Machine Learning and Artificial Intelligence
A.J. Tallón-Ballesteros and C. Chen (Eds.)

doi:10.3233/FAIA200770

AutoSynPose: Automatic Generation of Synthetic Datasets for 6D Object Pose Estimation

Heiko ENGEMANN [a,b], Shengzhi DU [a,1], Stephan KALLWEIT [a,b,2], Chuanfang NING [c] and Saqib ANWAR [b]
[a] *Tshwane University of Technology (TUT), Pretoria, South Africa*
[b] *FH Aachen – Aachen University of Applied Sciences, Aachen, Germany*
[c] *Ecole polytechnique fédérale de Lausanne (EPFL), Lausanne, Switzerland*

Abstract. We present an automated pipeline for the generation of synthetic datasets for six-dimension (6D) object pose estimation. Therefore, a completely automated generation process based on predefined settings is developed, which enables the user to create large datasets with a minimum of interaction and which is feasible for applications with a high object variance. The pipeline is based on the Unreal 4 (UE4) game engine and provides a high variation for domain randomization, such as object appearance, ambient lighting, camera-object transformation and distractor density. In addition to the object pose and bounding box, the metadata includes all randomization parameters, which enables further studies on randomization parameter tuning. The developed workflow is adaptable to other 3D objects and UE4 environments. An exemplary dataset is provided including five objects of the Yale-CMU-Berkeley (YCB) object set. The datasets consist of 6 million subsegments using 97 rendering locations in 12 different UE4 environments. Each dataset subsegment includes one RGB image, one depth image and one class segmentation image at pixel-level.

Keywords. synthetic dataset, 6D pose estimation, domain randomization

1. Introduction

Computer vision based on deep neural networks (DNNs) enables robots to perceive their environment in a human like manner. Intelligent robots depend on robust perception strategies to perform their key tasks: autonomous navigation [1,2] and adaptive manipulation [3,4]. State-of-the-art approaches for 6D pose (object position and orientation) estimation [5–9] as well as object tracking [10–12] and novel grasping techniques [13–16], enable adaptive task execution and closed loop control of high precision manipulation tasks.

One major factor closely related to the model performance is the quality and quantity of the training dataset [17]. Traditionally, the generation of a dataset for object detection is time-consuming and costly. It is a two-step task, which is divided into: data acquisition and data annotation, including manually executed operations [18]. The difficulty is

[1] Corresponding Author: Shengzhi Du, E-Mail: dus@tut.ac.za
[2] Corresponding Author: Stephan Kallweit, E-Mail: kallweit@fh-aachen.de

increased in case of 6D pose estimation datasets, caused by the need of object pose information in each dataset subsegment.

In contrast to data sets obtained from the real world, the generation of synthetic data can be fully automated, making it suitable for computer vision tasks with high object variations. The major challenge using synthetic data is closing the gap between the simulated data and its real-world counterpart, i.e. the reality gap. For this reason, different techniques were developed, divided into two basic categories: domain adaptation and reality match. Domain adaptation describes the problem to generalize a trained model from its source space to an unknown target space [19]. Reality match approaches simulate the feature rich real-world data as realistic as possible. It was proved that a model, fully trained on synthetic data, can provide state-of-the-art performance for 6D pose estimation [7]. This supports the hypothesis that synthetic data is an effective alternative when the acquisition and annotation of real-world data is not feasible.

In this paper, we present a fully automated pipeline for the generation of a 6D object pose dataset. The result is a significant reduction in preparation time and a minimization of necessary human interaction. The pipeline can be easily adapted to different objects and environments. It provides multiple improvements to comparable approaches, such as various parameter settings and detailed metadata for each dataset subsegment. An exemplary dataset is generated including six million subsegments. Each subsegment consists of an RGB, depth and class segmentation image at pixel level. Two metadata files provide the 6D object pose and a corresponding 3D bounding box as well as the parameter settings for randomization. Based on the detailed metadata, the data set can be divided into smaller sub-datasets, allowing the study of different parameter combinations.

2. Related Work

Methods to generate synthetic data for training neural networks can be split into two general categories: superimposing and rendering.

