2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Close the Sim2real Gap via Physically-based Structured Light Synthetic Data Simulation

Kaixin Bai[1,2], Lei Zhang[1,2], Zhaopeng Chen[2*], Fang Wan[3*], Jianwei Zhang[1]

***Abstract*— Despite the substantial progress in deep learning, its adoption in industrial robotics projects remains limited, primarily due to challenges in data acquisition and labeling. Previous sim2real approaches using domain randomization require extensive scene and model optimization. To address these issues, we introduce an innovative physically-based structured light simulation system, generating both RGB and physically realistic depth images, surpassing previous dataset generation tools. We create an RGBD dataset tailored for robotic industrial grasping scenarios and evaluate it across various tasks, including object detection, instance segmentation, and embedding sim2real visual perception in industrial robotic grasping. By reducing the sim2real gap and enhancing deep learning training, we facilitate the application of deep learning models in industrial settings. Project details are available at https://baikaixin-public.github.io/structured_light_3D_synthesizer/.**

I. Introduction

Data collection for computer and robotic vision tasks, particularly for object segmentation and 6D pose annotation [1], is labor-intensive and challenging. Additionally, gathering industrial data for deep learning models can be problematic due to factory rules, confidentiality, and safety concerns.

To address the challenges in obtaining real-world data, sim2real methods have been proposed to generate synthetic RGB images in 3D simulators for tasks such as robotic perception [2]–[5], autonomous driving [6], [7], intelligent agriculture solutions [8], [9], consumer and manufacturing [4], [10], and medical treatment [11], to reduce manual labor and improve the performance of deep learning models. Domain randomization [12] has been employed to generate photorealistic RGB images. This approach minimizes the sim2real gap by altering lighting, object material, and texture, but demands expert rendering knowledge and significant optimization within the 3D simulator.

While various tools have been created to generate photorealistic simulation datasets, they are constrained by the performance and capabilities of their respective simulation engines [2]. Differing strengths between game engines and film industry renderers, as well as inconsistencies like the left-handed coordinate system, present challenges for tasks like object pose estimation and robotic applications.

Decreasing prices of RGBD cameras have increased their use for computer and robot vision tasks, particularly for 3D visual perception and semantic information of the environment. This has led to an increase in using depth images as inputs to neural networks or in combination with RGB images [13], [14], to improve the performance of vision tasks. Additionally, RGBD images are used as inputs for multimodal perception tasks [15], [16]. Depth images have been used as inputs for neural networks to train robots for perception and grasping tasks [17], [18]. Researchers have attempted to reduce the gap between real and simulated depth images by generating physically-based depth images using stereo cameras or TOF cameras in virtual environments [19], [20], or applying post-processing using neural networks such as GANs [17], [21] to further align them with real ones.

Robotic tasks in industrial settings require visual recognition capabilities for diverse objects in terms of location, placement direction, type, shape, and size. These tasks often require identifying a large number of objects for grasping in cluttered scenes with objects of different sizes, which typically requires matching object instance segmentation or object detection to localize the objects and pose estimation based on point cloud or texture analysis to determine the grasping pose with high accuracy. Structure light cameras are widely used in industries like automotive, and logistics. They can provide 2D and 3D information with high precision and are adaptable to various requirements such as anti-ambient light, high accuracy, high reconstruction speed, and small size.

We propose a data generator for physically-based gray code structure light camera simulation. This generates photorealistic RGB and physically-based synthetic depth data, complete with 3D reconstruction algorithm noises, for robotic sim2real tasks. Our key contributions are:

- A physically-based gray code structured light camera simulation data generator, built using the Blender Cycles rendering engine and Optix AI denoiser, which generates photorealistic RGB data, physically-based synthetic depth data, and ground truth annotations for object pose, 2D/3D bounding boxes, and segmentation.
- A dataset with physically-realistic simulated RGBD data as a training set and real data as a test set, which can be utilized to evaluate the sim2real performance gap and generalization ability of vision perception tasks like object detection and instance segmentation.
- We provide a real-world demonstration of the effectiveness of our sim2real data generation and robot perception network based on this data generation method in actual robot tasks.

II. Related Work

A. Synthetic Dataset Generation

The trend of training robots on synthetic datasets and transferring to real-world datasets is gaining traction. Various simulation data generation tools and plug-ins have emerged, ranging from game engines like Omniverse [22] and Unreal Engine 4 [7], [23], [24], to 3D simulation tools like

*Corresponding authors.
[1]TAMS (Technical Aspects of Multimodal Systems), Department of Informatics, Universität Hamburg, Germany, [2]Agile Robots AG, Munich, Germany , [3]School of Design, Southern University of Science and Technology, Shenzhen, Guangdong, China

