The Journal of Open Source Software

scene_synthesizer: A Python Library for Procedural Scene Generation in Robot Manipulation

Clemens Eppner [1¶], **Adithyavairavan Murali**[1], **Caelan Garrett** [1], **Rowland O'Flaherty**[1], **Tucker Hermans** [1], **Wei Yang** [1], **and Dieter Fox** [1]

1 NVIDIA Research ¶ Corresponding author

DOI: 10.21105/joss.07561

Software

- Review
- Repository
- Archive

Editor: Chris Vernon

Reviewers:

- @AlexanderFabisch
- @Mechazo11

Submitted: 21 November 2024
Published: 28 January 2025

Summary

`scene_synthesizer` is a library for writing procedural scene generators in Python with a special focus on robot manipulation. The resulting scenes can be exported to various formats, enabling physics simulation or rendering data pipelines for training robotics or vision models.

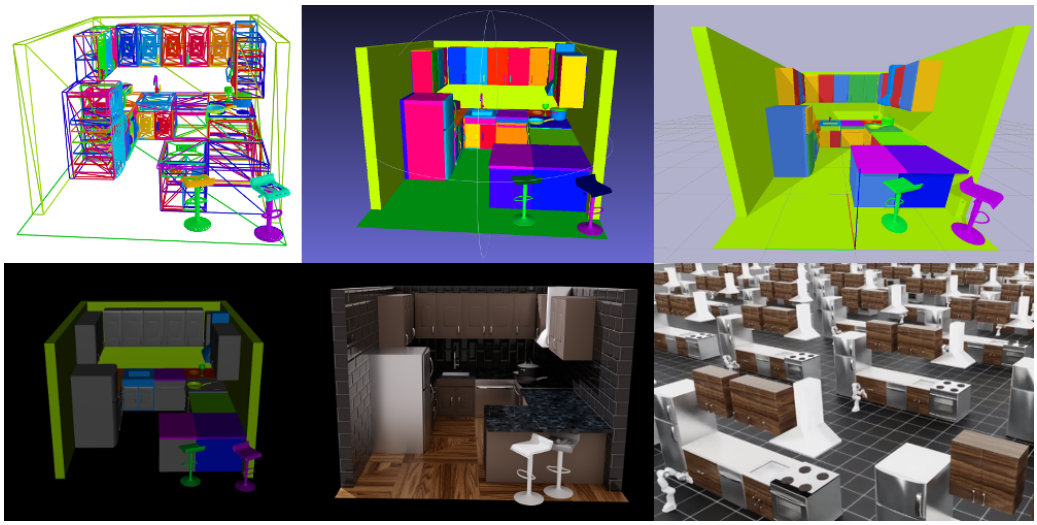

Figure 1: A synthetic kitchen scene. From top to bottom, left to right: Shown in the debug viewer, exported as mesh file in MeshLab, simulated in pybullet, Mujoco, Isaac Sim, and Isaac Lab.

Statement of Need

Simulation is an ever increasing data source for training deep learning models. In robotics, simulations have been successfully used to learn behaviors such as navigation, walking, flying or manipulation. The value of data generation in simulation mainly depends on the diversity and scale of scene layouts. Existing datasets (Ehsani et al., 2021; Garcia-Garcia et al., 2019; Mo et al., 2019; Nasiriany et al., 2024) are limited in that regard, whereas purely generative models still lack the ability to create scenes that can be used in physics simulator (Höllein et al., 2023; Schult et al., 2024; Yang et al., 2024). Other procedural pipelines either focus on learning visual models (Denninger et al., 2023; Greff et al., 2022; Raistrick et al., 2023), address specific use-cases such as autonomous driving (Fremont et al., 2020; Hess et al., 2021), or make it hard to be extended and customized since they are tightly integrated with a particular simulation platform (Deitke et al., 2022). With `scene_synthesizer` we present a

library that simplifies the process of writing scene randomizers in Python, with a particular focus on physics simulations for robot manipulation. It is fully simulator-agnostic.

Features & Functionality

`scene_synthesizer` leverages `trimesh` (Dawson-Haggerty et al., n.d.) for its scene representation, enabling access to a wide range of existing geometric algorithms. Assets can be either loaded from files (supporting all standard mesh formats, including those for articulated structures like USD, URDF, and MJCF) or instantiated procedurally from a library of 30 predefined objects, most of which are kitchen-themed. Asset placement is facilitated by defining object-agnostic anchor points or through automated labeling of support surfaces and containment volumes. The software allows users to specify physical properties such as collision geometry, mass, density, center of mass, friction, and joint constraints, including parameters like stiffness, damping, limits, maximum efforts, and velocities. Scenes can be fully articulated, with six common kitchen layouts provided as examples. While many included procedural assets and layouts are tailored to the kitchen domain, the software is versatile and does not impose constraints on the type of scenes that can be generated. The synthesized scenes can be exported to formats like USD and URDF, making them compatible with various physics simulators. `scene_synthesizer` has few dependencies and is easily extendable and customizable.

Example Use Cases

We have used `scene_synthesizer` to train neural robot motion planners (Fishman et al., 2022), neural collision checkers (Murali et al., 2023), pick-and-place policies (Yuan et al., 2023), visuomotor policies (Dalal et al., 2023), to fine-tune Vision-Language Models (Yuan et al., 2024), and in planning-based data generation pipelines (Garrett et al., 2024). In all these examples, the data was generated in simulation using procedurally created scenes featuring object-filled shelves, tables, microwaves, countertops, cabinets, drawers, etc.

Acknowledgements

We thank Jan Czarnowski for providing code during his internship which found its way into this project. We thank Adam Fishman, Ankur Handa, Shangru Li, Fabio Ramos, and Arsalan Mousavian for valuable feedback during the development of this project. We thank Melanie Miulli for selecting MDL materials.