The main idea of superimposing is to project object images onto background images. In [20] a method is presented to generate synthesized data from existing real-world datasets. Object images are extracted by cropping the images according to the provided metadata. The object images are projected onto another set of background images, using support surfaces estimated by semantic segmentation and plane fitting. The dataset in [21] is generated using rendering models and real-world images. This method was improved in [22] by adding variations like random illumination, noise and blurring of the object images before superimposing. The implementation of superimposing is straightforward and easy to automate. However, the missing interaction between the environment and the object results in a reality gap due to the absence of important real-world features, e.g. shadows.

Rendering refers to the generation of an image from 2D or 3D models as well as complete environments [23]. Typical rendering software provide numerous setting options for e.g. the ambient lighting, the object appearance, the shape and the poses. Due to the numerous setting options, the generation of datasets in detail is a very time-consuming process. Various methods were developed to provide synthetic data - as realistic as possible - based on rendering. Domain randomization techniques [4] increase the ability of generalizing the trained model to real-world data. An optical flow dataset [24] was presented derived from an animated short film by modifying the motion blur pipeline of the 3D creation suite Blender [25]. The resulting Sintel dataset has been

extended to depth and segmentation images. In [26] the Unity development platform [27] was used to create a virtual urban world, populated with cars, vans, pedestrians and cyclists. The resulting Synthia dataset was generated using a dynamic illumination engine and provides different scene appearances related to the four seasons. The approach proposed in [28] creates a unique virtual world for each dataset subsegment, instead of generating the whole dataset based on a single virtual world. The resulting Synscape dataset [29] is a photorealistic synthetic dataset for street scene parsing. The dataset SceneNet-RGBD [30] provides indoor scenes including household objects. It is an extension of the work presented in [31], using an automatic random scene generator based on 174683 potential 3D objects. In contrast to the previous mentioned approaches, the synthetic dataset Falling things (FAT) [32] is focusing the field of robotic manipulation. Therefore, it uses objects of the real-world Yale-CMU-Berkeley (YCB [33]) object set. In addition to RGB, depth and segmentation images, each dataset subsegment also includes metadata in form of 6D pose information. The SIDOD dataset [34] is closely related, since both datasets were generated using the Unreal Engine 4 (UE4 [35]) and the open-source custom plugin NVIDIA DeepLearning Data Synthesizer (NDDS [36]), a tool to extract view data during rendering.

3. Method

Our approach differentiates from others by focusing a high automation level to enable a wide parameter range for randomization and less human interaction. The pipeline was developed based on the game engine UE4, providing functionalities to predefine parameter settings for environments and objects, including the manual chosen rendering locations, later referred to as training spots. The actual rendering process is highly automated, which simplifies the generation of large object-related datasets and the usage of synthetic data in real world applications. An exemplary dataset was generated using a RTX 2080 Ti, where each rendering process per dataset subsegment takes around 50ms. The UE4 project, including one open-source environment is available at: http://autosynpose.fh-aachen.de.

3.1. Domain Randomization

The presented pipeline simulates the following aspects with an user adjustable variability: environment, training spot, object appearance, camera-object transformation, ambient lighting, number and size of geometric primitive distractors, presence of complex object distractors, and multiple instances of the object of interest.

The object appearance can be randomized by an adjustable percentage for the base color. In addition, the appearance parameters roughness, metallic and specular are configurable. The camera describes a motion alongside the surface of a sphere during the automated rendering processes (see Figure 1). The camera motion is reduced to a circular motion in case of training spots on flat surfaces, e.g. tables. The origin of the sphere coordinate system S is located at the manual selected training spot. The optical axes of the camera, which is represented by the z-axis of the camera coordinate system C, points towards the origin of the coordinate system S during the rendering processes.

The object of interest is randomly positioned in a 3D cube with adjustable dimensions. The so-called object cube describes a linear motion alongside the z-axis of the coordinate system C during the rendering processes. Therefore the location of the

object-cube origin can be described as a vector $\boldsymbol{o}_{oc} = (0,0,z_{oc})^T$ in reference to the coordinate system C, with 0.2 m $\leq z_{oc} \leq$ 1.5 m. The pivot point of the vector $\boldsymbol{o}_{oc}$ is located at the origin of the coordinate system S, during the motion of the camera and the object-cube. This approach generates a large number of dataset subsegments with different perspectives. An additional 3D cube encloses the object-cube. Inside this distractor-cube, the distractors are positioned randomly around the object of interest, including geometric primitive and object distractors. The distractor-cube will be partly outside the field of view of the camera at close distances. Thus, not all distractors will be present in the corresponding image data. Therefore, we introduce the concept of distractor density, which depends on the size of the distractor-cube, the size of the distractors itself and the number of distractors present.

