Proc. of the 9th Intl. Conf. of Asian Society for Precision Engg. and Nanotechnology (ASPEN 2022)
15–18 November 2022, Singapore. Edited by Nai Mui Ling Sharon and A. Senthil Kumar
doi://10.3850/978-981-18-6021-8_OR-02-0083

Digital Twin-based Object Detection of Camera Mounted Robot Arm using Deep Reinforcement Learning

Su-Young Park[1], Hyungjung Kim[2], Junghyuk Lee[1], Suhwan Jeong[1] and Sung-Hoon Ahn[1,2,#]

1 Department of Mechanical Engineering, Seoul National University, Seoul 08826, Republic of Korea
2 Institute of Advanced Machines and Design, Seoul National University, Seoul 08826, Republic of Korea
Corresponding Author / E-mail: ahnsh@snu.ac.kr, TEL: +82-2-880-9074, FAX: +82-2-888-9073

KEYWORDS: Digital twin, Robot arm, Object detection, Deep reinforcement learning, Simulation-to-reality

With the advancement of Industry 4.0, Digital Twin has been applied to robots used in various industries and is being used in research such as navigation, task automation, and collision prediction. The robot arm used in the traditional manufacturing industry is optimized to perform repetitive tasks in a pre-determined and fixed environment. In most cases, the robot arm moves to a fixed path or approaches and handles the target object using a vision system that estimates the location based on augmented reality marker. To detect a target object outside a specific range, recently, vision-based object detection and various related algorithms are being developed. However, there is no proper high-fidelity Digital Twin platform which can simulate the integrated robot automation system including a perception function. In addition, various evaluation of different locations of a target and obstacle is limited, and thus it is not possible to flexibly respond to changes in the environment. These limitations result in the decrease of reliability and simulation-to-reality transferability. This study proposes a robot arm object detection system using photo-realistic Digital Twin and deep reinforcement learning algorithm. A high-fidelity Digital Twin was reproduced on NVIDIA Omniverse platform, a state-of-the-art physics engine-based simulator. 3D models of the actual robot arm, camera, target object, and obstacle were reconstructed in Digital Twin. Virtual target images were created with NVIDIA Scene Imaging Interface, and it is learned with Deep Object Pose Estimation algorithm. Camera-mounted robot arm was also controlled by using reliable robot control package, ROS Moveit!. The target and obstacle were randomly generated on the working region by using domain randomization for each episode. The learned policy using deep reinforcement learning in Digital Twin were seamlessly deployed and evaluated in the actual robot system through ROS-based framework. This integrated system could robustly detect the target object which has various positions and obstacle. In the future, we plan to extend the system developed in this study to a mobile robot arm to explore and handle objects in more diverse environments. We hope that this study can increase the reliability of robot automation system and decrease the time of programming and developing robots.

NOMENCLATURE

AI = Artificial Intelligent
IoT = Internet of Things
DT = Digital Twin
DRL = Deep Reinforcement Learning
ROS = Robot Operating System
FOV = Field of View
DOPE = Deep Object Pose Estimation
CNN = Convolution Neural Network
PPO = Proximal Policy Optimization

1. Introduction

With recent advancement of Industry 4.0, new technologies are being applied across various industries. Especially, Smart Factory that incorporate numerous technologies such as AI, big data, and IoT are being developed and evaluated within the fields of manufacturing [1].

DT is also being applied to robots, which are major equipment in Smart Factory [2]. DT is a technology that implements a virtual system in which it is similar to the actual system into a virtual world. The performance and the efficiency of the actual system can be improved by feeding back the optimized results. DT-based robot automation can improve overall productivity by preventing future errors in advance and maximize operational efficiency [3, 4]. In particular, the reliability of vision-based perception must be secured in a smart factory where production conditions and target change frequently.

