should be performed by the robot and outputs a goal g, and the low-level motion selector then outputs an optimal action a to perform the goal g. Each selector has an independent Q-network with the parameter set denoted as θ_h and θ_l, respectively, and respectively gains a reward, r_h and r_l, as a result of its action.

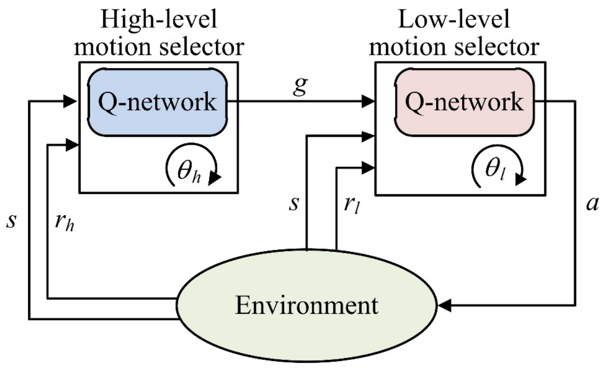

Fig. 2. Overview of hierarchical structure of a motion selector.

The temporal difference errors of both selectors can be obtained using (1). Since each Q-network receives an image as an input state, it has three convolutional layers for extracting the features from the image and outputs the action values through two fully connected layers.

3. LEARNING IN VIRTUAL ENVIRONMENT

The learning process for the target task is carried out in a virtual environment using a Gazebo simulator. A Sawyer robot is used as the agent, and an image containing the target object and the surrounding objects is obtained using a camera fixed in the space above the table on which the task is performed. In this study, in transferring the learning results to the real environment, the image obtained in the real environment is preprocessed to be similar to that obtained in the virtual environment. Thus, as shown in Fig. 3, the target object is set simply as a red rectangular parallelepiped block while the surrounding objects are set as green cubes in the virtual environment.

3.1. Robot motion

As shown in Fig. 4, the high-level selector determines the motion that the robot should perform among five motions: a grasping motion and four arranging motions. The low-level selector then determines the target position of each motion output from the high-level selector. When the high-level selector outputs a grasping motion, the low-level selector determines the grasping point among five points: the center point of the object and four offset points along the major axis. The grasp is performed along the minor axis of the object, and the center point of the object is provided by the simulator, as are the directions of the major and minor axes.

When the high-level selector outputs an arranging motion, the robot pushes the objects to the left or right side of the target object in the direction of the major axis. At this time, the low-level selector determines the starting point of the pushing motion, which is set near each vertex of the

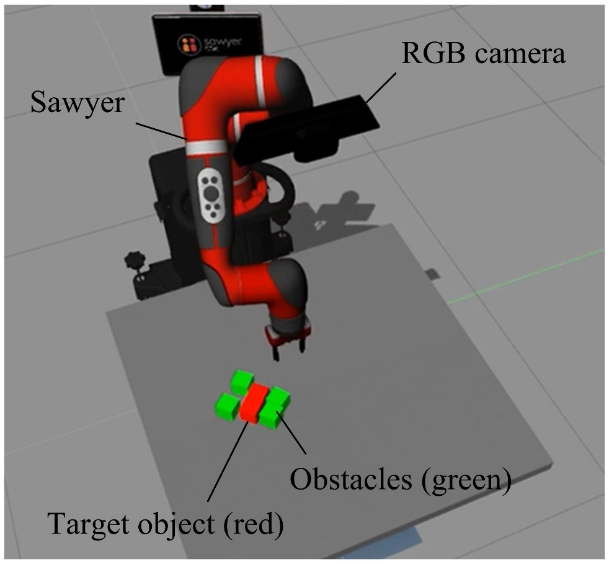

Fig. 3. Virtual environment for learning process.

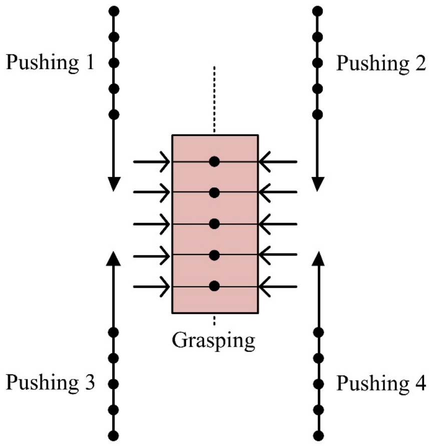

Fig. 4. Schematic description of robot motions.

target object in consideration of its length and width. Increasing the number of possible motions through the low-level selector allows the robot to perform the selected action while avoiding the surrounding objects. For example, when it would be difficult to grasp near the center point of the target object due to the presence of surrounding objects, the robot can instead pick up the object by grasping its end part.