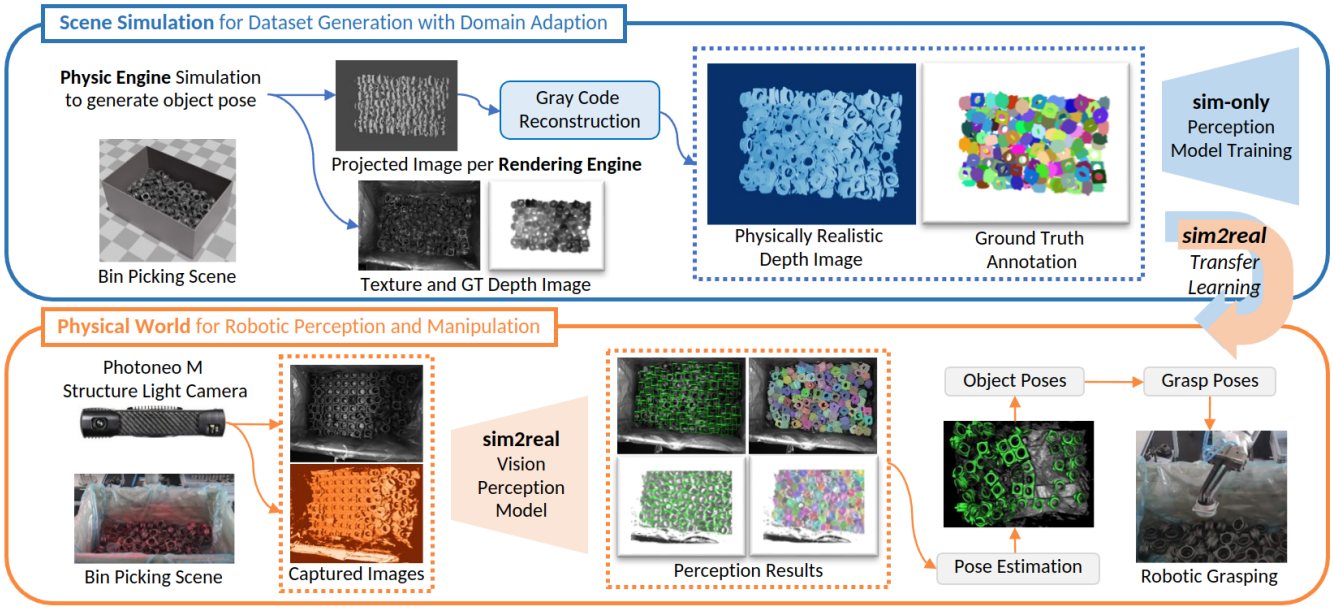

Fig. 1. Pipeline of physically-based sim2real transfer learning: We built a physically-based simulator with gravity to generate realistic data of cluttered scenarios. Then we use ray tracing to perform structured light projection, then decode and reconstruct the projected image to render physically-based realistic depth images with ground truth annotations for instance segmentation. Next, we train the vision perception using simulation only and perform sim2real transfer learning on the trained model to ensure good perception results in real-world scenarios. After that, we apply pose estimation to obtain the poses of objects and perform model-based robotic grasping.

Blender [5], [25]–[27] and PyBullet [3], [4]. These tools vary in supported programming languages, headless rendering capabilities, and real-time rendering performance. While game engines often excel in frame rates using raster-based methods, they may lack accurate light transport simulation due to the absence of ray tracing, which limits their rendering performance for reflective and transparent objects.

*1) **Sim2real Gap**:* The sim2real gap remains a major hurdle for deep learning methods trained on synthetic datasets for vision tasks and robotic manipulation. This gap arises due to disparities between synthetic RGB data and real-world conditions, influenced by environmental and camera parameters. Minimizing this gap often requires extensive optimization of object materials, lighting, and sensor characteristics. To tackle this, researchers employ domain randomization techniques to vary colors, lighting, and noise [12], [24], and domain adaptation methods to address data domain mismatches, particularly in GANs training [21], [28].

B. Physically-based Synthetic Depth Sensor Simulation

Industrial 3D cameras are commonly classified into passive stereo, active stereo, ToF, and structured light types. Physically-based sensor simulation is crucial for improving the quality of datasets for vision and robotic tasks. Depth images are increasingly favored in perception tasks due to their lower sim2real gaps. For example, Danielczuk et al. [29] explored object segmentation methods using depth images and non-photorealistic RGB images for feature learning.

Various methods have been developed to narrow the sim2real gap in depth images. For instance, Zakharov et al. [17] and Danielczuk et al. [18] utilize ground truth depth images to train robots for perception tasks. Others like Planche et al. [19] and Gschwandtner et al. [20] generate synthetic depth images using virtual stereo or TOF cameras. To further align synthetic and real depth images, Zakharov et al. [17] and Yuan et al. [21] employ neural networks like GANs for post-processing. The significance of depth image simulation across different 3D cameras is thus evident.

III. Method

To close the simulation-reality gap for gray code structured light cameras, we've created a data simulator using Blender. Using physically-based rendering, we simulate scenes and apply gray code patterns. Realistic depth images are then generated through 3D reconstruction. Our system employs NVIDIA's OptiX engine and GeForce RTX 3070 Ti GPU for rendering and includes OptiX's AI-accelerated denoiser [30] to enhance image render speed. Fig. 1 outlines the physically-based rendering pipeline for robotic pick-and-place tasks, while Fig. 2 shows pattern projection and 3D reconstruction. We focus on ray tracing's acknowledged benefits, without comparing it to other rendering techniques like rasterization rendering.

