Application of Synthetic Data on Object Detection Tasks

Huu Long Nguyen
School of Mechanical Engineering, Hanoi University of Science and Technology, Vietnam
long.nguyenhuu@hust.edu.vn

Duc Toan Le
School of Mechanical Engineering, Hanoi University of Science and Technology, Vietnam
toan.ld231033M@sis.hust.edu.vn

Hong Hai Hoang
School of Mechanical Engineering, Hanoi University of Science and Technology, Vietnam
hai.hoanghong@hust.edu.vn (corresponding author)

Received: 24 May 2024 | Revised: 5 June 2024 | Accepted: 12 June 2024

Licensed under a CC-BY 4.0 license | Copyright (c) by the authors | DOI: https://doi.org/10.48084/etasr.7929

ABSTRACT

Object detection is a computer vision task that identifies and locates one or more effective targets from image or video data. The accuracy of object detection heavily depends on the size and the diversity of the utilized dataset. However, preparing and labeling an adequate dataset to guarantee a high level of reliability can be time-consuming and labor-intensive, because the process of building data requires manually setting up the environment and capturing the dataset while keeping its variety in scenarios. There have been several efforts on object detection that take a long time to prepare the input data for training the models. To deal with this problem, synthetic data have emerged as a potential for the replacement of real-world data in data preparation for model training. In this paper, we provide a technique that can generate an enormous synthetic dataset with little human labor. Concretely, we have simulated the environment by applying the pyBullet library and capturing various types of input images. In order to examine its performance on the training model, we integrated a YOLOv5 object detection model to investigate the dataset. The output of the conducted model was deployed in a simulation robot system to examine its potential. YOLOv5 can reach a high accuracy of object detection at 93.1% mAP when solely training on our generated data. Our research provides a novelistic method to facilitate the understanding of the data generation process in preparing datasets for deep learning models.

Keywords-object detection; YOLO; synthetic data

I. INTRODUCTION

Computer vision works much the same as human vision [1]. However, human vision has the advantage of a lifetime of context to train how to distinguish things, determine the distance between one object to another, recognize the motion status of the subject, and determine whether an image is incorrect. Instead of using retinas, optic nerves or a visual brain, computers educate machines to accomplish these functions in a timely manner by using cameras, data, and algorithms. A trained system in product examination or product asset monitor can outperform human skills since it can check hundreds of products or processes within a few minutes [2].

In the realm of synthetic data generation, Generative Adversarial Networks (GANs) [3] utilize two competing neural networks—a generator and a discriminator—to produce data that closely mimic genuine data, though they face issues with strange data points, specific data type modifications, hyper parameter determination, and unstable loss convergence. In contrast, Bayesian Networks [4] generate synthetic data without the loss issues of GANs but require time-consuming procedures to obtain and compare independent probability distributions of individual properties, making them less suited for complex data formats like images. For object detection, classical methods such as spatial histogram features [5] and SIFT-based clustering [6] offer solutions with inherent limitations: the former struggles with obscured images, while the latter's mean shift clustering becomes computationally intensive with increasing object numbers. The RCNN technique [7] addresses these challenges by using high-capacity Convolutional Neural Networks (CNNs) to extract feature vectors, combining conventional detectors with pre-train deep ConvNets, though it still relies on class-specific linear SVMs for classification.

In this paper, we discuss one of computer primitive vision's tasks, object detection, as a demonstration of further complex missions. Object detection is a crucial computer vision job that deals with detecting instances of certain classes of visual things (such as people, animals, or automobiles) in digital images [8]. A quality object detection task means that it can classify, localize, and detect different objects in a single image. In order to achieve the requirement, the amount and variety of the used datasets have a significant impact on how accurate the item detection is. It depends on the user to prepare a large dataset and label each object in the data. However, this task can be time consuming and human effort is needed to complete the preparation for the training model. For instance, authors in [9] trained a robotic agent through 580,000 real-world grab efforts to implement 96% grip success on previously unseen items. This achievement required seven robots and several weeks to complete.

