A Tool for Building Multi-purpose and Multi-pose Synthetic Data Sets

Marco Ruiz, Jefferson Fontinele, Ricardo Perrone, Marcelo Santos, Luciano Oliveira

Abstract Modern computer vision methods typically require expensive data acquisition and accurate manual labeling. In this work, we instead leverage the recent progress in computer graphics to propose a novel approach of designing and generating large scale multi-purpose image data sets from 3D object models directly, captured from multiple categorized camera viewpoints and controlled environmental conditions. The set of rendered images provide data for geometric computer vision problems such as depth estimation, camera pose estimation, 3D box estimation, 3D reconstruction, camera calibration, and also pixel-perfect ground truth for scene understanding problems, such as: semantic and instance segmentation, object detection, just to cite a few. In this paper, we also survey the most well-known synthetic data sets used in computer vision tasks, pointing out the relevance of rendering images for training deep neural networks. When compared to similar tools, our generator contains a wide set of features easy to extend, besides allowing for building sets of images in the MSCOCO format, so ready for deep learning works. To the best of our knowledge, the proposed tool is the first one to generate large-scale, multi-pose, synthetic data sets automatically, allowing for training and evaluation of supervised methods for all of the covered features.

Keywords: Synthetic data; 3D Rendering; multi-purpose; multi-pose; tool.

Marco Ruiz ✉
Intelligent Vision Research Lab, Federal University of Bahia, Brazil, e-mail: marco.ruiz@ufba.br

Jefferson Fontinele
Intelligent Vision Research Lab, Federal University of Bahia, Brazil, e-mail: jeffersonfs@ufba.br

Ricardo Perrone
Intelligent Vision Research Lab, Federal University of Bahia, Brazil, e-mail: perrones@ufba.br

Marcelo Santos
Intelligent Vision Research Lab, Federal University of Bahia, Brazil, e-mail: marceloms@ufba.br

Luciano Oliveira
Intelligent Vision Research Lab, Federal University of Bahia, Brazil, e-mail: lrebouca@ufba.br

1 Introduction

Results of convolutional neural network (CNN) methods based on supervised learning strongly depend on large-scale training data sets. However, the annotation of thousands of images and intrinsic aspects are a tedious and huge task which demands time in high precision human observation, even more for pixel-wise annotations or 3D information. To cope with these issues, works like the MSCOCO [9] make use of any massive collaborative annotation mechanism. Nevertheless, this solution poses some challenges to work synergistically that affects the standardization of the full process, promoting inaccuracies in the categorization process, the location of bounding boxes and poor boundary pixel annotation. In effect, these particularities increase the divergence on labeling thousands of images, thus demanding a detailed verification process at the final stage. On the other hand, rendered data sets can be automatically annotated, but mostly have been specially designed for representing a few of the more common aspects of real-world problems, *e.g.*, pose, depth, cluttering, lighting, etc; without allowing to extend features or examples of a new non-common object class. The cost and time to collect and label images are less when images are rendered from 3D models. Furthermore, synthetic data provides access to a reliable set of data for research, while not compromising on the principle of reproducibility [1].

Particularly with respect to data sets for deep learning methods, we highlight four issues: (i) the scarcity of training images with accurate annotations from different categorized viewpoints; (ii) the lack of powerful features specifically linked to 3D tasks; (iii) an impediment for rapid deployment of detection systems of less common objects, and; (iv) the mammoth time of image annotation. Recently, a successful research direction to overcome these four issues is to train CNNs from built scenarios with rendered synthetic objects [4]. In this paper, we propose to overcome these concerns by making publicly available a software tool that automatically generates multi-purpose and multi-pose images from 3D models. To expose our approach, we compare it at the feature level with other works and then present five examples of data sets generated with our tool, such as the *Car poses* exhibited in Fig. 1.

