




BlenderProc2: A Procedural Pipeline for Photorealistic Rendering

Maximilian Denninger ^{1,2}, Dominik Winkelbauer ^{1,2}, Martin Sundermeyer ^{1,2}, Wout Boerdijk ^{1,2}, Markus Knauer ¹, Klaus H. Strobl ¹, Matthias Humt ^{1,2}, and Rudolph Triebel ^{1,2}

¹ German Aerospace Center (DLR) ² Technical University of Munich (TUM)

DOI: [10.21105/joss.04901](https://doi.org/10.21105/joss.04901)

Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Kevin M. Moerman](#)  

Reviewers:

- [@nicoguardo](#)
- [@natevm](#)
- [@SelvamArul](#)

Submitted: 21 September 2022

Published: 20 February 2023

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

Summary

BlenderProc2 is a procedural pipeline that can render realistic images for the training of neural networks. Our pipeline can be employed in various use cases, including segmentation, depth, normal and pose estimation, and many others. A key feature of our Blender extension is the simple-to-use python API, designed to be easily extendable. Furthermore, many public datasets, such as 3D FRONT ([Fu et al., 2021](#)) or Shapenet ([Chang et al., 2015](#)), are already supported, making it easier to clutter synthetic scenes with additional objects.

Statement of need

Deep learning thrives on the existence of vast and diverse datasets. Collecting those in the real world is often either too complicated or expensive. With BlenderProc2, we present a tool enabling the generation of vast and diverse datasets with a few lines of Python code. A particular focus is placed on the acknowledgment of the simulation-to-real gap and how to tackle this particular challenge in the dataset generation process. Even though the first version of BlenderProc was one of the first tools to generate photo-realistic, synthetic datasets, many more tools exist nowadays, compared in [Table 1](#) ([Greff et al., 2022](#); [Manolis Savva* et al., 2019](#); [Morrical et al., 2021](#); [Schwarz & Behnke, 2020](#); [To et al., 2018](#)). In contrast to the first version of BlenderProc, BlenderProc2 relies on an easy-to-use python API, whereas the first version used a YAML-based configuration approach ([Denninger et al., 2019, 2020](#)).

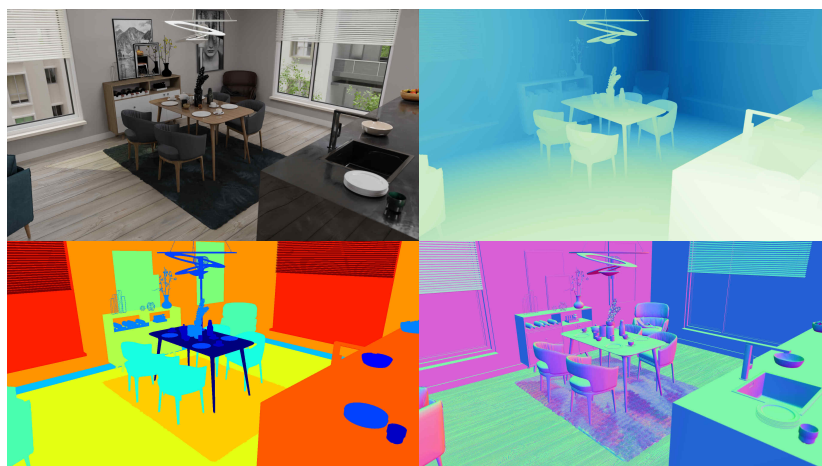


Figure 1: A rendered color, distance, semantic segmentation, and surface normal image from a synthetic scene.

	NDDS	NViSII	Habitat	Stilleben	Kubric	Ours
semantic segm.	✓	✓	✓	✓	✓	✓
depth rendering	✓	✓	✓	✓	✓	✓
optical flow	✗	✓	✓	✗	✓	✓
surface normals	✗	✓	✓	✓	✗	✓
object pose	✓	✓	✓	✓	✗	✓
bounding box	✓	✓	✓	✗	✗	✓
physics module	✓	✓	✓	✓	✓	✓
camera sampling	✓	✓	✓	✓	✓	✓
GUI-based debugging viewer	✓	✗	✗	✗	✗	✓
uses an open-source renderer	✓	✓	✗	✓	✓	✓
real-time	✓	✗	✓	✓	✗	✗

Table 1: Main features present or not present in different simulators

There are two groups of visual data generators. The first group is focused on speed and can generate dozens of images per second by typically relying on game engines to produce their images. These game engines, however, focus on producing an image that can trick the human mind into believing that a scene is real, which is not the same as generating a real image. In contrast to that, in the second group, we focus on the realism of the final images instead of on their generation speed. This realism is achieved by using a path tracer that follows the path of light beams from a light source to the camera. Physical material properties then determine how the light interacts with the 3D scene and appears in the image.