3.2. State

In order to determine whether to directly grasp the target object or remove the surrounding object first, it is necessary to know how the objects are arranged. In this case, factors such as the direction in which the objects lie or their color are irrelevant. That is, it is only necessary to judge whether the surrounding objects interfere with the grasping of the target object according to the grasping

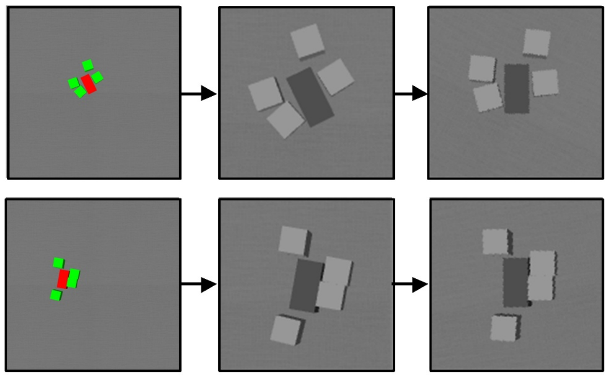

Fig. 5. Preprocessing of images for input states.

pose. Therefore, in this study, the RGB image obtained from the camera is preprocessed so that the image that will be used as a state contains only the information necessary to aid the decision of robot motion. To obtain this state, as shown in Fig. 5, only the image around the target object excluding color information is taken from the raw camera image containing the entire workspace. This image is then rotated so that the major axis of the target object is vertical.

The preprocessing of the images used for states has several advantages in the learning process. First, since the direction in which the target object is placed is excluded in learning, the range of the state to be estimated by the Q-network decreases. That is, the Q-network only estimates the action value for the input state, which always has the same direction as the target object. In addition, since only images aligned in the grasping direction of the object are used, the learning data can be collected more quickly, thus increasing the convergence rate of learning. Furthermore, by learning the action value function based only on the state depending on the arrangement of objects, it is possible to generalize the decision-making ability so that the results obtained by learning with simple-shaped objects can be applied to various objects. That is, the robot can learn, regardless of the shape and directions of objects, that picking up the target object is not possible if the adjacent objects block certain points at which the fingers of the gripper should be located to grasp the target.

3.3. Reward

As the high-level and low-level selectors each have their own reinforcement learning structures, their rewards are also defined separately. Specifically, the high-level selector is rewarded ($r_h = 1$) only when the robot successfully grasps the target object, and it is not rewarded ($r_h = 0$) otherwise, even if the surrounding objects are well removed by the robot. At this time, if the grasping motion is output but the grasping fails due to the surrounding objects, a penalty ($r_h = -1$) is imposed to clarify the learning objective.

The low-level selector is rewarded ($r_l = 1$) when the robot performs the motion output from the high-level selector, and it is not rewarded ($r_l = 0$) if the robot fails to do so. After the grasping motion is output from the high-level selector, it is considered to be successful when the robot picks up the target object over the predetermined height. All of the motions are considered to be failures if the robot is caught by an object while approaching the ground in order to perform the motion. If the robot fails to accomplish the selected motion even after performing all of the actions of the low-level selector, the high-level motion is regarded as a failure.

3.4. Learning process

The objects are randomly arranged at the beginning of the episode. The image containing the objects is transformed into the state s in the preprocessing stage described above, and is input to the Q-network of the high-level selector that outputs the goal g. The Q-network of the low-level selector receives g and s as inputs and outputs the action a. The actual motion of the robot is then performed based on g and a. The data set of (s, g, a, r_l, $s^{'}$) is stored in the replay memory D_l of the low-level selector when the reward r_l and the next state $s^{'}$ are obtained according to the result of the motion of the robot. If the robot achieves g or fails after performing repetitive motions chosen by the low-level selector, the robot obtains the reward r_h, and the data set of (s_0, g, r_h, $s^{'}$) is stored in the replay memory Dh of the high-level selector, where s_0 represents the state used to obtain g in the high-level selector.