However, there is no proper high-fidelity DT platform that can simulate the integrated robot automation system. Especially, evaluation of the perception system under various conditions is limited, and then it is hard to flexibly respond to unexpected conditions. These result in a decrease of simulation-to-reality transferability and reliability. Furthermore, the pause of the operating robots for the deployment of the new program into the robots cause downtime and reduces productivity [5].

This study proposes a DT-based seamless framework for flexible and reliable perception of camera-mounted robot arm using DRL. Photo-realistic DT capable of finding a target outside the camera FOV is developed by training with DRL. Finally, DRL-based target detection system is seamlessly deployed to the actual robot system based on the ROS-based control framework.

2. System configuration

A powerful RTX-based GPU is required for perception-based DRL in a photo-realistic environment. Therefore, in this study, Nvidia Omniverse platform and Isaac Sim robot development tool which is optimized based on Nvidia RTX GPU were used for DT construction [6]. In addition, the object detection algorithm and robot control algorithm were configured as ROS nodes to build a stable and reliable system.

2.1 Framework

A framework was built by using ROS, which is currently used for most robot development and provides a stable and reliable system control package (Fig. 1). The virtual robot, camera, and work environment were virtualized in DT, and the object detection algorithm and robot control algorithm (Moveit!) were implemented in the external ROS environment [7]. The policy can be learned based on the data processed through ROS by linking DRL and DT. The learned policy can be seamlessly deployed to the actual hardware system through ROS-based control software.

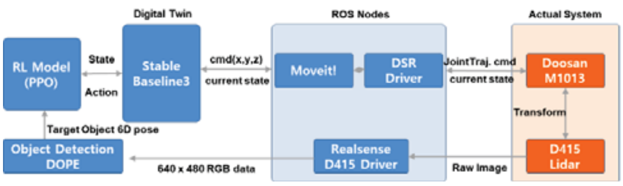

Fig. 1 Framework of the software modules for a camera-mounted robot arm and DRL-based object detection

2.2 Target Detection Pipeline

NVISII and DOPE algorithms were used to build the target detection pipeline. (Fig. 2). The DOPE algorithm trains the target object's belief map and vector field CNN and estimates the target's 6d pose [8]. Several domain-randomized images were generated using NVISII, a virtual image generation library, and these were used as DOPE training data [9]. This pipeline can train robust object detection algorithms in various environments with only a 3D modeling file.

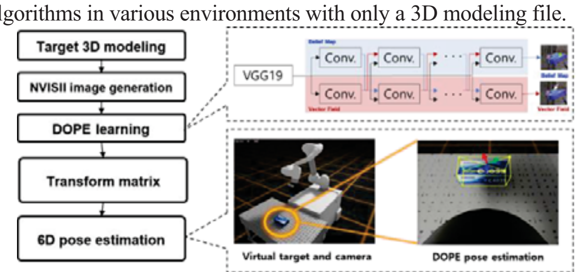

Fig. 2 DOPE-based object detection pipeline

2.3 DRL algorithm

DRL is a machine learning method that informs an agent of an action that can be taken and can learn by itself based on a current state and a reward from the action. DRL creates data through interaction between environment and agent, and uses it for neural network learning.
In this study, the PPO algorithm, which provides a clipping parameter ($\boldsymbol{\varepsilon}$), enables more stable and reliable learning for agents that take continuous actions [10]. The end-effector of the camera-mounted robot arm was set as an agent, and the movement per frame was set as $\mathrm{action}(a(\delta x, \delta y, \delta z), 0 \sim \pm 167\ \mathrm{mm/frame.})$ Observation was set as the absolute coordinates of the agent ($\mathrm{obs}(x, y, z)$), and reward equation (Eq. 1) according to the distance between the agent and the target and the success of target detection was defined. Hyper-parameters were also set (Table 1).

$$R = \boldsymbol{w}_{distance} * \boldsymbol{r}_{distance} + \boldsymbol{w}_{DOPE\ success} * \boldsymbol{r}_{DOPE\ success} - \boldsymbol{r}_{time\ penalty} \cdots (Eq. 1)$$

