An Effective Method for Minimizing Domain Gap in Sim2Real Object Recognition Using Domain Randomization

Tomohiro Ono*
Kyushu Institute of Technology,
2-4 Hibikino, Wakamatsu-ku, Kitakyushu, Fukuoka, 808-0196, Japan †

Akihiro Suzuki
Kyushu Institute of Technology,
2-4 Hibikino, Wakamatsu-ku, Kitakyushu, Fukuoka, 808-0196, Japan

Hakaru Tamukoh
Kyushu Institute of Technology,
2-4 Hibikino, Wakamatsu-ku, Kitakyushu, Fukuoka, 808-0196, Japan
E-mail: ono.tomohiro342@mail.kyutech.jp, suzuki@brain.kyutech.ac.jp, tamukoh@brain.kyutech.ac.jp
http://www.lsse.kyutech.ac.jp/english/

Abstract

Manual annotation is common, but problems occur, such as oversight and mislabeling via human error. These problems are known to affect the quality of datasets significantly. To resolve these problems, we propose a method to automatically generate high-quality and large datasets in a short time using a simulator. Our proposed method uses domain randomization to minimize domain gaps without faithfully reproducing real scenes. The generated dataset achieved more than 80% recognition accuracy against the real image dataset.

Keywords: Data-centric Deep Learning, Object Recognition, Sim2Real, Dataset Generation.

1. Introduction

With the advent of deep learning, various technologies have been realized. In particular, the impact of deep learning on object recognition has been significant, and various models and large datasets have been released for validation. Deep learning performance is determined by $Code(model, algorithm) * Data$. There are two approaches to improving this performance: model-centric, which aims to improve performance based on the model, and data-centric, which seeks to improve performance based on data. Many studies have been conducted on improving the performance of these models. However, the reality is that the performance of models has reached a plateau and has yet to further improve dramatically. Therefore, recently, the data-centric approach, which improves the quality of datasets, has attracted research attention[1].

Figure 1 shows the four essential elements that define the quality of the dataset. The first is consistent labeling (Fig. 1(a)). For example, consider the annotation of the image of two elephants shown in Fig. 2 Suppose the annotator is instructed to "enclose the elephants with bounding boxes (BBs)." At this time, there are three ways to annotate the elephants: annotate them with a single BB, as in Fig. 2(a), annotate the visible part of each elephant, as in Fig. 2(b), or annotate only the non-overlapping parts of each elephant, as in Fig. 2(c). Since

*.

© The 2023 International Conference on Artificial Life and Robotics (ICAROB2023), Feb. 9-12, on line, Oita, Japan

the optimal solution for these ways depends on the situation, it is necessary to establish clear rules in advance and annotate according to the rules. However, such rules are often very complex and vary widely. The second is the accuracy (Fig. 1(b)). Accuracy refers to the ability to annotate appropriately. Annotation is simple and may be affected by the presence or absence of a concentration or time margin, even for the same annotator. The third and fourth phases (Fig. 1(c) and (d)) contain missing labels and images that are mislabeled, respectively. As with accuracy, these are also caused by human error and can significantly degrade the quality of the dataset. In general, double-checking, or other approaches is required to ensure quality. The dataset quantity is also essential; generally, the larger the training dataset, the higher the accuracy. However, annotation costs increase proportionally to the volume of data, and the more people involved, the greater the risk of information leakage.

These problems are unavoidable if people are involved. In recent years, considerable research has been conducted on automatic dataset generation using 2D image synthesis[2][3] and 3D simulation[4][5][6]. These methods do not require manual annotation, and the simulation renderer can quickly obtain synthetic images and high-quality ground-truth annotations. Thus, high-quality datasets can be generated without human error. However, the distribution of the generated synthetic images differs from that of real images. This is the so-called domain gap. Domain randomization[7] and domain adaptation[8] were proposed to fill this domain gap in Sim2Real studies. In recent years, the appearance of neural radiance fields (NeRF)[9] and others have enabled the easy acquisition of high-quality 3D models. In this study, we have focused on regions other than objects for recognition, E.g., the background. We propose an effective method to minimize the domain gap using domain randomization without reproducing real scenes. Particularly, we propose a method to generate synthetic data that encompasses the distribution of real images by complexly changing various elements in the simulator (e.g., camera positions, angles, and background textures). We conclude by effectively using complex and diverse fractal images as background textures in Sim2Real. To improve the quality of the dataset, we propose a method to remove the annotations of objects that are missing in the image and have significant feature