A. Physically-Based Rendering

In our study, we employ physically-based rendering to simulate real-world scenes featuring a wooden box with densely arranged objects. This involves dividing the space above the box into voxel grids, sampling these grids based on object count, and then dropping the objects into the box while considering realistic physics like collisions. The final renders include synthetic RGB and ground-truth depth images, as well as instance segmentation. Real-world RGBD images and point clouds from an empty wooden box are incorporated for domain adaptation.

Structured light cameras often suffer from decoding errors due to varying light paths caused by material properties and

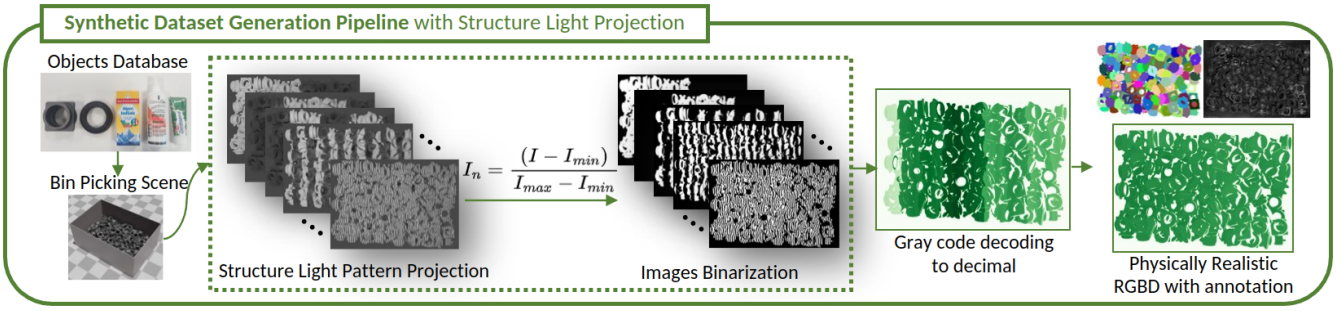

Fig. 2. The system architecture of the proposed pattern projection simulation and 3D reconstruction methods is as follows: First, we generate a scene using the proposed physically-based rendering technique. Then, we project various gray code patterns into the scene. These patterns serve as binary images for 3D reconstruction. The system includes a structured-light camera setup consisting of one projector and one camera. This setup captures the projected patterns and the scene. Finally, we estimate a synthetic depth image with realistic structured light noise using our proposed methods.

lighting conditions. This introduces noise into depth maps. To capture this realistically, our simulator uses ray tracing for gray code pattern projection, enabling accurate depth maps with noise characteristics similar to real-world applications. This approach bypasses the shortcomings of rasterization-based methods, which neglect light path calculations, resulting in less realistic images.

B. Pattern Projection

To simulate the rendering process based on the physical principles of structured light cameras in the real world, we present a projector based on a spotlight with textured light in Blender to simulate the pattern projection of structured light cameras. During pattern projection, the gray code pattern images are separately projected onto the top of the object scene using our proposed projector. The corresponding gray images are then rendered, as shown in Fig. 2.

The rendering of high-quality images is typically time-consuming in the film industry. In synthetic dataset generation, using the same workflow as film rendering is not cost-effective for vision tasks and robotic manipulation. In recent years, AI algorithms have been applied to reduce the rendering time for high-fidelity images [31]. For example, the AI denoiser with Optix integration in Blender can speed up rendering for pixels with a limited number of samples. It is even possible to achieve real-time rendering after just two frames while maintaining the quality of the rendered images [32]. To speed up our dataset generation, we use the AI denoiser to render both RGB images and projected pattern images after rendering 20 frames. The parameter is chosen based on our qualitative comparison experiment.

C. 3D Reconstruction with Gray Code Pattern

To reconstruct the scene from images rendered with gray code pattern projection, we first generate binary images, as illustrated in Fig. 2. Each point within the projected area experiences at least one brightness shift, achieved by incorporating both fully black and fully white images. For every pixel, we compute its temporal maximum and minimum values to establish a binarization threshold. A pixel is assigned a value of 1 if the threshold exceeds 0.5, and 0 otherwise. We then use 3D reconstruction structured-light techniques, as described in [33], to reconstruct the scene. In this process, the projector is modeled as a pinhole camera. We obtain decoding images based on the gray code encoding, and the object's imaging position on the camera plane in pixels is represented by (u_c, v_c). The virtual imaging position under the projector camera model is indicated by (u_p, v_p). Using this information, we can model the structured-light system with one projector and one camera using the following formula:

$$s_c \begin{bmatrix} u_c \\ v_c \\ 1 \end{bmatrix} = K_c[R_c|t_c] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, s_p \begin{bmatrix} u_p \\ v_p \\ 1 \end{bmatrix} = K_p[R_p|t_p] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{1}$$

where K_c and K_p are the intrinsic parameters of the camera and the projector. When the camera coordinate system coincides with the world coordinate system, rotation matrix R_c can be formulated as unit matrix and translation vector t_c of camera can be formulated as all zero matrix. By decoding the Gray code we can obtain the correspondence of each pixel under the camera and projector model. Finally, we can reconstruct the 3D scene with the following equation:

Fig. 3. Object Database. a) Metal Workpiece 1. b) Metal Circle Workpiece 2. c)YellowSaltCube (KIT object models database). d) HygieneSpray (KIT object models database). e) Toothpaste (KIT object models database). f) The corresponding real objects.

$$
\begin{aligned}
M_c =& K_c[R_c \mid t_c] = \begin{bmatrix} m_{11}^c & m_{12}^c & m_{13}^c & m_{14}^c \\ m_{21}^c & m_{22}^c & m_{23}^c & m_{24}^c \\ m_{31}^c & m_{32}^c & m_{33}^c & m_{34}^c \end{bmatrix} \\
M_p =& K_p[R_p \mid t_p] = \begin{bmatrix} m_{11}^p & m_{12}^p & m_{13}^p & m_{14}^p \\ m_{21}^p & m_{22}^p & m_{23}^p & m_{24}^p \\ m_{31}^p & m_{32}^p & m_{33}^p & m_{34}^p \end{bmatrix} \\
& \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = M \begin{bmatrix} m_{34}^c u_c - m_{14}^c \\ m_{34}^c v_c - m_{24}^c \\ m_{34}^p u_p - m_{14}^p \end{bmatrix} \\
M =& \begin{bmatrix} m_{11}^c - m_{31}^c u_c & m_{12}^c - m_{32}^c u_c & m_{13}^c - m_{33}^c u_c \\ m_{21}^c - m_{31}^c v_c & m_{22}^c - m_{32}^c v_c & m_{23}^c - m_{33}^c v_c \\ m_{11}^p - m_{31}^p u_p & m_{12}^p - m_{32}^p v_c & m_{13}^p - m_{33}^p u_p \end{bmatrix}^{-1}
\end{aligned} \tag{2}
$$

IV. Experiments

To validate the efficacy of our synthetic data simulator designed for gray code structured light cameras, we assess its performance on tasks like instance segmentation and object detection. For this, we employ a synthetic training dataset and a real-world testing dataset. Initially, we introduce an object database specifically crafted for generating cluttered environments. Subsequently, we produce an extensive, photo-realistic dataset featuring single-class, multi-instance scenes, suitable for tasks such as instance segmentation and object detection. A real-world dataset, aligned with our object database, serves as the testing set.

A. Object Database

In our object database, we have gathered two metal parts in a dark gray color from industry, as well as three household objects from the KIT object models database [34]. These objects encompass common items found in pick and place scenarios, including industrial components and supermarket merchandise frequently encountered in robotic tasks.

B. Synthetic and Real-world Datasets

To explore vision tasks and robotic operations in industrial settings, we create an extensive synthetic dataset featuring single-class, multi-instance scenes from our object database. This dataset comprises rendered RGB and depth images, synthetic depth images using gray code structured light reconstruction, along with ground truths for object poses, 2D bounding boxes, and instance segmentation masks. For the objects in our database, we employ Run-Length Encoding (RLE) for ground truth representation in synthetic data and use polygon encoding for real-world data to streamline manual annotations. In our physics-based rendering simulation, each scene is a single class with multiple instances. For the two types of metal parts, there are a maximum of 255 objects in the physics simulation and rendering in each scene, and for objects in the KIT dataset, there are a maximum of 10 objects in each bin box. In the synthetic domain adaption dataset, we generate 1,000 scenes for each object in the database after simulating pattern projection and 3D reconstruction. For each scene we generate colored image, ground truth depth image, our proposed depth image and instance segmentation map. For the real-world dataset, we collect 100 sets of data for each object as the test set. In addition, we generated a domain randomization dataset using Isaac Sim [22] for comparison.

C. Object Detection and Instance Segmentation Experiments

To validate our synthetic data pipeline, we separately examine object detection and instance segmentation tasks using both real and synthetic RGB and depth images. We benchmark using YOLOv3 and SOLOv2 for these tasks, and test on real-world datasets. Quantitative results are summarized in Table I. We opted for YOLOv3 due to its lack of data augmentation like Mosaic used in later versions, ensuring fairness in validating dataset efficacy.

To gauge the real-world efficacy of our domain-adapted dataset, we employ YOLOv7 to compare performance against a domain-randomized dataset in perception tasks. YOLOv7's data augmentation features align well with industrial scenarios. The randomized dataset, created with Isaac Sim, varies scene backgrounds, object quantities and poses, and lighting as shown in Fig. 5.