Table 1 Hyperparameters for PPO

Hyperparameter	Value
Graphic card	Nvidia RTX 2080 Ti
Learning model	Proximal Policy Optimization
Total time-step	180K
Episode length	3000
Batch size	3000
Clipping parameter $\boldsymbol{\varepsilon}$	3000
Discount factor $\boldsymbol{\gamma}$	0.9995
Learning rate	0.00025

3. Evaluations

We implemented the actual robot system in DT. The same model was virtualized in DT by reflecting the specification (Table 2) of a camera-mounted robot arm (Fig.3) equipped with a camera (Realsense D415) on the robot arm (Doosan M1013).

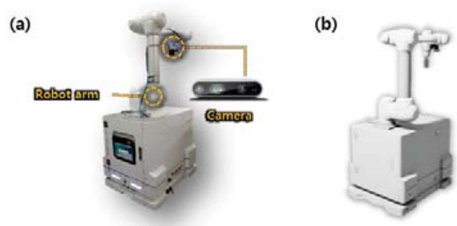

Fig. 3 Realization of the actual camera-mounted robot system into DT; (a) real robot (Doosan M1013+Realsense D415), (b) DT robot

Table 2 Main specifications of the robot arm and camera

Hardware	Main spec	Value
Robot arm (Doosan M1013)	Axis	6
	Max. range	1300 mm
	TCP Speed	1m/s
	Repeatability	± 0.1mm
Camera (Realsense D415)	Frame resolution	1920 x 1080
	Frame rate	30 fps

	FOV	69° x 42°
	Resolution	2 MP

3.1 Configuration of experimental environment

An environment to automate target detection of the camera-mounted robot arm was implemented (Fig. 4).

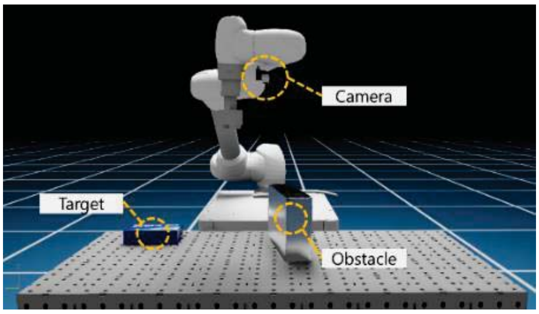

Fig. 4 Experimental setup for DLR-based object detection in DT

The target is randomly only generated to the left (type1) and right (type2) side of the initial FOV of the initial camera, and an obstacle is created around it to cover the target (Fig. 5).

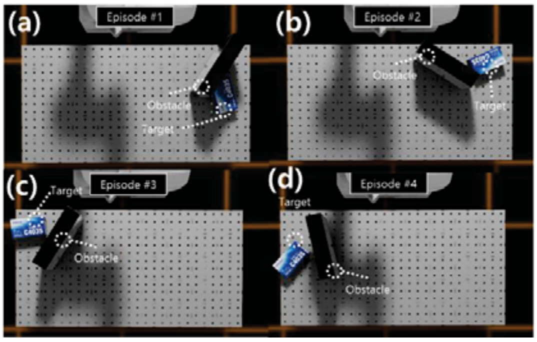

Fig. 5 Random generation of the target and obstacle in DT for the following scenarios, (a), (b) target and obstacle are generated on the right side (type1), and (c), (d) generated on the left side (type2).

3.2 Evaluation of DOPE-based target detection

DOPE-based target detection performance was evaluated for reliable perception system. The 6D pose of the randomly generated target was estimated with the DOPE, and the errors of the x, y, and z axes were visualized (Fig. 6). DOPE stably estimated the poses with an average error (n=300) of less than 5 mm in all axes (Table 3). Although the maximum error was 38.74 mm, it was judged that the system was sufficiently stable to determine the success of target detection.