References

Dalal, M., Mandlekar, A., Garrett, C., Handa, A., Salakhutdinov, R., & Fox, D. (2023). Imitating task and motion planning with visuomotor transformers. *Conference on Robot Learning*.

Dawson-Haggerty et al. (n.d.). *Trimesh* (Version 3.2.0). https://trimesh.org/

Deitke, M., VanderBilt, E., Herrasti, A., Weihs, L., Salvador, J., Ehsani, K., Han, W., Kolve, E., Farhadi, A., Kembhavi, A., & Mottaghi, R. (2022). ProcTHOR: Large-Scale Embodied AI Using Procedural Generation. *Advances in Neural Information Processing Systems (NeurIPS)*.

Denninger, M., Winkelbauer, D., Sundermeyer, M., Boerdijk, W., Knauer, M., Strobl, K. H., Humt, M., & Triebel, R. (2023). BlenderProc2: A procedural pipeline for photorealistic rendering. *Journal of Open Source Software*, *8*(82), 4901. https://doi.org/10.21105/joss.04901

Ehsani, K., Han, W., Herrasti, A., VanderBilt, E., Weihs, L., Kolve, E., Kembhavi, A., & Mottaghi, R. (2021). ManipulaTHOR: A framework for visual object manipulation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. https://doi.org/10.1109/cvpr46437.2021.00447

Fishman, A., Murali, A., Eppner, C., Peele, B., Boots, B., & Fox, D. (2022). *Motion policy networks*. https://arxiv.org/abs/2210.12209

Fremont, D. J., Kim, E., Dreossi, T., Ghosh, S., Yue, X., Sangiovanni-Vincentelli, A. L., & Seshia, S. A. (2020). Scenic: A language for scenario specification and data generation. *CoRR*, *abs/2010.06580*. https://arxiv.org/abs/2010.06580

Garcia-Garcia, A., Martinez-Gonzalez, P., Oprea, S., Castro-Vargas, J. A., Orts-Escolano, S., Garcia-Rodriguez, J., & Jover-Alvarez, A. (2019). *The RobotriX: An eXtremely photorealistic and very-large-scale indoor dataset of sequences with robot trajectories and interactions*. https://doi.org/10.1109/iros.2018.8594495

Garrett, C. R., Ramos, F. T., Akinola, I., Degirmenci, A., Eppner, C., Fox, D., Hermans, T. R., Mandlekar, A. U., Mousavian, A., Narang, Y. S., O'Flaherty, R. W., Sundaralingam, B., & Yang, W. (2024). *Techniques for training machine learning models using robot simulation data* (Patent No. US20240338598A1). https://patentimages.storage.googleapis.com/98/5d/bb/6ac5fcbb5c0745/US20240338598A1.pdf

Greff, K., Belletti, F., Beyer, L., Doersch, C., Du, Y., Duckworth, D., Fleet, D. J., Gnanapragasam, 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. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. https://doi.org/10.1109/cvpr52688.2022.00373

Hess, T., Mundt, M., Pliushch, I., & Ramesh, V. (2021). A procedural world generation framework for systematic evaluation of continual learning. *arXiv Preprint arXiv:2106.02585*.

Höllein, L., Cao, A., Owens, A., Johnson, J., & Nießner, M. (2023). Text2Room: Extracting textured 3D meshes from 2D text-to-image models. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 7909–7920. https://doi.org/10.1109/iccv51070.2023.00727

Mo, K., Zhu, S., Chang, A. X., Yi, L., Tripathi, S., Guibas, L. J., & Su, H. (2019). PartNet: A large-scale benchmark for fine-grained and hierarchical part-level 3D object understanding. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. https://doi.org/10.1109/cvpr.2019.00100

Murali, A., Mousavian, A., Eppner, C., Fishman, A., & Fox, D. (2023, May). CabiNet: Scaling neural collision detection for object rearrangement with procedural scene generation. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. https://doi.org/10.1109/icra48891.2023.10161528

Nasiriany, S., Maddukuri, A., Zhang, L., Parikh, A., Lo, A., Joshi, A., Mandlekar, A., & Zhu, Y. (2024). RoboCasa: Large-scale simulation of everyday tasks for generalist robots. *Robotics: Science and Systems*.

Raistrick, A., Lipson, L., Ma, Z., Mei, L., Wang, M., Zuo, Y., Kayan, K., Wen, H., Han, B., Wang, Y., Newell, A., Law, H., Goyal, A., Yang, K., & Deng, J. (2023). Infinite photorealistic worlds using procedural generation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 12630–12641. https://doi.org/10.1109/cvpr52729.2023.01215

Schult, J., Tsai, S., Höllein, L., Wu, B., Wang, J., Ma, C.-Y., Li, K., Wang, X., Wimbauer, F., He, Z., Zhang, P., Leibe, B., Vajda, P., & Hou, J. (2024). ControlRoom3D: Room generation using semantic proxy rooms. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. https://doi.org/10.1109/cvpr52733.2024.00593

Yang, Y., Jia, B., Zhi, P., & Huang, S. (2024). *PhyScene: Physically interactable 3D scene synthesis for embodied AI*. https://doi.org/10.1109/cvpr52733.2024.01539

Yuan, W., Duan, J., Blukis, V., Pumacay, W., Krishna, R., Murali, A., Mousavian, A., & Fox, D. (2024). *RoboPoint: A vision-language model for spatial affordance prediction for robotics*. https://arxiv.org/abs/2406.10721

Yuan, W., Murali, A., Mousavian, A., & Fox, D. (2023). *M2T2: Multi-task masked transformer for object-centric pick and place*. https://arxiv.org/abs/2311.00926