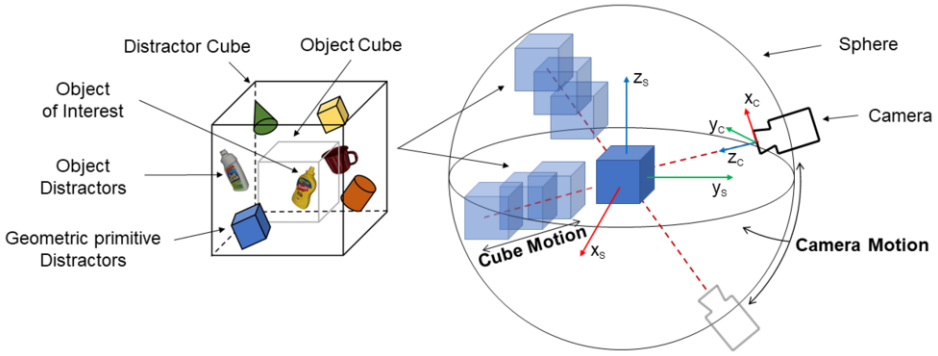

Figure 1. Spherical camera motion around the object of interest.

3.2. Metadata

The presented pipeline provides two metadata files in JSON format for each dataset subsegment. The first one is generated by NDDS, including the 6D object pose and its bounding box in the coordinate system C. The second file contains detailed information of the parameter configuration for domain randomization as shown in table 1.

Table 1. Description of the metadata file for domain randomization.

Category	Name	Description
Generating Info	EnvironmentNo	Environment Number
	SpotNo	Spot Number
Appearance	HueShiftPercent	Shift in object hue in percent
	Roughness	Object roughness value
	Metallic	Object metallic value
Distractors	WithDistractors	Presence of distractors
	DistractorSize	Size of primitive distractors in 10 mm
	DistractorCount	Number of distractors in scene
	WithObjectDistractors	Presence of object distractors
Configurations	WithLightningVariations	Presence of lightning variations
	MultipleInstances	Presence of multiple object instances
	InstanceCount	Number of object instances
	MultipleObjects	Presence of additional objects

4. Dataset Analysis

The proposed pipeline was used to generate the exemplary dataset *AutoSynPose*, which can be downloaded here: http://autosynpose.fh-aachen.de. The dataset includes six million subsegments rendered at 97 training spots spread over 12 environments. The environments vary from realistic indoor to game-like outdoor scenes (see Figure 2) in order to bypass the machine learning specific problem that the factors of variation cannot always be directly observed [37].

Figure 2. Environments of the AutoSynPose dataset.

As object of interest, the mustard bottle of the YCB dataset was chosen. In addition, the YCB objects: toy plane, meat can, power drill and hammer are in 17% of the dataset present. Each subsegment consist of one RGB image, one depth image and one segmentation image providing class instance information at pixel level. The subsegment does not include object instance information to reduce the size of the dataset. Figure 3 shows one subsegment including the object of interest and geometric primitive as well as object distractors.

Figure 3. Subsegment of the AutoSynPose dataset.

Figure 4 shows the statistics of the dataset, across the different environments and manually selected training spots. Furthermore, Figure 4 shows the composition of the dataset regarding the presence of distractors, lightning variations, multiple instances of the object of interest and multiple objects.

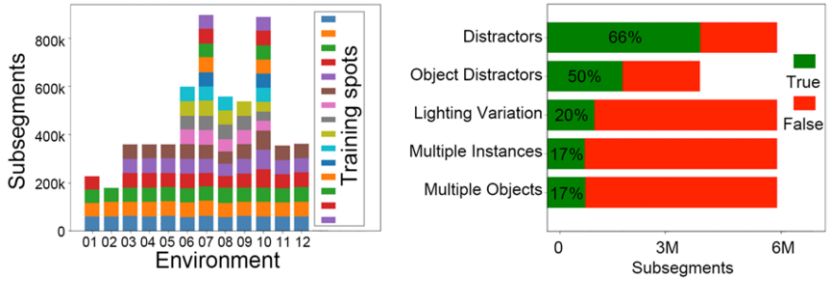

Figure 4. Dataset statistics. Left: Distribution of subsegments over environments and training spots. Right: Dataset composition.