When the robot successfully grasps the target object, the episode is terminated, and the objects are placed at new locations for the next episode. The parameters of the Q-network, θ_h and θ_l, are updated at each step based on the temporal difference error presented in (1) using the target network and a batch sample drawn randomly from D_l and D_h. As the training proceeds, the criterion for deciding whether to pick up the target object directly or to push the adjacent objects interfering with grasping is gradually set in the Q-network of the high-level selector.

The learning process is summarized in Algorithm 1. In this learning process, the parameters of the target network are updated by those of the Q-network every 200 steps. Since an episode is composed of a small number of steps, the discount factor is set to a relatively small value of 0.7.

3.5. Learning results

Learning in the virtual environment was conducted with 3,000 episodes. As expected, the robot has learned to directly grasp the target object when it does not need to arrange the surrounding objects, as shown in Fig. 6(a). By contrast, when the surrounding objects are regarded as obstacles, the robot first performs the pushing motion to remove them, then grasps the target object, as shown

Algorithm 1: Learning process.

```
Initialize Q-networks and target networks $\{Q_{\theta_h}, Q_{\theta_h'}, Q_{\theta_l}, Q_{\theta_l'}\}$
Initialize reply memories $\{D_h, Dl\}$
Initialize exploration probability $\{e_h, e_l\}$ to 1
for episodes do
  Initialize environment
  Get and process state $s$
  while not done do
    $g \leftarrow EGREEDY(s, e_h)$
    $s_0 \leftarrow s$
    while not (g is reached or failed) do
      $a \leftarrow EGREEDY(s, g, e_l)$
      Execute $a$ and obtain $r_l$
      Get and process next state $s'$
      Store $(s, g, a, r_l, s')$ in $D_l$
      $s \leftarrow s'$
      Update parameters $Q_{\theta_h}$ and $Q_{\theta_l}$ with random samples from $D_h$ and $D_l$
      if $step \% M = 0$ then
        $Q_{\theta_h'} \leftarrow Q_{\theta_h}, Q_{\theta_l'} \leftarrow Q_{\theta_l}$
      end if
    end while
    Obtain $r_h$
    Store $(s_0, g, r_h, s')$ in $D_h$
  end while
  Update $e_h$ and $e_l$
end for
```

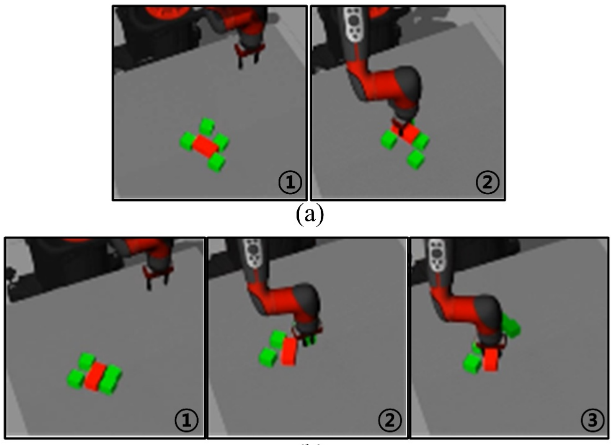

Fig. 6. Grasping task after learning: (a) in the case that the robot can directly grasp the target block and (b) in the case that the robot needs to remove the surrounding objects before grasping the target.

in Fig. 6(b). The tasks were executed for evaluation using the learning results discussed above, and the robot succeeded in grasping the target object 27 times out of 30 trials, achieving a grasping success rate of 90%. Note that a trial was considered a failure when the robot was caught by the object while approaching the ground to perform the selected motion. Furthermore, 68 meaningful motions were performed out of a total of 80 motions during the 30 grasping tasks, thereby achieving a motion efficiency of 85%. Note that a meaningful motion is a motion that is necessary for the robot to grasp the target object. For example, if the robot can grasp the target object despite the surrounding objects, the motion of grasping the target is considered to be a meaningful motion, but the motion of removing the surroundings is considered meaningless. Thus, the motion efficiency indicates how well the robot learned as intended.

In order to confirm the learning process, the robot was evaluated using the parameter sets of the Q-networks stored every 100 episodes during the learning. For each stored parameter set, the tasks was repeated 30 times to measure the success rate of grasping. Furthermore, to investigate the effect of the proposed method on the learning process, additional learning processes were performed under different conditions (cases 1 and 2), and the same evaluation processes were conducted. In case 1, the proposed preprocessing method was partially applied to the image (converted to the grayscale image and focused on the target object), and in case 2, the image was not preprocessed but just converted to grayscale.