2 Related work

Can synthetic images be trained to represent inherent characteristics of real scenes?. Mainly focused on basic computer vision problems, this discussion has been frequently addressed during the last years [12, 18, 22]. The use of existing 3D models has been advocated in the past [4] and remains an appealing strategy [8]. Results expose that CNNs are capable of extracting discriminative features from synthesized images when evaluated on real scenarios, demonstrating competent and even better results compared to methods purely trained with real images. Training on synthetic rendered images represents an alternative solution for feeding networks with trustworthy data with perfect object semantic annotations [8]. In this way, the problem of

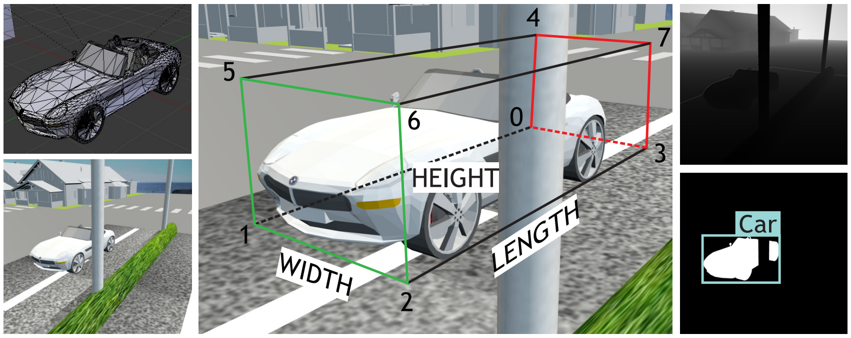


Fig. 1 The *Car poses* data set was generated with our tool by placing 3D car models in a virtual environment, then rendered from several camera viewpoints. Each snapshot consists of a 3D model (top left) and its corresponding RGB rendered image (bottom left), 3D bounding box coordinates and real size of cars (center), depth data in meters (top right), 2D bounding box coordinates and pixel-wise segmentation (bottom right), 3D poses of all target objects (not shown), among other annotations. Best viewed in color.

domain bias [1] (sometimes referred as domain gap [5]) between synthetic and real-world images has been conquered by using photorealistic scenarios [2, 20] combined with 3D representations [20], or by transferring style from real-world scenarios to rendered images [1]. Our virtual environment enables users to freely rich the scene with realistic 2D and 3D representations.

Tools or methods for building synthetic data sets: a software library whose purpose is quite similar to ours is introduced in [19]. The main goal is to allow the computer vision community for easily extending or generating data sets from 3D models, accordingly with different parameters like lighting, pose, and texture, and image metadata like image label, object outline, etc; finally outputting 2D bounding boxes automatically labeled, and a caption that users manually input to summarize the scene. To demonstrate the software's potential, the authors exposed two data sets that together with the code for data sets generation, are publicly available. Similarly, the work in [8] extracts a set of poses and class discriminant features from synthetic 3D object models, to build a viewpoint-independent detector. The approach they follow to extract multiple rendered views of synthetic 3D models is pretty similar to our method for automatic labeling. The latter generated a data set from 58 synthetic models containing 3D annotations similar to us, except the automatic annotation of depth, segmentation masks, and real object dimensions. The authors focused explicitly on the 2D/3D object detection application and did not make available the generated data set nor the code to build it.

Following a truly virtual approach, CARLA [3] and AirSim [16] bring a virtual environment capable of simulating a dynamic world, including physical phenomena like rain and other weather conditions. These simulators provide an open-source platform to build algorithms for autonomous vehicles, encompassing the features to design and develop by hand convenient hiper-realistic data sets automatically annotated. Similarly, the MINOS simulator [15] is designed to support the development of multi-sensory models for goal-directed navigation in indoor environments. MINOS is used to benchmark deep-learning-based navigation methods, showing that current deep reinforcement learning approaches fail in large realistic environments, also demonstrating that multi-modality is beneficial in learning to navigate cluttered scenes. A wide set of images can be rendered from the virtual indoor scene, including depth, GPS positions, and segmentation mask annotations. However, the simulator is not available to categorize camera poses nor other features for applications that aim to exploit the 3D domain of objects.