In order to address this issue, synthetic data have shown promise as a viable alternative for actual data when preparing data for model training activities. Furthermore, the utilized dataset should satisfy the 4V characteristics [10] that are Volume, Velocity, Variety, and Value. The first characteristic represents the required volume of the data, the second means that data are being formed at unprecedented speed and must be deal within a timely manner. The third expresses that big data contain all kinds of data types and the last one indicates the low-value density. In this research, we present a method for quickly and easily producing a sizable synthetic dataset. The pyBullet library was used to mimic the environment, and several input picture kinds were collected. We used the successful object detection model YOLOv5 to study the dataset and evaluate its performance on the training model. The output of the model was used in a robot simulation system to test the viability of the hypothesis. Our research offers a cutting-edge technique to make it easier to comprehend the data production process while assembling a dataset for a deep learning model.

II. METHODOLOGY

Figure 1 illustrates the procedure for synthetic data collection using the pyBullet library, divided into two main blocks: Setup Parameter and Collecting Data. In the setup parameter block, planes and objects are deployed into the environment under the URDF, SDF, or MJCF files, which are supported by the pyBullet library. Our research utilizes objects from the YCB dataset [11], which are collected by two state-of-the-art systems: UC Berkley's scanning rig and the Google scanner. Positioning objects involves specifying x, y, and z coordinates, while orientations are set using quaternions, a four-dimensional representation of 3D rotations. pyBullet' s getQuaternionFromEuler function simplifies this process by converting intuitive rotations about the x, y, and z axes into quaternions. Additionally, key camera parameters—location, direction, field of view, and aspect ratio—are configured to ensure high-quality data capture. This thorough setup is crucial for generating realistic and diverse synthetic datasets, essential for training robust object detection models, and highlights the sophistication and potential of our simulation framework in advancing computer vision and robotics applications.

Figure 2 highlights the characteristics of synthetic data, showcasing variations in viewpoint and environment conditions during image capture. As mentioned above, the strategy of the experiment involves changing both the viewpoint and the light of the surroundings. In each image, the color parameters (red index, green index, or blue index) are adjusted orderly to simulate shifts in ambient lighting. Moreover, varying viewpoints allow for clearer observation of objects as the camera moves around them. Beyond lighting and viewpoint diversity, synthetic data generation introduces numerous possibilities, such as placing objects in different positions and orientations. This comprehensive variability enhances the robustness and applicability of the synthetic dataset, making it a valuable asset for training advanced object detection models.

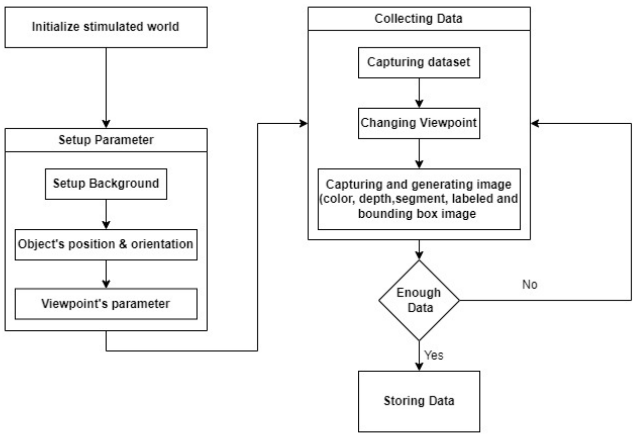
Initialize stimulated world
Setup Parameter
Setup Background
Object's position & orientation
Viewpoint's parameter
Collecting Data
Capturing dataset
Changing Viewpoint
Capturing and generating image (color, depth,segment, labeled and bounding box image
Enough Data
No
Yes
Storing Data

Fig. 1. Flow chart of the synthetic data generation.