Fig. 7 compares the grasping success rates achieved after the evaluation. As shown in the graph, during 3000 episodes, the success rate of grasping achieved with the parameter sets trained using the proposed method increases rapidly. However, in the results of case 1, the increase rate is significantly low, and the success rate only reaches about the half of the proposed method's results after 3000 episodes of training. In the results of case 2, the success rate hardly increases, which means that without the preprocessing of the image, the current structures of the Q-networks may take a very long time or may not be able to produce satisfactory results. This indicates that the proposed preprocessing method effectively enhances the learning speed and improves the learning efficiency.

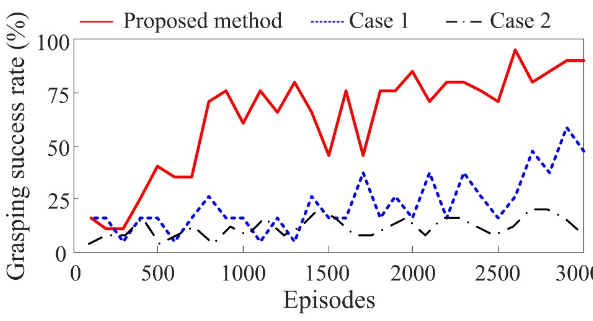

Fig. 7. Comparison of the learning results.

4. TRANSFER TO REAL WORLD

In a virtual environment, since the pose information of the object is provided by a simulator, learning can be performed without the need for object detection. However, in a real environment, in order to grasp the target object, it is necessary to first detect the target and acquire its pose information. Moreover, to apply the learning result earned in the virtual environment to the real environment without any additional learning, a method for minimizing the difference between the two environments is needed. In this section, the preprocessing method for reducing the difference is discussed, and the experimental results in the real environment are presented.

4.1. Image preprocessing

In this study, the Mask R-CNN algorithm [19] is used to perform the grasping tasks in the real environment since it offers both the position and shape information of the detected object. With the shape information, the image obtained from the real environment can be made to be similar to the state used in the virtual environment. Fig. 8 shows the process of converting the RGB image obtained from the camera into the input state for the Q-network. First, the Mask R-CNN algorithm receives the RGB image as input (as shown in the first row of Fig. 8) and creates the position and shape information of the objects as output (as shown in the second row of Fig. 8). Note that each object is segmented independently. After the color of the shape image of the target object is set to red and that of the surrounding objects is set to green, they are integrated into one image. Then, the color of the background is set to gray, which makes the image similar to the camera image obtained in the virtual environment, as shown in the third row of Fig. 8. Finally, as was the case in the preprocessing used in the learning stage, the image converted to grayscale is rotated so that the major axis of the target object is in the vertical direction, and only the image around the target object is taken for the input state. The major axis of the target is obtained by applying the principal component analysis (PCA) scheme [20] to the shape image.

The state obtained through this process is very similar to the state used in learning in the virtual environment, as can be confirmed by comparing Figs. 5 and 8. By preprocessing the images, the difference between the virtual and the real environments can be minimized, and the learning result in the virtual environment can be applied to the real environment without any additional learning. Furthermore, this method can be applied regardless of the background since the detailed textures of the surroundings except the objects are ignored and set to grayscale during the preprocessing

It takes about 0.2 s to calculate the Mask R-CNN algorithm and about 0.8 s for preprocessing using the PC with GTX-1080. Since this process is carried out before the robot performs the selected motion, the computation time does not affect the operation of the robot.

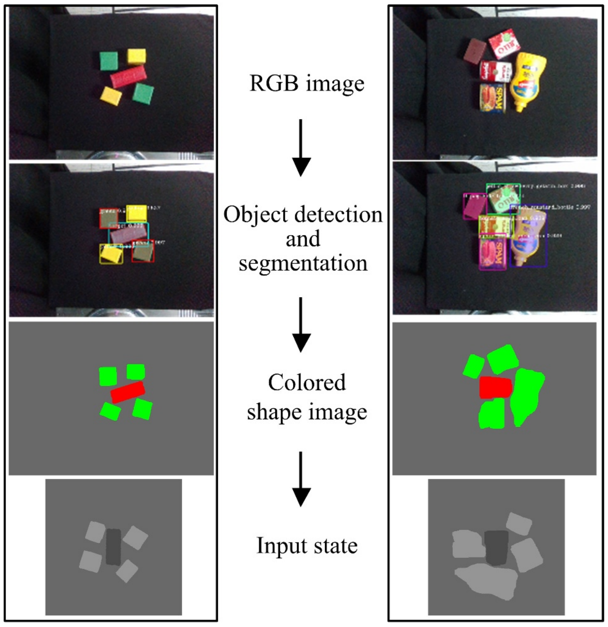

Fig. 8. Preprocessing of input state in the real world.