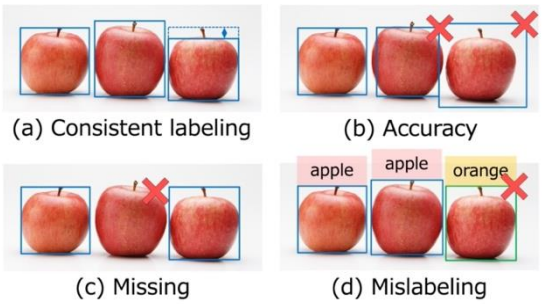

Fig. 1. Quality of dataset.

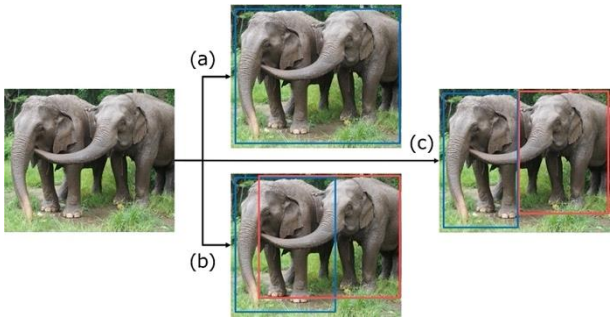

Fig. 2. Example of inconsistent labeling: when instructed to enclose the elephants in bounding boxes.

deficits. We trained the synthetic data generated by the proposed method on You Only Look Once v7 (YOLOv7)[10] and experiments for real images.

2. Proposed Method

We propose a dataset generation method using the real-time physics simulator PyBullet[11].

Figure 3 presents an overview of the proposed method. The procedure is as follows.

1. Acquire 3D models
   First, a 3D model of the object to be recognized is acquired. A high-quality 3D model can be easily obtained using NeRF, a 3D scanner, or a smartphone application. 3D modeling software can also be used to create the 3D model.
2. Initialize the simulator
   Set up the simulator environment for the assumed scene. For example, we set up simple furniture such as a desk, chair, and shelves, because we assumed that the simulator would be used in a home environment. Changing the scene according to the assumed environment enabled generation of a high-quality dataset. Each piece of furniture was

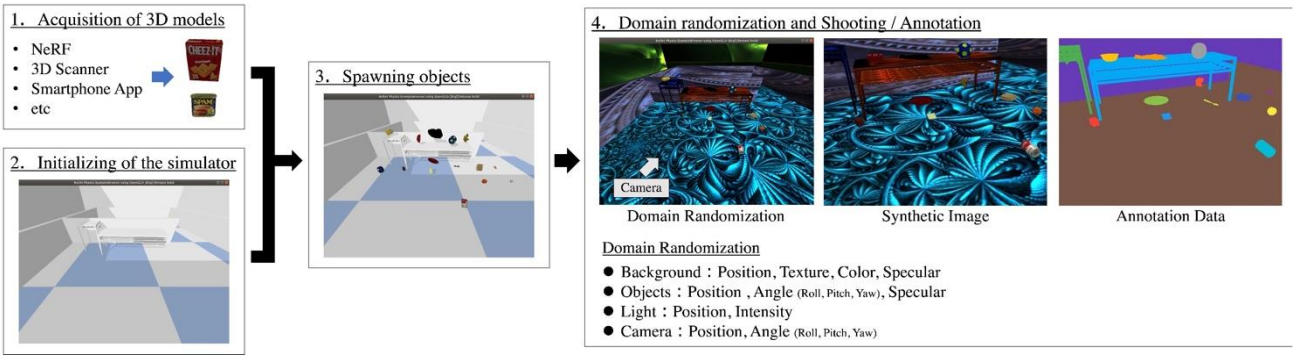

Fig. 3. Overview of the proposed method.

randomly rearranged after a certain number of data acquisitions.

3. Spawn objects

The 3D model created in Step 1 is generated in the simulator environment set up in Step 2. At this time, the position and orientation of the objects are randomly determined. In addition, the objects are generated in advance at a higher position than the ground and placed on the ground using physical operations to make the object placement look real. After spawning objects, the physics operations are temporarily disabled to speed up the subsequent dataset generation process.