Figure 5 shows a statistical analysis of the six million dataset subsegments. The edge length of the object-cube was set to 0.3m and the edge length of the distractor-cube was set to 0.75m.

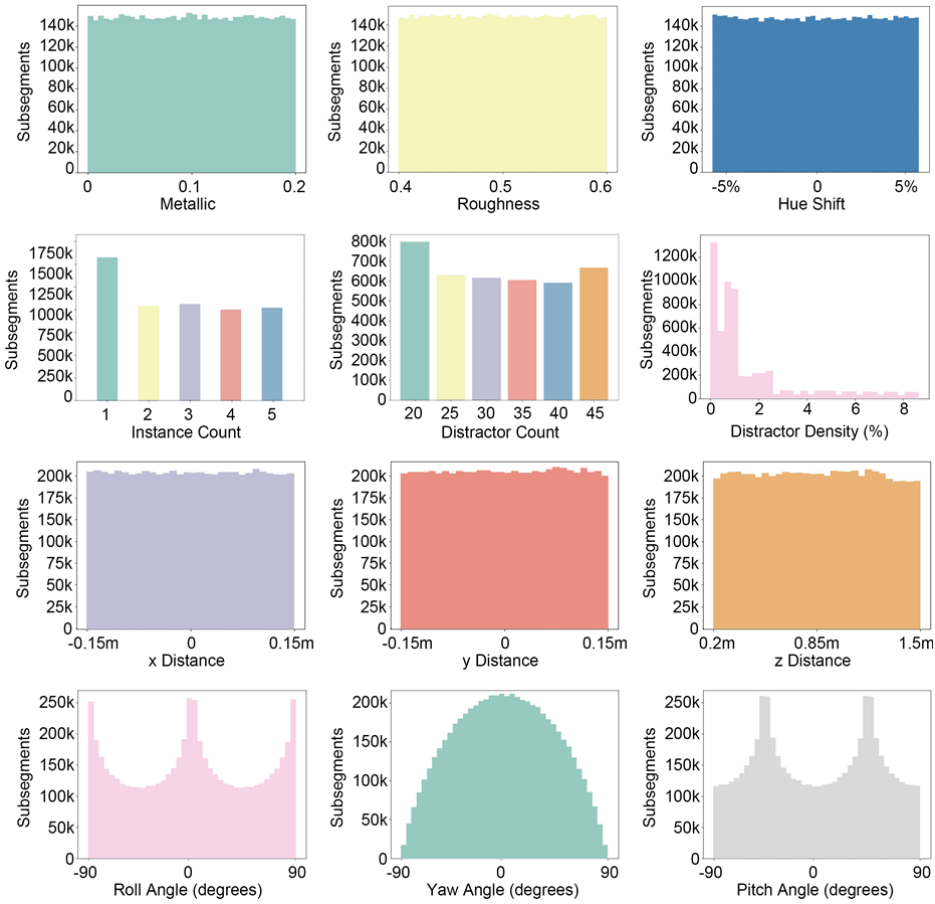

Figure 5. Statistical analysis of the dataset: Object appearance, object instance count, geometric primitive distractors, and camera-object transformation.

The first column of Figure 5 shows the randomization of the object appearance, which is equally distributed around the selected parameter settings. The second column shows the distribution of the instance count and the geometric primitive distractors. The distractor density inside the distractor-cube varies from 0% up to 8%, with a peak in the range between 0% and 1%. The third column shows the distribution of the 3D object position in the camera coordinate system, which is almost uniformly distributed. The distributions of the Euler angles reflect the orientation of the object in the camera coordinate system.

Table 2 shows our dataset in contrast to comparable state-of-the-art datasets. To our knowledge the AutoSynPose dataset is the only 6D object pose dataset providing detailed information about parameter settings used for domain randomization at subsegment level.

Table 2. Comparison of synthetic datasets (Masks* IS – instance segmentation, CS – class segmentation; FD* – Flying Distractors. DI* – Domain Randomization Information).

Dataset	#Segments	Scenes	Masks*	Poses	B-Box	FD*	DI*
Sintel (2015) [24]	1.6 k	movie	IS	3D	✗	✗	✗
Synthia (2016) [26]	200 k	urban	CS	✗	✗	✗	✗
SN-RGBD (2016) [31]	5 M	household	IS CS	3D	✗	✗	✗
Synscape (2018) [29]	25 k	urban	IS CS	3D	2D	✗	✗
FAT (2018) [32]	60 k	household	IS CS	6D	3D	✗	✗
SIDOD (2019) [34]	115 k	household	IS CS	6D	3D	✓	✗
Ours (2020)	6 M	various	CS	6D	3D	✓	✓

The SIDOD and the FAT dataset are closely related to our dataset. Table 3 shows a detailed comparison of the three datasets.