4.2. Experiments

The experiments were conducted in the real world using the learning results from the virtual environment without additional training. Since only an image, which is independent of the robot, was used as a state in learning, it is not necessary to use the same robot that was used in the learning stage. Therefore, in the experiments, the learning results in the virtual environment were applied to a 6 DOF robot developed in the laboratory. The Robotiq 2F-85 gripper and the Intel RealSense D-415 camera were also used in the experiments.

First, in order to verify the effectiveness of the proposed image preprocessing method, experiments were conducted using the blocks similar to those used in the virtual environment. During the experiments, the robot conducted 30 grasping tasks with the blocks closely placed, as shown in Fig. 9. A trial was regarded as a failure when the robot was blocked by the objects while approaching the ground to perform the selected motion. As a result of experiments, a grasping success rate of 86.7% and a motion efficiency of 82.1% were achieved, which were consistent with the results obtained in the virtual environment in Section 3.5. This result indicates that the proposed method can transfer the learning results directly to the real environment.

To confirm that the difference between the two environments is effectively reduced by the proposed method, the robot performed 30 additional grasping tasks using partially preprocessed images, in which the segmentation was

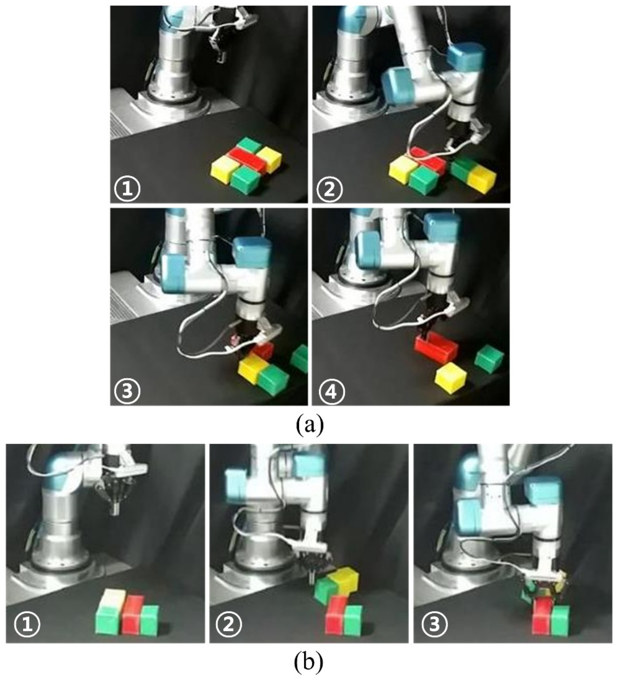

Fig. 9. Grasping procedure in cluttered environments when the target object is a red block: (a) in the case that the robot should clear both sides and (b) in the case that the robot can grasp the target object after clearing only the left side.

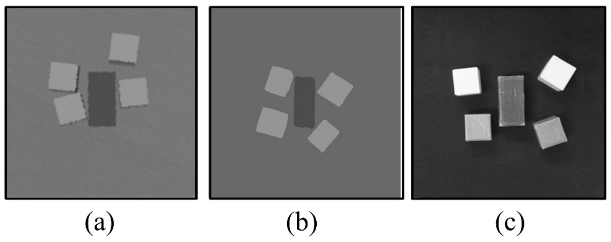

Fig. 10. Comparison of images used as states: (a) a state used during learning in the virtual environment, (b) a fully preprocessed state in the real environment, and (c) a partially preprocessed state in the real environment.

Table 1. Experimental results with simple objects.