4. Randomize the domain, shoot, and annotate

By simulating the environment in a complex way, we improved the dataset's quality. Here, the background conditions (including furniture), objects, light source, and camera were randomly changed, and synthetic image and annotation data were generated. These data were generated by using the rendering function of the simulator. The annotation data was outputted according to the Object Detection in COCO format [12], and used for object detection, semantic segmentation, panoptic segmentation, and instance segmentation. In this case, we used monochromatic, Perlin noise and fractal images for the background textures. Our method was characterized by the fact that it did not use realistic textures. We applied this to domain randomization, and from ten to one hundred different synthetic images and annotations were generated per scene. Then, we returned to Step 2 to generate a predetermined number of synthetic images and annotations.

Furthermore, we can quickly generate data by running the simulator in parallel. Using an Intel i9-12900 K CPU (16 core, 24 threads, 3.20 GHz) and a GeForce RTX 3090 GPU, approximately 100,000 images could be processed in one hour when run ten in parallel.

3. Experiments

The synthetic dataset generated by the proposed method was trained on YOLOv7, a real-time object detection model, and verification was conducted to see if real images could be recognized.

We use 56 classes from the YCB object and model sets[13][14] published as benchmark objects as training classes. We manually annotated 50 real images as validation data. We generated three datasets, 10,000, 100,000, and 500,000 images. The human-annotated dataset was prepared for comparison with the proposed synthetic and real data in terms of dataset performance. The dataset contained 7,093 images clipped from 18 videos at 15 frames per second (fps), and the videos were filmed at 30 fps.

These datasets were used to train YOLOv7 in 32 batch sizes and 20 epochs, and COCO metrics calculate the mean average precision (mAP), which is used to compare dataset performance.

3.1. *Result*

The experimental results in Table 1 show that the mAP score improves with the number of images. Furthermore, the proposed method achieves higher accuracy than the real-world environment dataset, indicating its effectiveness.

Because of these considerations, a domain randomization technique was required to realize Sim2Real object detection. In particular, complex, and

Table. 1. Results of recognition accuracy.

Dataset	10,000	100,000	500,000	Real
mAP_{50}	0.586	0.823	**0.841**	0.667
mAP	0.471	**0.675**	**0.675**	0.528

diverse textures, such as fractal images, assisted in the Sim2Real effect without preparing a texture of a real scene.

Compared with the lead time to prepare both images and annotations, it took five annotators almost two weeks to obtain approximately 7,000 real images. In contrast, it took only one hour to get 100,000 synthetic images and annotations data using the proposed method.

This comparison and the accuracy above comparisons indicate that the proposed method excels in accuracy and reduces the time cost.

4. Conclusions

This study proposes dataset generation using a 3D simulator and domain randomization for Sim2Real. The experimental results showed that accuracy was higher when trained by a synthetic dataset than when trained by a real image dataset with human annotation. The proposed method of applying a fractal image as a texture in the simulator realized Sim2Real object detection. However, the performance in Sim2Real is still about 84%, and this is open to further study.

Future work will consider dataset generation and model updating as one step in a deep learning training procedure and will be designed to improve the deep learning in a data-centric manner.

Acknowledgements

JSPS KAKENHI (grant number:20J23242) supported this work. This study is based on results obtained from a project, JPNP16007, commissioned by the New Energy and Industrial Technology Development Organization (NEDO).