Table 3. Detailed comparison with the synthetic datasets FAT and SIDOD.

Parameter	FAT (2018) [32]	SIDOD (2019) [34]	Ours (2020)
No. of objects	21	21	5
No. of environments	3	3	12
No. of training spots	15	18	97
Object appearance variations	✗	✗	✓
Lightning variations	✓	✓	✓
Object position distribution	normal	normal	uniform
Object orientation distribution	mixed	mixed	mixed

We were able to significantly increase the number of environments and training spots, because of the high automation level of the proposed pipeline. In addition to the lightning domain, our pipeline also provides configurable settings for the object appearance domain. Another improvement is the uniform distribution of the object position in the camera coordinate system. The compared datasets provide a normal distribution for x and y, whereby the mean is located at the optical axis of the camera. The same applies for the camera-object distance, whereby the mean is located at the mid of the intended range. A uniform distribution represents a larger number of different perspective viewpoints and reduces overfitting problems at test time. The focus of our dataset is to enable studies on the impact of different parameter settings for domain randomization. Therefore, the AutoSynPose dataset only includes five different objects, whereby the mustard bottle of the YCB object set is prioritized.

5. Conclusion

We developed an automated synthetic dataset generating pipeline, based on the game engine UE4 and the open-source plugin NDDS. Adjustable parameters for domain randomization and the rendering process allow a simple change between different objects of interests and environments. The required knowledge about game and rendering engines is reduced. The pipeline generates a uniform distribution of the 3D object position in reference to the camera coordinate system. In addition, the presented camera-object motion concept covers different perspective viewpoints per training spot. To our knowledge, it is the first approach providing detailed information about the parameter settings used for domain randomization at subsegment level. In addition, we introduced the concept of distractor density as a metric for distractors. We hope that the proposed

pipeline makes it easier for other researchers to generate synthetic data from 3D models and environments.

The presented pipeline can be used to synthesize a variety of different datasets. However, the identification of the important factors of variation, which cannot directly be observed, has the potential to decrease the reality gap of future synthetic dataset generation. Therefore, the exemplary generated dataset AutoSynPose aims to study the effect of randomization parameters in detail.

6. References

[1] Milioto A, Mandtler L, Stachniss C. Fast instance and semantic segmentation exploiting local connectivity, metric learning, and one-shot detection for robotics. In International Conference on Robotics and Automation (ICRA), 2019, pp. 5481–5487.
[2] Zou Q, Jiang H, Dai Q, Yue Y, Chen L, Wang Q. Robust lane detection from continuous driving scenes using deep neural networks. In IEEE Transactions on Vehicular Technology, 2019, 69(1), pp. 41-54.
[3] Schwarz M, Behnke S. Semantic RGB-D perception for cognitive service robots. In RGB-D Image Analysis and Processing, Springer, Cham, 2019, pp. 285–307.
[4] Tobin J, Fong R, Ray A, Schneider J, Zaremba W, Abbeel P. Domain randomization for transferring deep neural networks from simulation to the real world. In IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2017, pp. 23–30.
[5] Hu Y, Hugonot J, Fua P, Salzmann M. Segmentation-driven 6D object pose estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 3385-3394.
[6] Xiang Y, Schmidt T, Narayanan V, Fox D. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. arXiv preprint arXiv:1711.00199 (2017).
[7] 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).
[8] Deng X, Mousavian A, Xiang Y, Xia F, Bretl T, Fox D. Poserbpf: A rao-blackwellized particle filter for 6d object pose tracking. arXiv preprint arXiv:1905.09304 (2019).
[9] Wang C, Xu D, Zhu Y, Martín-Martín R, Lu C, Fei-Fei L, Savarese S. Densefusion: 6d object pose estimation by iterative dense fusion. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 3343–3352.
[10] Choi C, Christensen H I. 3D textureless object detection and tracking: An edge-based approach. In IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2012, pp. 3877–3884.
[11] Crivellaro A, Rad M, Verdie Y, Yi K M, Fua P, Lepetit V. A novel representation of parts for accurate 3D object detection and tracking in monocular images. In IEEE International Conference on Computer Vision (ICCV), 2015, pp. 4391–4399.
[12] Yuan D, Kang W, He Z. Robust visual tracking with correlation filters and metric learning. Knowledge-Based Systems (2020), 105697.
[13] Kumra S, Kanan C. Robotic grasp detection using deep convolutional neural networks. In IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2017, pp. 769–776.
[14] Morrison D, Corke P, Leitner J. Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach. arXiv preprint arXiv:1804.05172 (2018).
[15] Morrison D, Corke P, Leitner J. Learning robust, real-time, reactive robotic grasping. The International Journal of Robotics Research 39 (2020), 183–201.
[16] Mahler J, Matl M, Satish V, Danielczuk M, DeRose B, McKinley S, Goldberg K. Learning ambidextrous robot grasping policies. Science Robotics 4 (2019).
[17] Voulodimos A, Doulamis N, Doulamis A, Protopapadakis E. Deep learning for computer vision: A brief review. Computational intelligence and neuroscience 2018 (2018).
[18] Weiss G M, Provost F. Learning when training data are costly: The effect of class distribution on tree induction. Journal of Artificial Intelligence Research 19 (2003), 315–354.
[19] Wang M, Deng W. Deep visual domain adaptation: A survey. Neurocomputing 312 (2018), 135–153.
[20] Georgakis G, Mousavian A, Berg A C, Kosecka J. Synthesizing training data for object detection in indoor scenes. arXiv preprint arXiv:1702.07836 (2017).
[21] Josifovski J, Kerzel M, Pregizer C, Posniak L, Wermter S. Object detection and pose estimation based on convolutional neural networks trained with synthetic data. In IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2018, pp. 6269–6276.