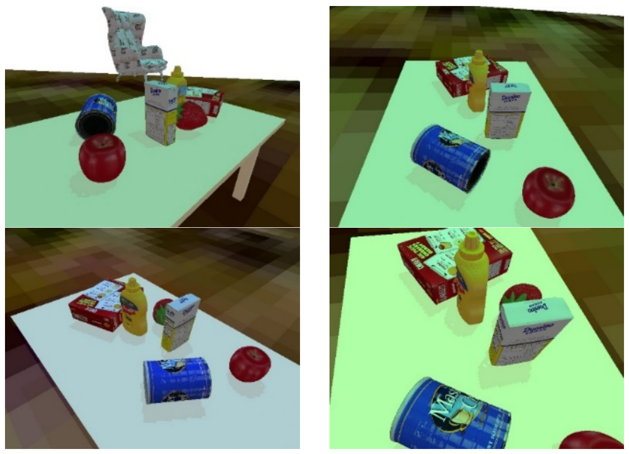

Fig. 2. Examples of different viewpoints and contexts in simulation environment.

After completing the background setup, the model begins data collection. During this phase, the camera's viewpoint is frequently changed in order to provide an overview scene of the object. Furthermore, the light conditions are varied in order to maintain the diversity of the data. Once the capturing phase is done, the model generates the RGB images, depth images, segmented areas, bounding box images, and labeled files with bounding box's annotations formatted for YOLOv5. The process is repeated until the image reaches the required volume. The resulted files are then organized into separate folders, ensuring structured and accessible data for subsequent analysis and model training.

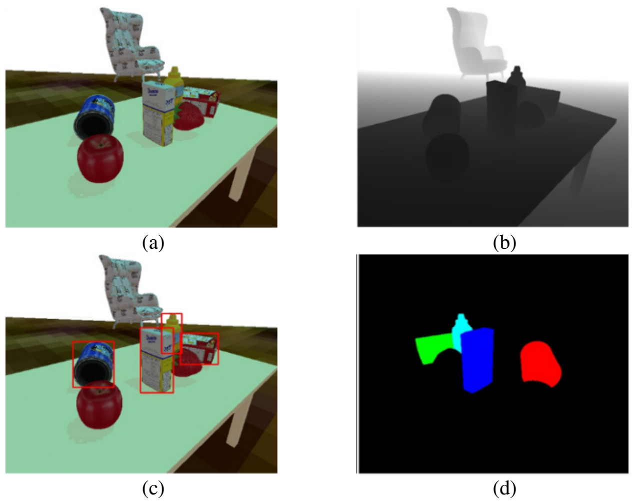

Fig. 3. Examples of synthetic images and automatically labeled images. The virtual camera in simulation environment gathers data including raw and labeled images. (a) RGB image, (b) depth image, (c) detection labeled image, (d) segmentation labeled image.

III. EXPERIMENT PREPARATIONS AND RESULTS

In this paper, there are three results related to three parts of our study: Synthetic data generation, YOLOv5 training process, and object detection inference.

A. Data Generation and Model Training

YOLOv5 (You Only Look Once) is a native extension of YOLOv3. In this update, the author has brought a lot of changes that make YOLOv5 perform better and easier to adapt than its previous version [12]. An in-depth analysis of implementing YOLOv5 for object identification is provided in [13]. Authors in [14] applied pre-trained object detection models in order to enhance the performance of logo detection tasks. In this paper, we selected YOLOv5 as our main training model as shown in Figure 4.

After generating the needed dataset, we split the data into a training folder and validate folder with 80% of the dataset used for training and 20% used for evaluation. We then cloned the YOLOv5 repository from the author's GitHub and run it on the Colab platform to implement object detection on the YCB real image dataset. The purpose of this paper is to evaluate the performance of object detection task when using the synthetic dataset, thus the structure of the YOLOv5 model is employed in the training process. In this experiment, we focused on 4 objects which are equivalent to 4 classes in the custom model file.

Figure 5 describes the training results. There are three different types of loss: bounding box regression loss, objectness loss, and classification loss. The box loss measures how accurately the algorithm can identify an object's center and how completely the predicted bounding box contains the object. The objectness loss represents the error between the prediction that the boundary box contains the object and the actual label in that square. The classification loss shows the algorithm's ability to correctly anticipate the class of a given item. Moreover, precision measures how much of the bounding box predictions are correct, while the recall indicates the chances that the bounding boxes are correctly predicted. Throughout the training process depicted in Figure 5, the reduction in loss remains stable, indicating consistent improvement. The model rapidly enhances its performance in terms of precision, recall, and mean average precision (mAP). Using the YOLOv5s model, we achieved a mAP of 94.6% at 0.5 IoU (Intersection over Union) and 93.1% at 0.5:0.95 IoU. These results demonstrate the effectiveness of the synthetic dataset in training a robust object detection model, underscoring the potential of our simulation framework for generating high-quality synthetic data.