The most significant advantage of BlenderProc2 is its large toolbox, as it provides tools to set, for example, the intrinsic parameters of a camera (including its lens distortion) or to import a complete URDF model to specify a robot. Further, it is possible to construct random rooms and automatically drop or place objects from the BOP datasets in them, e.g. allowing the training of networks to succeed in the task of 6D pose estimation. It is also possible to emulate an active stereo sensor with a random or designed pattern of structured light or to sample random items or surfaces within an existing dataset, where BlenderProc2 provides tools to extract the correct surface per object category. Finally, we do not only support the rendering of color, depth, distance, surface normals, and semantic segmentation. BlenderProc2 is also capable of rendering optical flow and normalized object coordinates (NOCS) and then save the data either in the hdf5 container or in the BOP or COCO formats.

Acknowledgements

We thank the many people who helped make this a successful open-source project.

References

- Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., & Yu, F. (2015). *ShapeNet: An information-rich 3D model repository* (arXiv:1512.03012 [cs.GR]). Stanford University — Princeton University — Toyota Technological Institute at Chicago. <https://doi.org/10.48550/ARXIV.1512.03012>
- Denninger, M., Sundermeyer, M., Winkelbauer, D., Olefir, D., Hodan, T., Zidan, Y., Elbadrawy, M., Knauer, M., Katam, H., & Lodhi, A. (2020). Blenderproc: Reducing the reality gap with photorealistic rendering. *International Conference on Robotics: Science and Systems, RSS 2020*.
- Denninger, M., Sundermeyer, M., Winkelbauer, D., Zidan, Y., Olefir, D., Elbadrawy, M., Lodhi, A., & Katam, H. (2019). BlenderProc. *arXiv Preprint arXiv:1911.01911*. <https://doi.org/10.48550/ARXIV.1911.01911>
- Fu, H., Cai, B., Gao, L., Zhang, L.-X., Wang, J., Li, C., Zeng, Q., Sun, C., Jia, R., Zhao, B., & others. (2021). 3D-FRONT: 3D furnished rooms with layouts and semantics. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 10933–10942. <https://doi.org/10.48550/ARXIV.2011.09127>
- Greff, K., Belletti, F., Beyer, L., Doersch, C., Du, Y., Duckworth, D., Fleet, D. J., Gnanaprasam, D., Golemo, F., Herrmann, C., Kipf, T., Kundu, A., Lagun, D., Laradji, I., Liu, H.-T. (Derek), Meyer, H., Miao, Y., Nowrouzezahrai, D., Oztireli, C., ... Tagliasacchi, A. (2022). *Kubric: A scalable dataset generator*. <https://doi.org/10.48550/ARXIV.2203.03570>
- Manolis Savva*, Abhishek Kadian*, Oleksandr Maksymets*, Zhao, Y., Wijmans, E., Jain, B., Straub, J., Liu, J., Koltun, V., Malik, J., Parikh, D., & Batra, D. (2019). Habitat: A Platform for Embodied AI Research. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. <https://doi.org/10.1109/ICCV.2019.00943>
- Morrill, N., Tremblay, J., Lin, Y., Tyree, S., Birchfield, S., Pascucci, V., & Wald, I. (2021). *NViSII: A scriptable tool for photorealistic image generation*. <https://arxiv.org/abs/2105.13962>
- Schwarz, M., & Behnke, S. (2020). *Stilleben: Realistic scene synthesis for deep learning in robotics*. <https://doi.org/10.48550/ARXIV.2005.05659>
- To, T., Tremblay, J., McKay, D., Yamaguchi, Y., Leung, K., Balanon, A., Cheng, J., Hodge, W., & Birchfield, S. (2018). *NDDS: NVIDIA deep learning dataset synthesizer*.