[22] Hinterstoisser S, Lepetit V, Wohlhart P, Konolige K. On pre-trained image features and synthetic images for deep learning. In Proceedings of the European Conference on Computer Vision *(ECCV),* 2018, pp. 682-697.
[23] Akenine-Möller T, Haines E, Hoffman N. Real-time rendering*,* Crc Press, 2019.
[24] Butler D J, Wulff J, Stanley G B, Black M J. A naturalistic open source movie for optical flow evaluation. In European conference on computer vision, 2012, pp. 611–625.
[25] Blender Online Community, Blender - a 3D modelling and rendering package, http://www.blender.org [cited 2020 July 18].
[26] Ros G, Sellart L, Materzynska J, Vazquez D, Lopez A M. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 3234–3243.
[27] Hass J K. A history of the unity game engine (2014).
[28] Tsirikoglou A, Kronander J, Wrenninge M, Unger J. Procedural modeling and physically based rendering for synthetic data generation in automotive applications. arXiv preprint arXiv:1710.06270 (2017).
[29] Wrenninge M, Unger J. Synscapes: A photorealistic synthetic dataset for street scene parsing. arXiv preprint arXiv:1810.08705 (2018).
[30] McCormac J, Handa A, Leutenegger S, Davison A J. SceneNet RGB-D: 5M photorealistic images of synthetic indoor trajectories with ground truth. arXiv preprint arXiv:1612.05079 (2016).
[31] Handa A, Patraucean V, Badrinarayanan V, Stent S, Cipolla R. Understanding real world indoor scenes with synthetic data. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 4077–4085.
[32] Tremblay J, To T, Birchfield S. Falling things: A synthetic dataset for 3d object detection and pose estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2018, pp. 2038–2041.
[33] Calli B, Singh A, Walsman A, Srinivasa S, Abbeel P, Dollar A M. The YCB object and model set: Towards common benchmarks for manipulation research. In International Conference on Advanced Robotics (ICAR), 2015, pp. 510–517.
[34] Jalal M, Spjut J, Boudaoud B, Betke M. SIDOD: A synthetic image dataset for 3D object pose recognition with distractors. In IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2019, pp. 475–477.
[35] Epic Games. Unreal Engine, http://www.unrealengine.com [cited 2020 July 18].
[36] To T, Tremblay J, McKay D, Yamaguchi Y, Leung K, Balanon A, Cheng J, Hodge W, Birchfield S. NDDS: Nvidia Deep Learning Dataset Synthesizer, 2018.
[37] Goodfellow I, Bengio Y, Courville A. Deep Learning*,* MIT Press, 2016.