Experimental conditions	Success rate of grasping	Motion efficiency
Virtual env. with state in Fig. 10(a)	90.0%	85.0%
Real env. with state in Fig. 10(b)	86.7%	82.1%
Real env. with state in Fig. 10(c)	13.3%	40.2%

not conducted, as states. Fig. 10 compares the states used in the learning process and experiments. Experiments using partially preprocessed images showed poor results of a grasping success rate of 13.3% and a motion efficiency of 40.2%. Since the robot, in many cases, repeated the pushing motion over and over, it was regarded as a failure when the robot could not grasp the target after carrying out 5 motions. Table 1 lists the evaluation results in the virtual and real environments. These results demonstrate that the proposed method successfully reduces the environmental difference between the simulator and the real world.

Further experiments were conducted with 20 objects in home environments that were not used in learning, as shown in Fig. 11, to confirm the generalization of the learning results. The robot performed 20 grasping tasks for each object with various combinations of surrounding objects. The average success rate of grasping was 81.8% and the average motion efficiency was 76.9%. Although the success rate and motion efficiency decreased slightly compared to the first experiment, the robot tended to perform the appropriate motions in each situation. Moreover, most failures occurred when the robot could not grasp the target object even after appropriately performing the pushing motions to arrange the surroundings. For example, the robot failed 8 times to grasp the SPAM due to the gripper size when the grasping pose was given diagonally as

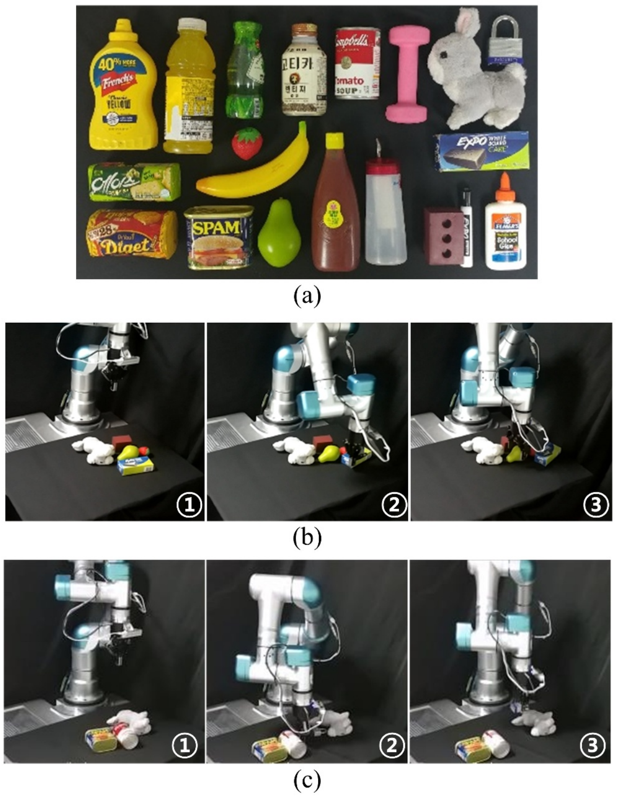

Fig. 11. Real-world experiments with various objects in home environments: (a) Objects used for experiments (b) grasping task when the target is a green plastic pear, and (c) grasping task when the target is a padlock.

a result of PCA. If these cases are excluded, the grasping success rate of 83.8% and the motion efficiency of 78.4% can be achieved. This indicates that the decision-making ability learned using simple blocks can be generalized and is thus applicable to various objects as well.

5. CONCLUSION AND FUTURE WORK

In this study, Q-learning based reinforcement learning, which has a hierarchical structure, is employed to learn the decision-making ability to grasp a target object in a cluttered environment. Learning was performed on simple objects in a virtual environment, and the learning results show a grasping success rate of 90% and 85% motion efficiency. The learning results were applied to an actual robot without any additional learning through the proposed preprocessing method, and a grasping success rate of 86.7% along with a motion efficiency of 82.1% were obtained as a result of the experiment using objects similar to those used in the virtual environment. Furthermore, it was confirmed that the robot properly determined the order of motions to use for grasping the target object, even when various objects were used. The experiment obtained a grasping success rate of 83.8% and a motion efficiency of 78.4%. These results show that the difference between the virtual and real environments can be overcome and that the learning results can be generalized by using the proposed method.

The grasping pose used in this study was fixed in the direction of the minor axis of the object. In such a grasping method, the robot could fail to grasp the target object even if the surrounding objects were well arranged. In order to overcome these limitations, we are planning to apply a grasping method which includes a way of estimating a more suitable grasping pose for objects of various shapes using depth information.