SURREAL [21] is not just a large-scale data set of people synthetically-generated rendered from 3D sequences of human motion capture data, but a publicly available software library to create new images or video sequences from a pre-configured virtual scene. Similar to our tool, the library allows to design some assets in the scene *e.g*, camera position, lighting, and textures, also using a database of images to compose the background for the rendered frames.

Synthetic data sets: much of the data sets are thought primarily for evaluating new CNN methods or aiming to release new benchmarks either for 3D context learning applications [17], visual odometry [6], 3D detection or reconstruction [6, 22], SLAM [6], or viewpoint estimation [18]. Despite coming from simulated environments, tightly intertwined problems are not addressed and consequently other context-relevant features are no longer explored. On the other hand, a small portion of the works create data for more general purposes, for instance for the context of driving scenarios [5, 7, 14], to wide range of 3D object reconstruction problems [23], or very specialized for particular tasks such as optical flow [2], or 3D scene understanding [23]. These latter manifest a greater effort to exploit the automatic annotation from virtual environments, delivering a larger number of features than other researchers can benefit from similar applications.

To show a comprehensive resume of the literature review, we group all discussed works here, in Table 1; highlighting the features that are most common among them and meeting the scope of computer vision applications that our tool could address.

Contributions

Rather than releasing a static data set, we propose a software tool intentionally designed to support the process of building well-planned data sets. This approach contains a wide set of features including a configurable scenario and discretization parameters for rendering images automatically labeled and formatted for supervised learning applications (especially for the training process). As part of this work, very relevant works are reviewed and compared from a feature-oriented strategy. According to the literature review, our work has several key strengths in comparison to others publicly available.

Here, we are focused on how to improve the process of generating a new data set or even expand an existing one. Our initial motivation came from the scarce availability of labeled data designed to support traffic surveillance problems. In particular, the majority of existing data is not well-suited to represent complex traffic scene and driving scenarios such as *top-down* traffic monitoring, law enforce, anomalies detection, 3D object detection, and segmentation, etc. In any application,

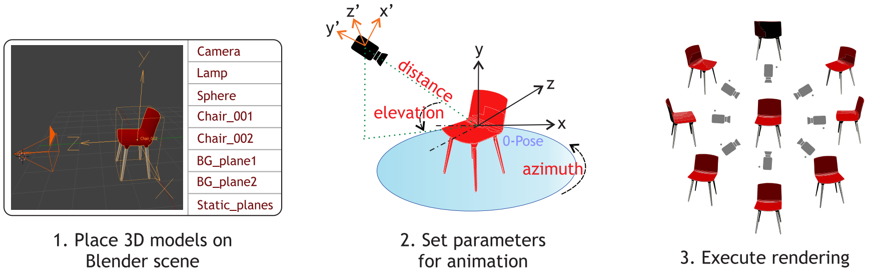
Table 1 Comparison of synthetic image data sets and tools.