D. Robotic grasping experiment

To evaluate our vision perception model in robotic grasping experiments, we build the setup of model-based and half-model-based grasping experiments. Both setups are designed to address the sim2real performance gap. For the model-based grasping experiment, we employ the Diana7 robot and Photoneo L industrial part. Meanwhile, the half-model grasping experiment involves the UR5e robot and a Photoneo M camera as shown in Fig. 6. In our experiments, we employ the objects from our dataset to execute bin-picking tasks, adhering to the half-model-based grasping procedure as detailed in [35] and Dex-Net3.0 [36]. The performance of visual perception has a profound influence on the computational burden of the grasping algorithm and directly affects the success rate of grasping. Our approach involves initially detecting the object in a depth image via an object detection technique, and then applying a grasping detection method. We further examine the variations in grasping success rate and manipulation speed following the integration of our visual perception approach. Both setups are devised with the aim of demonstrating that the use of sim2real in visual perception tasks can significantly enhance the performance metrics, such as success rate and algorithm runtime, in robotic grasping applications.

V. Results

A. Qualitative Study

Assessing the disparity between reconstructed depth data from our synthetic data simulator and real-world scene data, we conduct a qualitative analysis via localized visualizations. Fig. 7 a) depicts a real-world scene's local depth image of cluttered metal workpiece 1, while Fig. 7 b) displays synthetic data from a structured-light camera (green) and rendered 3D model data (blue). Our generator can simulate shadow and sharp noise of structured-light cameras based on the difference between its synthetic data and 3D model renderings.

Fig. 7 b) further illustrates the efficacy of our proposed structured light-based data simulator. The depicted point cloud noise from our simulator closely matches the noise from a real structured-light camera, suggesting minimal data disparity between our simulated depth images and point

TABLE I

QUANTITATIVE EVALUATION RESULTS OF OBJECT DETECTION AND INSTANCE SEGMENTATION.

Tasks		Object Detection (YOLOv3)*			Instance Segmentation (SOLOv2)*		
Object	Input Type	Test Set Type**		Sim2real Gap***	Test Set Type**		Sim2real Gap***
		Synthetic	Real		Synthetic	Real	
Metal Workpiece 1.	RGB	0.624	0.547	0.077	0.711	0.620	0.091
	Ground Truth Depth	0.613	0.527	0.086	0.692	0.557	0.135
	Proposed Synthetic Depth	0.634	**0.617**	0.017	0.698	**0.646**	0.052
Metal Circle Workpiece 2.	RGB	0.673	0.614	0.059	0.646	0.565	0.081
	Ground Truth Depth	0.662	0.633	0.029	0.636	0.544	0.092
	Proposed Synthetic Depth	0.674	**0.628**	0.046	0.634	**0.627**	0.007
Toothpaste	RGB	1.000	0.885	0.115	1.000	0.953	0.047
	Ground Truth Depth	0.999	0.950	0.049	1.000	0.970	0.03
	Proposed Synthetic Depth	1.000	**0.980**	0.020	1.000	**0.979**	0.021
YellowSaltCube	RGB	1.000	0.826	0.174	1.000	0.621	0.379
	Ground Truth Depth	0.996	0.678	0.318	0.998	0.802	0.196
	Proposed Synthetic Depth	1.000	**0.827**	0.173	1.000	**0.863**	0.137
HygieneSpray	RGB	1.000	0.947	0.053	1.000	0.963	0.037
	Ground Truth Depth	0.999	0.797	0.202	1.000	0.875	0.125
	Proposed Synthetic Depth	0.980	**0.953**	0.027	1.000	**0.966**	0.034

* Average precision (AP) @[Intersection over Union (IoU)=0.50]
** Synthetic: Evaluation with synthetic dataset. Real: Evaluation with real-world dataset.
*** Sim2real Gap: Difference of evaluation results in different types of test set.

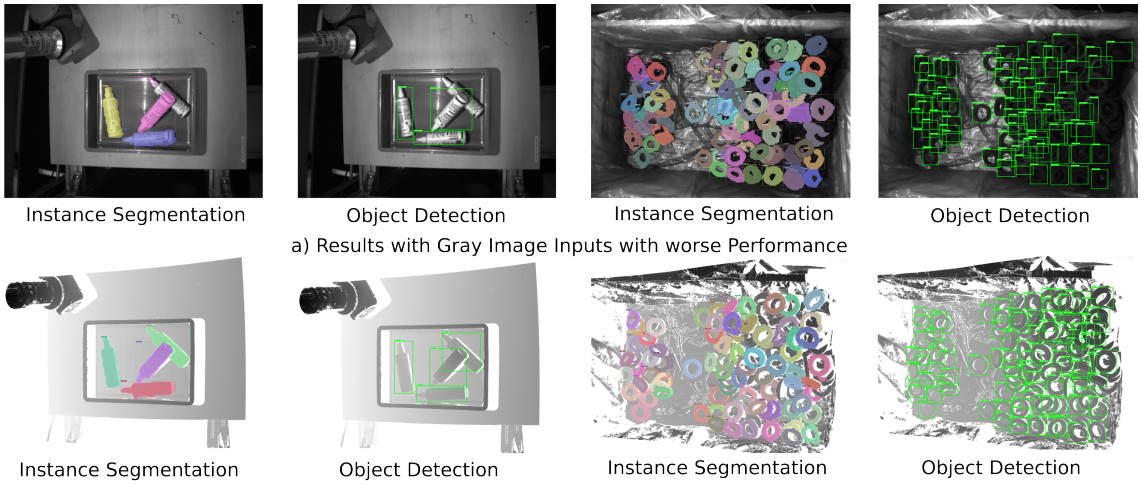

Fig. 4. Qualitative Visualization results of visual perception in real-world dataset using trained models of our synthetic datasets. The depth image-based deep learning model has better performance than the model with texture image as input.

clouds and the real counterparts. This infers that our simulator likely incurs less performance loss in visual perception tasks using depth images or point clouds.