REFERENCES

[1] S. Caldera, A. Rassau, and D. Chai, "Review of deep learning methods in robotic grasp detection," *Multimodal Technologies and Interaction*, vol. 2, no. 3, Article number 57, 2018.

[2] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-driven grasp synthesis—a survey," *IEEE Transactions on Robotics*, vol. 30, no. 2, pp. 289-309, 2014.

[3] J. Weisz and P. K. Allen, "Pose error robust grasping from contact wrench space metrics," *Proc. of IEEE Int. Conf. on Robotics and Automation*, pp. 557-562, 2012.

[4] M. Gualtieri, A. Ten Pas, K. Saenko, and R. Platt, "High precision grasp pose detection in dense clutter," *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 598-605, 2016.

[5] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg, "Dex-Net 2.0: deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics," *Proc. of Robotics: Science and Systems*, 2017.

[6] S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *Int. Symposium on Experimental Robotics*, vol. 37, no. 4-5, pp. 421-436, 2017.

[7] J. Won, Y. Park, and I. H. Suh, "Scattering for robotic grasping in cluttered environments," *Int. Conf. on Ubiquitous Robots*, 2018.

[8] M. R. Dogar and S. S. Srinivasa, "A planning framework for non-prehensile manipulation under clutter and uncertainty," *Autonomous Robots*, vol. 33, no. 3, pp. 217-236, 2012.

[9] S. Amarjyoti, "Deep reinforcement learning for robotic manipulation-The state of the art," arXiv preprint arXiv:1701.08878, 2017.

[10] A. Zeng, S. Song, S. Welker, J. Lee, A. Rodriguez, and T. Funkhouser, "Learning synergies between pushing and grasping with self-supervised deep reinforcement learning," *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 4238-4245, 2018.

[11] R. Meyes, H. Tercan, S. Roggendorf, T. Thiele, C. Buscher, M. Obdenbusch, C. Brecher, S. Jeschke, and T. Meisen, "Motion planning for industrial robots using reinforcement learning," *Procedia CIRP*, vol. 63, pp. 107-112, 2017.

[12] R. Tang and H. Yuan, "Cyclic error correction based Q-learning for mobile robots navigation," *Int. Journal of Control, Automation, and Systems*, vol. 15, no. 4, pp. 1790-1798, 2017.

[13] L. Pinto and A. Gupta, "Supersizing self-supervision: Learning to grasp from 50K tries and 700 robot hours," *Proc. of IEEE Int. Conf. on Robotics and Automation*, pp. 3406-3413, 2016.

[14] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 23-30, 2017.

[15] S. James, A. J. Davison, and E. Johns, "Transferring end-to-end visuomotor control from simulation to real world for a multi-stage task," *Proc. of Conf. on Robot Learning*, 2017.

[16] K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige, S. Levine, and V. Vanhouke, "Using simulation and domain adaptation to improve efficiency of deep robotic grasping," *Proc. of IEEE Int. Conf. on Robotics and Automation*, pp. 4243-4250, 2018.

[17] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," arXiv preprint arXiv:1312.5602, 2013.

[18] T. D. Kulkarni, K. Narasimhan, A. Saeedi, and J. Tenenbaum, "Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation," *Advances in Neural Information Processing Systems*, pp. 3675-3683, 2016.

[19] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-CNN," *Proc. of IEEE Int. Conf. on Computer Vision*, pp. 2961-2969, 2017.

[20] Q. Lei, G. Chen, and M. Wisse, "Fast grasping of unknown objects using principal component analysis," *AIP Advances*, vol. 7, no. 9, 095126, 2017.

Kuk-Hyun Ahn received his B.S. degree in Mechanical Engineering from Korea University in Seoul, Korea, in 2014. He is currently pursuing a Ph.D. degree from the School of Mechanical Engineering, Korea University. His research interests include manipulator control, reinforcement learning, and AI-based robot applications.

Jae-Bok Song received his B.S. and M.S. degrees in Mechanical Engineering from Seoul National University in Seoul, Korea, in 1983 and 1985, respectively. He received his Ph.D. degree in Mechanical Engineering from M.I.T. in 1992. He is a professor in the Department of Mechanical Engineering in Korea University since 1993. His research interests include robot safety, manipulator design and control, and AI-based robot applications. He is a senior member of IEEE.