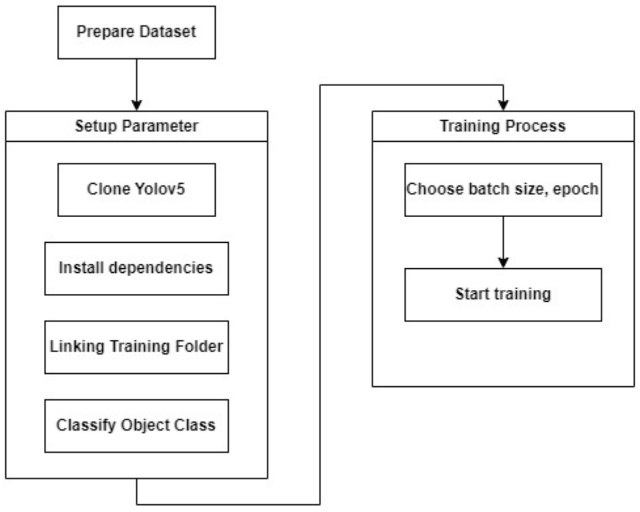

Fig. 4. Flow chart of the model training process with YOLOv5.

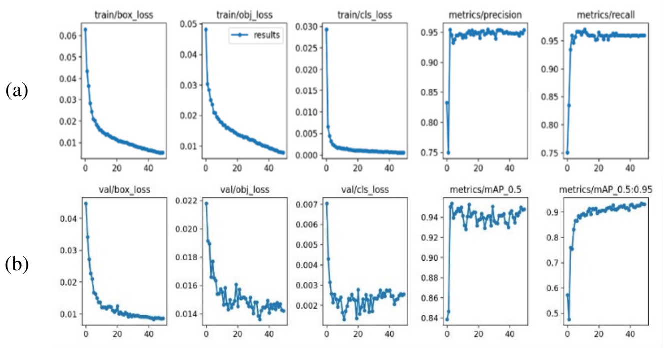

Fig. 5. Results of the training process with generated synthetic data. (a) Training evaluation results indicate metrics of object bounding box loss, objectness loss, object classification loss, precision, and recall. (b) Testing evaluation results illustrate same metrics but on the valuation dataset.

It can be clearly seen form Figure 6 that all the objects are well-detected in each image, regardless of the varying camera angles and light condition. However, the detection score for the cracker box, while adequate, could benefit from additional training data to further enhance its accuracy. This suggests that while the synthetic dataset is robust, increasing the volume of data could improve detection performance for specific objects.

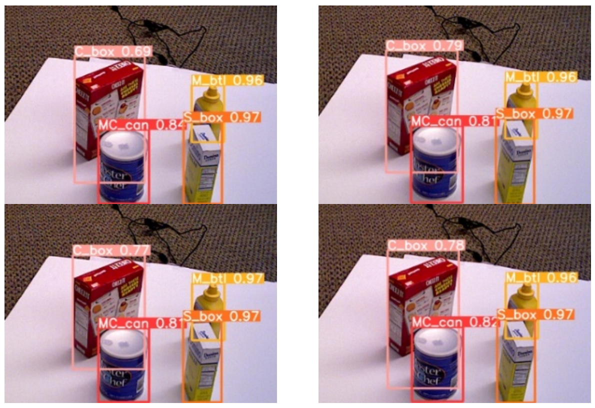

Fig. 6. Examples of object detection results with the generated synthetic data.