Data set /Method	# frames	2D Box	3D Box in pxs	Mask	Disparity/ Depth	3D Model	Pose Categ.	Calibrat. Matrices	Optical flow	Real size
[2]	+1k	✗	✗	✓	✓	✗	✗	✗	✓	✗
[5]	+21k	✓	✓	✓	✓	✗	✓	✓	✓	✗
[14]	+213k	✓	✗	✓	✓	✗	✓	✓	✓	✗
[17]	+45k	✗	✗	✗	✓	✓	✗	✗	✗	✗
[15]	+45k	✗	✗	✓	✓	✗	✗	✗	✗	✗
[21]	+6M	✗	✗	✓	✓	✗	✓	✗	✓	✗
[11]	+35k	✗	✗	✗	✓	✓	✓	✓	✓	✗
[6]	N/A	✗	✗	✗	✓	✓	✓	✗	✗	✗
[7]	200k	✓	✓	✓	✓	✗	✓	✗	✗	✗
[13]	+250k	✓	✓	✓	✓	✓	✗	✗	✓	✗
[16]	Unlimited	✓	✗	✓	✓	✓	✓	✓	✗	✗
[3]	Unlimited	✓	✗	✓	✓	✓	✗	✓	✗	✓
[18]	N/A	✗	✗	✗	✗	✓	✓	✗	✗	✗
[10]	14k	✓	✓	✓	✗	✓	✓	✓	✗	✗
[23]	+30k	✓	✓	✓	✗	✓	✓	✗	✗	✗
[22]	+90k	✓	✗	✓	✗	✓	✓	✓	✗	✗
[8]	N/A	✓	✓	✗	✗	✓	✓	✗	✗	✗
[20]	60k	✓	✓	✓	✓	✓	✓	✗	✗	✓
[19]	Unlimited	✓	✗	✗	✗	✓	✓	✗	✗	✗
Ours	*Unlimited*	✓	✓	✓	✓	✓	✓	✓	✗	✓

pan-tilt-zoom cameras pose an important challenge related to the camera viewpoints that most of the published data sets do not address. Furthermore, expanding a data set by applying the same techniques of acquisition/annotation may be prohibitively very high in terms of cost and time. From the above, our main contribution is a software tool to design and automatically generate multi-purpose synthetic object data sets from multiple camera viewpoints and environmental conditions. The data contains features for 3D context learning and geometric computer vision problems such as depth estimation, pose estimation, 3D box estimation, 3D reconstruction, camera calibration, and also a pixel-perfect ground truth for scene understanding problems such as scene-level and instance-level semantic segmentation, and object detection (see Table 1).

Different of other tools [8, 19] and simulators [3, 16], our approach is implemented to give the final user flexibility on easily adjust normalized parameters (*e.g.* pose, lighting, blurring, backgrounds, output type, viewpoints, and number of rendered images) while improving the interoperability with other data sets. Thus, all outputs can be automatically annotated and stored in two optional JavaScript Object Notation (JSON)-style notations: Native style or MSCOCO [9] style. By applying a parameterized procedure to automatically generate images, we face the practical difficulty of gathering thousands of real examples at the same time that guaranteeing a reasonable level of variability of object poses. This kind of data is well-suited for traffic monitoring scenarios, where the camera can move in several directions, capturing uncountable poses to be detected and recognized. Code and data sets are available in https://github.com/IvisionLab/traffic-analysis/tree/master/synthetic-dataset-generator.

3 Building data sets



1. Place 3D models on Blender scene

2. Set parameters for animation

3. Execute rendering

Fig. 2 Overview of our data set generation approach. The first stage is the 3D model selection and placing in the scene. The second stage is the discretization parameter setting for the animation process. The last stage is to run the rendering for automatic labeling.

The pipeline for the data rendering process is depicted in Fig. 2, consisting of three basic steps: place the models in the scene, set the discretization parameters, and execute the rendering script. To accomplish it, a virtual environment was designed in Blender[1] and some Python scripts were integrated for parameters setting, in the same way as other research works of similar purposes [11, 21].

Components of the virtual scene: any 2D or 3D object allowed by Blender can be imported into the scene. The scenario can be modeled at the user's disposition, for instance by adding multiple lighting sources (*e.g.* sun, lamps, spots, hemis), new image processing parameters (*e.g.* blurring, edge highlighting, shadows) or full 3D environments (*e.g.* neighborhoods, rooms). The blender scene already contains some flat objects and one 3D environment of a neighborhood, with the objective of contextualizing the objects in random and fixed backgrounds. Furthermore, a node was created to apply a blending step "smoothen" on the RGB image before rendering, just as [4]. Extending new nodes graphically is simple and directly alters the output files of the rendered images.