B. Quantitative Studies

1) Object Detection and Instance Segmentation: Table I details the quantitative results of our data simulation methods in object detection and instance segmentation tasks, with varying input types (RGB or depth images) and test datasets (synthetic or real-world).

Initially, a sim2real gap emerges when evaluating the trained model on real-world data across all objects. This gap is measured by subtracting the average precision results of intersection over union (IoU) in real-world data from the synthetic data results. This sim2real gap is positive for selected household objects, while qualitative results suggest its presence for two metal industrial objects, attributed to issues like shadows and sharp noise.

Depth images are less sensitive to lighting and appearance changes, offering robustness against environmental variations. Using depth images as input can lessen a model's computational load, often containing less information than RGB images. They allow the distinction between object shape and texture, enhancing certain perception tasks' performance. For instance, object recognition often emphasizes shape over texture, and using a depth map as input enables model focus on the object's shape. Moreover, depth images provide additional information about camera-object distances, beneficial for tasks like 3D reconstruction and robot navigation.

In addition, we conducted tests on the YOLOv7 model using datasets based on both our domain adaptation approach and domain randomization to ascertain the potential performance of our proposal in real-world projects. The experiments demonstrated that the domain adaptation-based RGB dataset achieved an IoU of 0.628 on the real dataset for the Metal Workpiece 1 object, whereas the model based on our proposed depth image achieved an IoU of 0.648. Both exceeded the IoU of 0.584 derived from the domain adaptation-based RGB dataset. The analysis suggests that in industrial grasping scenarios, which are usually static, the

Fig. 5. The creation of a domain-randomized dataset using the Isaac Sim platform. The dataset generation features varied scene backgrounds, object numbers and poses, and lighting conditions in terms of intensity and color. Each parameter is randomized to increase the diversity of the dataset.

Fig. 6. Comparative experimental setups showcasing robotic grasping. On the left, a model-based grasping scenario is depicted, utilizing a Diana7 robot and a Photoneo L camera for industrial part manipulation. On the right, a half-model based grasping approach is illustrated, employing a UR5e robot and a Photoneo M camera.

performance boost brought by domain adaptation surpasses that of domain randomization.

In conclusion, utilizing depth images as input in deep learning perception tasks can bolster model robustness, efficiency, and performance, providing a more durable representation of 3D structure and facilitating shape-texture separation. Moreover, they exhibit less sensitivity to lighting and appearance changes.

C. Robotic Grasping Experiment

Incorporating sim2real perception into our half-model-based grasping pipeline resulted in an uptick in successful grasping rates from 95.6% to 98.0% with Dex-Net3.0. The introduction of precise object detection concentrates the sampling of grasp points more on a single object, ultimately increasing the probability of grasp points being centered on the object's plane. As a result, the success rate of grasping also improves.

The sim2real approach significantly improved the model-based grasping task, which traditionally relies on error-prone and limited manual data annotation for object detection training. This method not only reduced project development

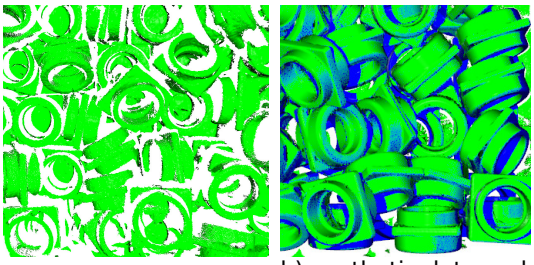

Fig. 7. Qualitative Results of Structured Light Noise. a) Localized area of real-world point cloud of cluttered objects (Metal Workpiece 1.). b) Partical area of synthetic data with synthetic shadows(zero value in depth image) and sharp noise(flying noise), rendered point cloud in green and 3D model with ground truth pose in blue.

time from two weeks to two days but also enhanced detection precision, thus bolstering system robustness. It mitigated issues related to oversized bounding boxes increasing pose estimation time and undersized ones compromising estimation accuracy and grasping success.

The vision perception model in our experiment, trained within a simulated environment and leveraging our proposed synthetic structured light-based depth images, performed on par in real-world scenarios, successfully completing the robotic grasping task. The design of our data generation pipeline, mindful of sensor noise generation, enables effective domain adaptation in real-world applications.