B. Robot Simulation

After building the model, we implement it in a simulated robot arm system with the help of MoveIt, an open-source project that is the result of the combined efforts of a large international community and multiple organizations [15]. Firstly, the arm is set to move to a known position, normally at a top place that the camera can observe clearly the whole view. Meanwhile, the gripper starts to open. After observing the object clearly, the system camera can run the model that was built by YOLOv5 to detect the object in 2D. By identifying the object's center in 2D, we then command the arm to move closer to the object. In the gripper, a distance sensor is attached to check if the object is already in the finger range. If the distance is not satisfied, the arm will continue to move closer to the object. At the moment that the distance is enough, we set a command to close the gripper. Then the arm can lift the object up and move to a desire location as shown in Figure 7.

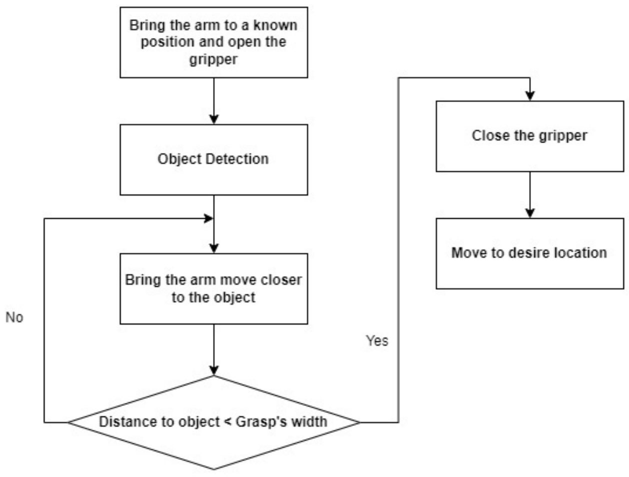
Bring the arm to a known position and open the gripper
Object Detection
Bring the arm move closer to the object
Distance to object < Grasp's width
No
Yes
Close the gripper
Move to desire location

Fig. 7. Motion planning of the robot arm.

In Figure 8, we can see the visualization of the motion planning that has been set up in the MoveIt environment. A cricket ball is used to test the success of the model. In 5 trials, the robot arm is always successful in grasping the object, lifting it in the air, and moving it to another location. The experiment was built on a computer equipped with Intel core i5 8th Gen and Nvidia GeForce GTX 1050 graphics card. One Linux-based operating system that can provide great flexibility to interact with pyBullet is Ubuntu. The advantage of Ubuntu in this context is its responsiveness, lightness, and high security. Apart from these factors, Ubuntu has great community support. Because of these reasons, we chose Ubuntu as the main operating system to run the project.

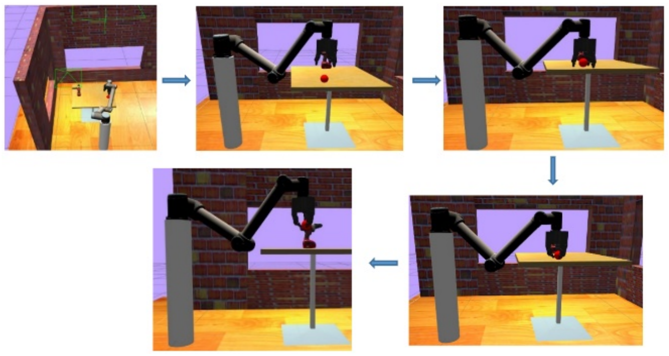

Fig. 8. Example of grasping robot simulation.

IV. CONCLUSION

In this paper, the development of a comprehensive simulation framework utilizes the pyBullet library to generate synthetic data and represents a significant advancement in the field of object detection. This framework automates the creation of extensive datasets, effectively bypassing the labor-intensive and time-consuming traditional data collection methods. Compared to traditional methods like GANs and Bayesian Networks, our approach offers a more efficient and effective solution to dataset preparation challenges by using the open source pyBullet library to surpass the limitations with obscured images and computation complexity. By leveraging sophisticated simulation techniques, we can generate diverse and realistic data to provide a straightforward solution for addressing the demand for labeled data in artificial intelligent research and development, thereby enhancing the efficiency and scalability of dataset production.

Furthermore, our experiments with a simulated grasping robot validate the framework's potential for real-world application. The positive outcomes of these tests highlight the capability of synthetic data to train models effectively, achieving high accuracy in complex tasks. This success not only substantiates our research but also paves the way for broader utilization of synthetic data in various domains, from autonomous robotics to advanced computer vision systems, demonstrating the promising future of this innovative approach.