Placing the models: if perfect poses annotation is a concern, so all the objects must be roughly aligned to have orientation 0 (front face facing to the camera), and each model must be centered on the point (0,0) of the scene from its geometric centroid.

Setting parameters: the viewpoint is parameterized as a tuple (α, φ, d) of camera rotation parameters, where α is the azimuth, φ the elevation, and d the distance to the camera. Three principal parameters of discretization should be defined

[1] Free open-source 3D software https://www.blender.org

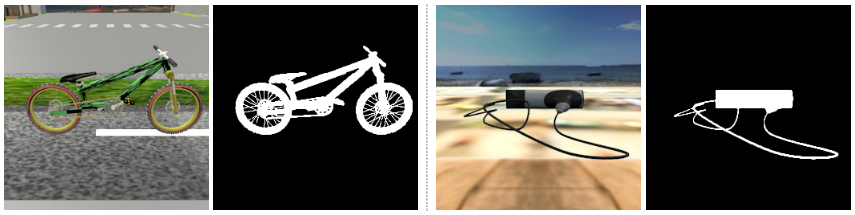


Fig. 3 Examples of rendered images and segmentation masks of the Bike poses and Hairdryer poses data sets.

to create the animation of the camera: the range of distances $D(d)$, the range of elevations $E(\varphi)$, and the range of azimuth rotation $A(\alpha)$, with $[\alpha, \varphi, d] \in \mathbb{R}$ (see Fig. 2(2)). These ranges are split into discretized steps. Thereby, the number of rendered frames directly depends on set the parameters chosen, and the added number of 3D models of the category to be built (target objects) as $NumberOfFrames = NumberOfModels * NumberOfPoses$, where the number of poses results of the multiplication of the of the three arrays cardinality, which conforms the viewpoint tuple, as $NumberOfPoses = |D(d)| * |E(\varphi)| * |A(\alpha)|$.

Execute rendering: here is when the poses are conceived by animating a sphere who drives the camera over several 3D positions according to α, φ and d values. Each *pose* constitutes a key combination of these three parameters. Then, the 3D target models are stored in OBJ format. From these models, the coordinates of their 3D boxes in (x, y, z) are extracted directly from the 3D environment virtually calibrated. The eight vertices come for designing a tight 3D box faces fitted around each 3D model in a standardized manner. For each of the poses obtained from the animation, the following is performed:

- Calibration matrices: intrinsic and extrinsic calibration matrices are computed from the scene. First, the 3x3 matrix K called the camera matrix, and finally the extrinsic pose parameters $R|t$, which both compose the calibration 4x4 matrix P calculated by multiplying the camera matrix K by the Rotation R matrix and the Translation t vector, like this: $P = K * [R|t]$. Any 3D point ev of the calibrated scene can be transformed accurately to the pixel space by multiplying it by the camera matrix, as follows: $projectedPoint = P * ev$.
- 3D Scene Layout: all 3D boxes are labeled vertex to vertex, following the same labeling pattern. In the same way, the faces of the 3D boxes are labeled as Top, Bottom, Lef, Right, Back or Front. These 3D positions correspond to the feature's location in the image after backprojection onto the object geometry.
- Render images: at this moment, all the necessary parameters for the production of the data set are defined, hence the rendering process is ordered. Here, RGB images, binary masks and depth maps are stored in folders. From the binary segmentation masks, the 2D boxes that circumscribe the target objects, and the segmentation polygons are extracted following the MSCOCO style. As aforementioned, the tool allows to generate data sets in native format, one proposed by us, or in MS COCO data set format. The structure of the files and JSON files is different for each case.

Table 2 Comparison of the five data sets generated with our tool.