VI. Conclusion

Despite the progress in deep learning for perception, its industrial application remains limited due to the high cost and time required for data annotation and model adaptation. To address this, we introduce a sim2real data generation tool designed for 3D structured light cameras, commonly used in industrial robotics. The tool uses physics-based simulations to generate realistic depth maps and annotations, enabling efficient sim2real transfer learning with minimal performance loss. This innovation is crucial for integrating deep learning into industrial contexts.

Our quantitative analysis validates the tool's efficacy in perception tasks, highlighting that our physically realistic synthetic depth inputs accelerate domain adaptation and improve network performance. This significantly reduces the time needed for domain randomization, a common bottleneck in prior works.

Looking ahead, our roadmap includes expanding the Fraunhofer IPA Bin-Picking dataset and optimizing pose estimation and robotic grasping algorithms using our generated dataset, catering to a broader range of industrial applications.

Acknowledgment

This research has received funding from the German Research Foundation (DFG) and the National Science Foundation of China (NSFC) in project Crossmodal Learning, DFG TRR-169/NSFC 61621136008, partially supported by ULTRACEPT (778602).

REFERENCES

[1] P. Marion, P. R. Florence, L. Manuelli, and R. Tedrake, "Label fusion: A pipeline for generating ground truth labels for real rgbd data of cluttered scenes," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 3235–3242.

[2] N. Morrical, J. Tremblay, Y. Lin, S. Tyree, S. Birchfield, V. Pascucci, and I. Wald, "Nvisii: A scriptable tool for photorealistic image generation," *arXiv preprint arXiv:2105.13962*, 2021.

[3] J. Josifovski, M. Kerzel, C. Pregizer, L. Posniak, and S. Wermter, "Object detection and pose estimation based on convolutional neural networks trained with synthetic data," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 6269–6276.

[4] A. Ummadisingu, K. Takahashi, and N. Fukaya, "Cluttered food grasping with adaptive fingers and synthetic-data trained object detection," *arXiv preprint arXiv:2203.05187*, 2022.

[5] J. Arents, B. Lesser, A. Bizuns, R. Kadikis, E. Buls, and M. Greitans, "Synthetic data of randomly piled, similar objects for deep learning-based object detection," in *International Conference on Image Analysis and Processing*. Springer, 2022, pp. 706–717.

[6] T. Pollok, L. Junglas, B. Ruf, and A. Schumann, "Unrealgt: using unreal engine to generate ground truth datasets," in *International Symposium on Visual Computing*. Springer, 2019, pp. 670–682.

[7] S. Khan, B. Phan, R. Salay, and K. Czarnecki, "Procsy: Procedural synthetic dataset generation towards influence factor studies of semantic segmentation networks." in *CVPR workshops*, 2019, pp. 88–96.

[8] Y. Toda, F. Okura, J. Ito, S. Okada, T. Kinoshita, H. Tsuji, and D. Saisho, "Training instance segmentation neural network with synthetic datasets for crop seed phenotyping," *Communications biology*, vol. 3, no. 1, pp. 1–12, 2020.

[9] R. Barth, J. IJsselmuiden, J. Hemming, and E. J. Van Henten, "Data synthesis methods for semantic segmentation in agriculture: A capsicum annuum dataset," *Computers and electronics in agriculture*, vol. 144, pp. 284–296, 2018.

[10] M. Z. Wong, K. Kunii, M. Baylis, W. H. Ong, P. Kroupa, and S. Koller, "Synthetic dataset generation for object-to-model deep learning in industrial applications," *PeerJ Computer Science*, vol. 5, p. e222, 2019.

[11] J. Cartucho, S. Tukra, Y. Li, D. S. Elson, and S. Giannarou, "Visionblender: a tool to efficiently generate computer vision datasets for robotic surgery," *Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization*, pp. 1–8, 2020.

[12] 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," in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2017, pp. 23–30.

[13] Y. Xiang, C. Xie, A. Mousavian, and D. Fox, "Learning rgb-d feature embeddings for unseen object instance segmentation," in *Conference on Robot Learning (CoRL)*, 2020.

[14] L. Huang, B. Zhang, Z. Guo, Y. Xiao, Z. Cao, and J. Yuan, "Survey on depth and rgb image-based 3d hand shape and pose estimation," *Virtual Reality & Intelligent Hardware*, vol. 3, no. 3, pp. 207–234, 2021.

[15] X. Xu, Y. Li, G. Wu, and J. Luo, "Multi-modal deep feature learning for rgb-d object detection," *Pattern Recognition*, vol. 72, pp. 300–313, 2017.

[16] C. Wang, D. Xu, Y. Zhu, R. Martín-Martín, C. Lu, L. Fei-Fei, and S. Savarese, "Densefusion: 6d object pose estimation by iterative dense fusion," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 3343–3352.

[17] S. Zakharov, B. Planche, Z. Wu, A. Hutter, H. Kosch, and S. Ilic, "Keep it unreal: Bridging the realism gap for 2.5 d recognition with geometry priors only," in *2018 International Conference on 3D Vision (3DV)*. IEEE, 2018, pp. 1–11.