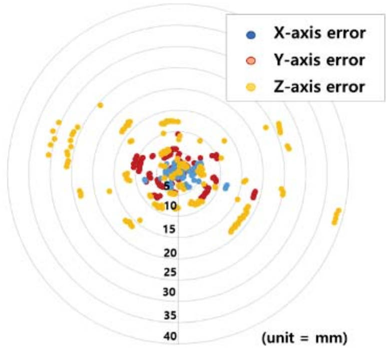

Fig. 6 Error distributions of DOPE, (a) Radial graph of errors between x, y, z actual positions and DOPE results

Table 3 Validation results of DOPE

Error	**X-axis**	**Y-axis**	**Z-axis**
Average	4.24 mm	1.98 mm	3.72 mm
Maximum	18.62 mm	11.75 mm	38.74 mm

4. Results and Discussion

Target detection policy was trained based on DRL with our framework. In the early stage of learning, the agent (=end effector) searched the surroundings non-directionally (Fig. 7).

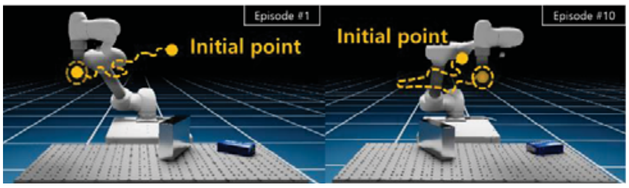

Fig. 7 Random and non-directional robot arm movement at the beginning of the learning

During the learning procedure, the agent made various attempts and learned the policy in the direction of increasing reward (Fig. 8).

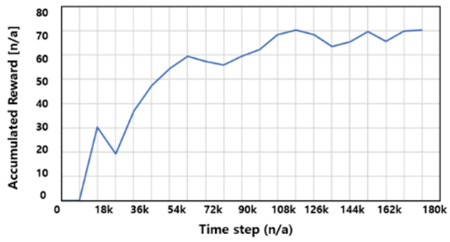

Fig. 8 Accumulated reward trend according to time step

After learning, DRL-based target detection system was validated in DT (Fig. 9). As a result, the agent searched point1 and point2 in order.

DRL-based object detection system was also validated in the actual system (Fig. 10). Learned policy by DRL is deployed from DT to actual system seamlessly using our framework. In the actual system, the same as DT, point1 and point2 were always searched in order.

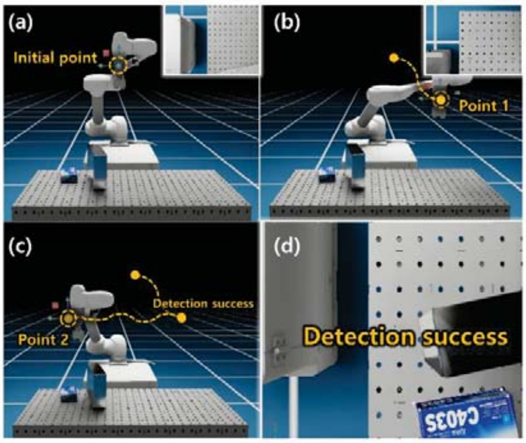

Fig. 9 DRL-based robot arm trajectory for target detection in DT for the type 2, (a) End effector located at initial point, (b) searching near the point 1 (c) searching near the point 2 (d) target detection successful

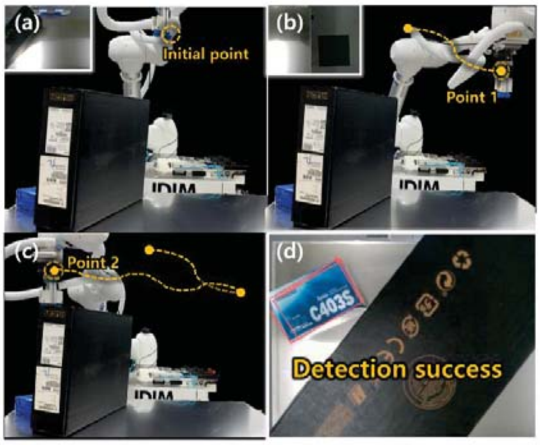

Fig. 10 Evaluation of actual robot system in case of type 2, (a) End effector located at initial point, (b) searching near the point 1 (c) searching near the point 2 (d) target detection successful

As a result of repeated experiments, stable target detection was possible in both DT and actual systems through relatively short reinforcement learning (Table 4).