References

1. “Data-centric approach vs model-centric approach”, https://www.linkedin.com/pulse/data-centric-approach-vs-model-centric-steve-nouri/?trk=public_post
2. Y. Ishida, H. Tamukoh, “Semi-Automatic Dataset Generation for Object Detection and Recognition and its Evaluation on Domestic Service Robots,” Journal of Robotics and Mechatronics, Vol. 32, No. 1, pp. 245-253, 2020.
3. Y. Abe, Y. Ishida, T. Ono, H. Tamukoh, “Acceleration of training dataset generation by 3D scanning of objects,” The 2020 International Conference on Artificial Life and Robotics (ICAROB2020), OS20-4, Oita, Japan, January 13-16 (14), 2020.
4. M. Denninger, M. Sundermeyer, D. Winkelbauer, Y. Zidan, D. Oler, M. Elbadrawy, A. Lodhi, H. Katam, ″Blenderproc: Reducing the reality gap with photorealistic rendering″, In International Conference on Robotics: Sciene and Systems (RSS), 2020.
5. S. Max, B. Sven, ″Stillleben: Realistic scene synthesis for deep learning in robotics″, In IEEE International Conference on Robotics and Automation (ICRA), 2020.
6. G. Klaus, B. Francois, B. Lucas, D. Carl, D. Yilun, D. Daniel, F. David, J. G. Dan, G. Florian, H. Charles, K. Thomas, K. Abhijit, L. Dmitry, L. Issam, L. Hsueh-Ti, M. Henning, M. Yishu, N. Derek, O. Cengiz, P. Etienne, R. Noha, R. Daniel, S. Sara, S. M. S. Mehdi, S. Matan, S. Vincent, S. Austin, S. Deqing, V. Suhani, W. Ziyu, W. Tianhao, M. Yi Kwang, Z. Fangcheng, T. Andrea, “Kubric: a scalable dataset generator”, In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2022.
7. J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, P. Abbeel, ″Domain randomization for transferring deep neural networks from simulation to the real world″, In 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 23-30, 2017.
8. B. Imbusch, M. Schwarz, S. Behnke, “Synthetic-to-real domain adaptation using contrastive unpaired translation”, In Proceedings of 18th IEEE International Conference on Automation Science and Engineering (CASE), 2022.
9. B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, R. Ng, “Nerf: Representing scenes as neural radiance fields for view synthesis”, In European Conference on Computer Vision (ECCV), 2020.
10. C. Wang, A. Bochkovskiy, H. M. Liao, “YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors”, arXiv preprint arXiv:2207.02696, 2022.
11. “Pybullet, a python module for physics simulation for games, robotics and machine learning”, http://pybullet.org/
12. T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Doll'ar, C. L. Zitnick, “Microsoft COCO: common objects in context”, Computing Research Repository (CoRR), abs/1405.0312, 2014.
13. B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel and A. M. Dollar, ″Benchmarking in Manipulation Research: Using the Yale-CMU-Berkeley Object and Model Set″, in IEEE Robotics & Automation Magazine, vol. 22, no. 3, pp. 36-52, Sept. 2015.
14. B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel and A. M. Dollar, ″The YCB object and Model set: Towards common benchmarks for manipulation research″, International Conference on Advanced Robotics (ICAR), Istanbul, 2015, pp. 510-517.

Authors Introduction

Mr. Tomohiro Ono

He received the B.Eng. degree from National Institute of Technology, Ube College, Japan, in 2018. He received the M.Eng. from Kyushu Institute of Technology, Japan, in 2020. He is currently in a Ph.D. student in the graduate school of Life Science and Systems Engineering, Kyushu Institute of Technology. Since 2020, he has also been a research fellow of the Japan Society for the Promotion of Science (JSPS). His research interest includes image processing, motion planning and domestic service robots. He is a student member of RSJ.

Mr. Akihiro Suzuki

He obtained B.E. degree in electrical and electronic engineering from Tokyo University of Agriculture and Technology, Tokyo, Japan, in 2014. He has begun working under Tamukoh laboratory in Kyushu Institute of Technology, Fukuoka, Japan. He obtained a M.E. and a Ph.D degree in the Kyushu Institute of Technology, in 2016 and in 2019, respectively. He has been a postdoctoral research fellow in Tamukoh laboratory. His research interests include deep neural networks in computer vision domain.

Prof. Hakaru Tamukoh

He received the B.Eng. degree from Miyazaki University, Japan, in 2001. He received the M.Eng and the Ph.D. degree from Kyushu Institute of Technology, Japan, in 2003 and 2006, respectively. He was a postdoctoral research fellow of 21st century center of excellent program at Kyushu Institute of Technology, from April 2006 to September 2007. He was an assistant professor of Tokyo University of Agriculture and Technology, from October 2007 to January 2013. He is currently an associate professor in the graduate school of Life Science and System Engineering, Kyushu Institute of Technology, Japan. His research interest includes hardware/software complex system, digital hardware design, neural networks, soft-computing and home service robots. He is a member of IEICE, SOFT, JNNS, IEEE, JSAI and RSJ.