Parameter	Car poses	Bike poses	Chair poses	Boat poses	Hairdryer poses
Azimuth	0-360°in 5°steps	0-340°in 20°steps	0-170°in 10°steps	0-300°in 10°steps	0-180°in 20°steps
Elevation	0-90°in 10°steps	0-20°in 20°steps	0-80°in 10°steps	0-60°in 10°steps	10-30°in 20°steps
Distance	5, 7 [meters]	4, 6, 8 [meters]	4, 5 [meters]	7, 8, 9 [meters]	2, 3, 4 [meters]
# Poses	2,736	108	324	651	60
# 3D target Models	100	6	6	10	6
# RGB images	273,600	648	1,944	6,510	360

Putting in test our generator: five data sets are generated for demonstration: the *Car poses* with thousands of camera viewpoints from 100 full-annotated vehicles. The other four data sets are similar for the categories: bike, chair, boat, and hairdryer. The *Car poses* is the biggest with a hundred models of cars and vans of different models, additionally annotated with real dimensions (length, height, and width) obtained online from car manufacturers and some pre-annotated from [12]. For consistency in the size of objects, a length of 1.8m was manually set for the 3D cars as in [12]. Table 2 summarizes the main characteristics of the data sets created, while Fig. 3 illustrates examples of some of their rendered images, in which the fine-grained labeling is noteworthy. All of these five data sets were rendered in approximately 13 hours using an Intel Core i7 quad-core CPU @ 2.8 GHz accelerated by Nvidia GeForce GTX 1070, and a RAM of 12GB.

4 Discussion and conclusion

It was proposed a generator tool of synthetic data sets whose characteristics are well suited for fine-tuning CNNs, or pre-training or training from scratch. Our generator allows for building a large labeled set of multi-viewpoint rendered images to facilitate user specific experiments. As little human effort is involved in this process, it can scale very well.

The use of a modeling software can better deal with inherent difficulties on annotation than traditional ways using real images, once the modeling process already embeds it in a more precise manner. For instance, there is no divergence on labeling each object model or even annotate all its pixels, because the modeling process already demands the designer to name each 3D object model and its boundary. Hence, all elements of the virtual environment are accurate annotated and labeled before the rendering process. Consequently, there is a significant reduction of time and costs compared to conventional annotation over thousands of real images by hand-craft processes.

When comparing our approach with the most similar [8, 19, 21, 22], the ours has a larger number of characteristics that makes it more applicable to different scenarios. It is vital to recognize that, although our work contains several annotations than others miss, some others works are better fitted for the specific scenarios for which they were carefully designed, such as the Virtual KITTI [5] and Falling Things [20] data sets which contain more hyper-realistic scenarios for autonomous vehicles applications.

In the same way, the simulators as CARLA [3] and AirSim [16] provide hiper-realistic interactive environments well suited for many computer vision problems, although considerable knowledge of game engines and computer graphics is already needed for extending new features such as 3D box annotations or camera viewpoints categorization. According to the literature review, there are many other works using rendered images for computer vision applications, nevertheless, most of them do not release the code for creating or extending the data. Adding animation to the scene, optical flow annotations, and test the data sets on a diversity of applications, will be the future direction of this work.

References

[1] Atapour-Abarghouei, A., Breckon, T.P.: Real-time monocular depth estimation using synthetic data with domain adaptation via image style transfer. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2800–2810 (2018)

[2] Butler, D.J., Wulff, J., Stanley, G.B., Black, M.J.: A naturalistic open source movie for optical flow evaluation. In: A. Fitzgibbon et al. (Eds.) (ed.) European Conf. on Computer Vision (ECCV), Part IV, LNCS 7577, pp. 611–625. Springer-Verlag (2012)

[3] Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V.: CARLA: An open urban driving simulator. In: Proceedings of the 1st Annual Conference on Robot Learning, pp. 1–16 (2017)

[4] Dwibedi, D., Misra, I., Hebert, M.: Cut, paste and learn: Surprisingly easy synthesis for instance detection. In: The IEEE International Conference on Computer Vision (ICCV), pp. 1310–1319 (2017)