Table 4 DLR-based target detection results in DT and the actual system

	DT	Actual system
Time for teaching	1 hours	-
Success rate (Type 1,2)	100 % (n=50)	100 % (n=10)
Time for detection (Average)	1.9 s (Type 1) 16.1 s (Type 2)	2.3 s (Type 1) 18.2 s (Type 2)

We predicted that a target under the type 1 condition was continuously and repeatedly generated during DRL, and the agent received a high reward in that episode and converged to this result.

5. Conclusions

In this study, we implemented DT to automate the target detection of a camera-mounted robot arm and the robot were able to successfully detect randomly generated targets using DRL. This DT was able to train the target detecting policy and performed in the actual system with seamless deployment. This DT works fully offline, so the learned policy can be deployed immediately without interruption of the actual robot in operation. In addition, it is possible to learn and evaluate a reliable vision-based perception system through photo-realistic rendering. In the future, we plan to evaluate more diverse and complex tasks by adding obstacle avoidance and target handling function.

ACKNOWLEDGEMENT

We would like to acknowledge the financial support from K-CLOUD project (No. 2020-Tech-06) of KOREA HYDRO & NUCLEAR POWER CO., LTD (KHNP) of Republic of Korea.

REFERENCES

1. Kim, H., Jung, W. K., Park, Y. C., Lee, J. W., & Ahn, S. H., "Broken stitch detection method for sewing operation using CNN feature map and image-processing techniques," *Expert Systems with Applications*, *188*, 116014, 2021
2. Moon, D. H., Cho, H. I., Kim, H. S., Sunwoo, H., & Jung, J. Y., "A case study of the body shop design in an automotive factory using 3D simulation," *International Journal of Production Research*, *44*(18-19), 4121-4135, 2006
3. Matulis, M., & Harvey, C., "A robot arm digital twin utilising reinforcement learning," *Computers & Graphics*, *95*, 106-114, 2021
4. Pedersen, M. R., Nalpantidis, L., Andersen, R. S., Schou, C., Bøgh, S., Krüger, V., & Madsen, O., "Robot skills for manufacturing: From concept to industrial deployment," *Robotics and Computer-Integrated Manufacturing*, *37*, 282-291, 2016
5. Lyu, H., "Industrial robot offline programming system development and its applications," *Ph.D. Thesis*, Hong Kong University of Science and Technology, 2019
6. Hummel, M., & Kooten, K. V., "Leveraging nvidia omniverse for in situ visualization," In *International Conference on High Performance Computing* (pp. 634-642), 2019
7. David Coleman, Ioan A. Şucan, Sachin Chitta, Nikolaus Correll, "Reducing the Barrier to Entry of Complex Robotic Software: a MoveIt! Case Study," *Journal of Software Engineering for Robotics*, 5(1):3–16, 2014
8. Tremblay, J., To, T., Sundaralingam, B., Xiang, Y., Fox, D., & Birchfield, S., "Deep object pose estimation for semantic robotic grasping of household objects," *arXiv preprint arXiv:1809.10790*, 2018
9. Morrical, N., Tremblay, J., Lin, Y., Tyree, S., Birchfield, S., Pascucci, V., & Wald, I., "NViSII: A scriptable tool for photorealistic image generation," *arXiv preprint arXiv:2105.13962*, 2021
10. Yu, C., Velu, A., Vinitsky, E., Wang, Y., Bayen, A., & Wu, Y., "The surprising effectiveness of ppo in cooperative, multi-agent games," *arXiv preprint arXiv:2103.01955*, 2021