REFERENCES

[1] R. Szeliski, *Computer Vision: Algorithms and Applications*. London, UK ; New York, NY, USA: Springer, 2010.

[2] A. Buetti-Dinh *et al.*, "Deep neural networks outperform human expert's capacity in characterizing bioleaching bacterial biofilm composition," *Biotechnology Reports*, vol. 22, Jun. 2019, Art. no. e00321, https://doi.org/10.1016/j.btre.2019.e00321.

[3] C. Joshi, I. Kaloskampis, and L. Nolan, "Generative adversarial networks (GANs) for synthetic dataset generation with binary classes," *Data Science Campus*, Feb. 21, 2019. https://datasciencecampus.ons.gov.uk/projects/generative-adversarial-networks-gans-for-synthetic-dataset-generation-with-binary-classes/.

[4] J. Young, P. Graham, and R. Penny, "Using Bayesian Networks to Create Synthetic Data," *Journal of Official Statistics*, vol. 25, no. 4, pp. 549–567, 2009.

[5] H. Zhang, W. Gao, X. Chen, and D. Zhao, "Object detection using spatial histogram features," *Image and Vision Computing*, vol. 24, no. 4, pp. 327–341, Apr. 2006, https://doi.org/10.1016/j.imavis.2005.11.010.

[6] P. Piccinini, A. Prati, and R. Cucchiara, "Real-time object detection and localization with SIFT-based clustering," *Image and Vision Computing*, vol. 30, no. 8, pp. 573–587, Aug. 2012, https://doi.org/10.1016/j.imavis.2012.06.004.

[7] Y. Ren, C. Zhu, and S. Xiao, "Object Detection Based on Fast/Faster RCNN Employing Fully Convolutional Architectures," *Mathematical Problems in Engineering*, vol. 2018, no. 1, 2018, Art. no. 3598316, https://doi.org/10.1155/2018/3598316.

[8] Z. Zou, K. Chen, Z. Shi, Y. Guo, and J. Ye, "Object Detection in 20 Years: A Survey," *Proceedings of the IEEE*, vol. 111, no. 3, pp. 257–276, Jan. 2023, https://doi.org/10.1109/JPROC.2023.3238524.

[9] D. Kalashnikov *et al.*, "Scalable Deep Reinforcement Learning for Vision-Based Robotic Manipulation," in *Proceedings of The 2nd Conference on Robot Learning*, Oct. 2018, pp. 651–673.

[10] M. Kothapalli and P. Manohar, "The Challenges of Data Quality and Data Quality Assessment in the Big Data," Apr. 2023, https://doi.org/10.13140/RG.2.2.21308.10885.

[11] B. Calli *et al.*, "Yale-CMU-Berkeley dataset for robotic manipulation research," *The International Journal of Robotics Research*, vol. 36, no. 3, pp. 261–268, Mar. 2017, https://doi.org/10.1177/0278364917700714.

[12] D. Thuan, "Evolution of Yolo algorithm and Yolov5: The State-of-the-Art object detention algorithm," B.Sc. thesis, Oulu University of Applied Sciences, 2021.

[13] T. Saidani, "Deep Learning Approach: YOLOv5-based Custom Object Detection," *Engineering, Technology & Applied Science Research*, vol. 13, no. 6, pp. 12158–12163, Dec. 2023, https://doi.org/10.48084/etasr.6397.

[14] S. Sahel, M. Alsahafi, M. Alghamdi, and T. Alsubait, "Logo Detection Using Deep Learning with Pretrained CNN Models," *Engineering, Technology & Applied Science Research*, vol. 11, no. 1, pp. 6724–6729, Feb. 2021, https://doi.org/10.48084/etasr.3919.

[15] S. Chitta, I. Sucan, and S. Cousins, "MoveIt![ROS topics]," *IEEE Robotics & Automation* Magazine, vol. 19, no. 1, pp. 18-19, 2012, https://doi.org/10.1109/MRA.2011.2181749.