[18] M. Danielczuk, M. Matl, S. Gupta, A. Li, A. Lee, J. Mahler, and K. Goldberg, "Segmenting unknown 3d objects from real depth images using mask r-cnn trained on synthetic point clouds," *arXiv preprint arXiv:1809.05825*, vol. 16, 2018.

[19] B. Planche, Z. Wu, K. Ma, S. Sun, S. Kluckner, O. Lehmann, T. Chen, A. Hutter, S. Zakharov, H. Kosch *et al.*, "Depthsynth: Real-time realistic synthetic data generation from cad models for 2.5 d recognition," in *2017 International Conference on 3D Vision (3DV)*. IEEE, 2017, pp. 1–10.

[20] M. Gschwandtner, R. Kwitt, A. Uhl, and W. Pree, "Blensor: Blender sensor simulation toolbox," in *International Symposium on Visual Computing*. Springer, 2011, pp. 199–208.

[21] C. Yuan, Y. Shi, Q. Feng, C. Chang, Z. Chen, A. C. Knoll, and J. Zhang, "Sim-to-real transfer of robotic assembly with visual inputs using cyclegan and force control," *arXiv preprint arXiv:2208.14104*, 2022.

[22] M. Rojas, G. Hermosilla, D. Yunge, and G. Farias, "An easy to use deep reinforcement learning library for ai mobile robots in isaac sim," *Applied Sciences*, vol. 12, no. 17, p. 8429, 2022.

[23] W. Qiu, F. Zhong, Y. Zhang, S. Qiao, Z. Xiao, T. S. Kim, and Y. Wang, "Unrealcv: Virtual worlds for computer vision," in *Proceedings of the 25th ACM international conference on Multimedia*, 2017, pp. 1221–1224.

[24] T. To, J. Tremblay, D. McKay, Y. Yamaguchi, K. Leung, A. Balanon, J. Cheng, W. Hodge, and S. Birchfield, "NDDS: NVIDIA deep learning dataset synthesizer," 2018, https://github.com/NVIDIA/Dataset_Synthesizer.

[25] C. Heindl, L. Brunner, S. Zambal, and J. Scharinger, "Blendtorch: A real-time, adaptive domain randomization library," in *International Conference on Pattern Recognition*. Springer, 2021, pp. 538–551.

[26] C. Mata, N. Locascio, M. A. Sheikh, K. Kihara, and D. Fischetti, "Standardsim: A synthetic dataset for retail environments," in *International Conference on Image Analysis and Processing*. Springer, 2022, pp. 65–76.

[27] S. Im, H. Ha, H.-G. Jeon, S. Lin, and I. S. Kweon, "Deep depth from uncalibrated small motion clip," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 4, pp. 1225–1238, 2019.

[28] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan, "Unsupervised pixel-level domain adaptation with generative adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 3722–3731.

[29] Y. Xiang, C. Xie, A. Mousavian, and D. Fox, "Learning rgb-d feature embeddings for unseen object instance segmentation," in *Conference on Robot Learning*. PMLR, 2021, pp. 461–470.

[30] S. G. Parker, J. Bigler, A. Dietrich, H. Friedrich, J. Hoberock, D. Luebke, D. McAllister, M. McGuire, K. Morley, A. Robison *et al.*, "Optix: a general purpose ray tracing engine," *Acm transactions on graphics (tog)*, vol. 29, no. 4, pp. 1–13, 2010.

[31] C. R. A. Chaitanya, A. S. Kaplanyan, C. Schied, M. Salvi, A. Lefohn, D. Nowrouzezahrai, and T. Aila, "Interactive reconstruction of monte carlo image sequences using a recurrent denoising autoencoder," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, pp. 1–12, 2017.

[32] X. Zhang, R. Chen, F. Xiang, Y. Qin, J. Gu, Z. Ling, M. Liu, P. Zeng, S. Han, Z. Huang *et al.*, "Close the visual domain gap by physics-grounded active stereovision depth sensor simulation," *arXiv preprint arXiv:2201.11924*, 2022.

[33] B. Li, Y. An, D. Cappelleri, J. Xu, and S. Zhang, "High-accuracy, high-speed 3d structured light imaging techniques and potential applications to intelligent robotics," *International Journal of Intelligent Robotics and Applications*, vol. 1, no. 1, pp. 86–103, 2017.

[34] A. Kasper, Z. Xue, and R. Dillmann, "The kit object models database: An object model database for object recognition, localization and manipulation in service robotics," *The International Journal of Robotics Research*, vol. 31, no. 8, pp. 927–934, 2012.

[35] G. Du, K. Wang, S. Lian, and K. Zhao, "Vision-based robotic grasping from object localization, object pose estimation to grasp estimation for parallel grippers: a review," *Artificial Intelligence Review*, vol. 54, no. 3, pp. 1677–1734, 2021.

[36] J. Mahler, M. Matl, X. Liu, A. Li, D. Gealy, and K. Goldberg, "Dex-net 3.0: Computing robust vacuum suction grasp targets in point clouds using a new analytic model and deep learning," in *2018 IEEE International Conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 5620–5627.