[5] Gaidon, A., Wang, Q., Cabon, Y., Vig, E.: Virtualworlds as proxy for multi-object tracking analysis. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) pp. 4340–4349 (2016)

[6] Handa, A., Whelan, T., McDonald, J., Davison, A.: A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM. IEEE International Conference on Robotics and Automation (ICRA) pp. 1524–1531 (2014)

[7] Johnson-Roberson, M., Barto, C., Mehta, R., Sridhar, S.N., Rosaen, K., Vasudevan, R.: Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks? IEEE International Conference on Robotics and Automation (ICRA) pp. 746–753 (2017)

[8] Liebelt, J., Schmid, C., Schertler, K.: Viewpoint-independent object class detection using 3d feature maps. In: 2008 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8 (2008). DOI 10.1109/CVPR.2008.4587614

[9] Lin, T., Maire, M., Belongie, S.J., Bourdev, L.D., Girshick, R.B., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: common objects in context. CoRR **abs/1405.0312**, 740–755 (2014)

[10] Matzen, K., Snavely, N.: Nyc3dcars: A dataset of 3d vehicles in geographic context. In: International Conference on Computer Vision (ICCV), pp. 761–768 (2013)

[11] Mayer, N., Ilg, E., Hausser, P., Fischer, P., Cremers, D., Dosovitskiy, A., Brox, T.: A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4040–4048 (2016)

[12] Pepik, B., Stark, M., Gehler, P., Schiele, B.: Teaching 3d geometry to deformable part models. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 3362–3369 (2012). DOI 10.1109/CVPR.2012.6248075

[13] Richter, S.R., Hayder, Z., Koltun, V.: Playing for benchmarks. International Conference on Computer Vision (ICCV) pp. 2232–2241 (2017)

[14] 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: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3234–3243 (2016). DOI 10.1109/CVPR.2016.352

[15] Savva, M., Chang, A.X., Dosovitskiy, A., Funkhouser, T., Koltun, V.: MINOS: Multimodal indoor simulator for navigation in complex environments. arXiv:1712.03931 **abs/1712.03931** (2017)

[16] Shah, S., Dey, D., Lovett, C., Kapoor, A.: Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In: Field and Service Robotics (2017)

[17] Song, S., Yu, F., Zeng, A., Chang, A.X., Savva, M., Funkhouser, T.: Semantic scene completion from a single depth image. IEEE Conference on Computer Vision and Pattern Recognition pp. 190–198 (2017)

[18] Su, H., Qi, C.R., Li, Y., Guibas, L.J.: Render for cnn: Viewpoint estimation in images using cnns trained with rendered 3d model views. In: 2015 IEEE International Conference on Computer Vision (ICCV), pp. 2686–2694 (2015). DOI 10.1109/ICCV.2015.308

[19] Sun, B., Peng, X., Saenko, K.: Generating large scale image datasets from 3d cad models. In: CVPR 2015 Workshop on the future of datasets in vision (2015)

[20] 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, pp. 2038–2041 (2018)

[21] Varol, G., Romero, J., Martin, X., Mahmood, N., Black, M.J., Laptev, I., Schmid, C.: Learning from synthetic humans. Conference on Computer Vision and Pattern Recognition (CVPR) pp. 4627–4635 (2017)

[22] Xiang, Y., Kim, W., Chen, W., Ji, J., Choy, C., Su, H., Mottaghi, R., Guibas, L., Savarese, S.: Objectnet3d: A large scale database for 3d object recognition. In: European Conference Computer Vision (ECCV), pp. 160–176 (2016)

[23] Xiang, Y., Mottaghi, R., Savarese, S.: Beyond pascal: A benchmark for 3d object detection in the wild. In: IEEE Winter Conference on Applications of Computer Vision, pp. 75–82 (2014). DOI 10.1109/WACV.2014.6836101