Creating Dynamic Simulation Environments With Unreal Engine 5

Daisy Abbott, Anjali Nuggehalli, Francisco Morales Puente, Chau Vu, Ella Zhu, Anthony J. Clark
Computer Science Department
Pomona College
Claremont, California, USA
anthony.clark@pomona.edu

***Abstract*—Simulation is a vital component for many machine learning-based systems. In this abstract, we present our work using Unreal Engine 5 to create realistic environments for training a neural network used in the navigation system of a mobile robot. We explore the use of randomized textures to create dynamic environments, and we evaluate trained models in environments with both randomly changing and static textures.**

***Index Terms*—simulation, machine learning, robotics**

I. Visions

We are building a simulation environment to find and synthesize both mundane and exceptional environments. Autonomous systems should easily handle known, mundane scenarios (e.g., navigating a wide hallway with markers and no dynamic, external actors), but they should also operate safely in exceptional scenarios (e.g., in the presence of many people behaving in unexpected ways, wearing unusual clothing, and making unusual sounds). We are building a simulation system in which we can design data generation algorithms for both of these cases and everything in between. We will find the scenarios that are unhandled by our systems and use them to train updated models.

Synthetic data will play a major part in the future of robotics. Reconstruction techniques such as photogrammetry [1] and NeRF [2], AI-generated content (AIGC) like stable diffusion [3], and more conventional procedural generation algorithms will be combined to generate data that is both realistic and challenging. Models trained with these environments will have a better chance of surviving the transition from a virtual space into the real world.

II. Project Summary

Autonomous systems are going to have a ubiquitous presence in our day to day lives. Developers of these systems are increasingly reliant upon machine learning models to help make decisions. This is particularly true for systems operating in unstructured, real-world environments where it is difficult to hand-design a response to every possible interaction. Since machine learning models are only as effective as the data on which they are trained, roboticists have turned to creating datasets in virtual environments as they are safer, more dynamic, cost-effective, easier to manipulate, and do not result in damaged robots [4].

Using simulation comes at the cost of efficacy—models created using synthetic data perform differently in simulation and reality. In order to bridge the gap between simulation and reality, we explore a variety of possibilities in our hyper-realistic environments created using Unreal Engine 5 (UE5). These features include varied lighting, textures, and dynamic environments. In this abstract we focus on varied textures.

Figure 1 shows our UE5 environment along with the corresponding real-world building. We modeled the building using Blender, but we are also exploring the use of photogrammetric models, which can also be imported into UE5, and NeRF. Similar to domain randomization [5], the simulation enables us to dynamically change wall, ceiling, and floor textures, and will enable us to change lighting conditions and add dynamic actors into the scene.

We created several agents to navigate the simulation environment. Different agents follow different paths and collect data with different characteristics. For example, a "perfect" navigator will follow a path down the center of each hallway and turn 90 degrees at each corner, whereas a "wandering" navigator will follow a similar path but occasionally take random actions causing it to deviate from the perfect path and collect data from more varied viewpoints. Our initial datasets include images captured from the view of a camera mounted on a small wheeled robot, as well as information about the *correct* action to take at each step. Correct actions are computed using a map of the environment that is not available to the robot during the training process.

We generate data by sending commands to UE5 from an external script using the open sound control (OSC) protocol, which is built into UE5. This networked setup enables quick iteration and experimentation with different data generation patterns. We collected four datasets for the experiments presented in this abstract by paring: (1) perfect navigation and static textures, (2) perfect navigation and randomized textures, (3) wandering navigation and static textures, and (4) wandering navigation and randomized textures. Datasets (2) and (4) have textures randomly changed after every 20 actions executed by a navigator. Datasets are relatively small comprising only a few thousand images each.

For results presented here, we train a ResNet-18 [6] model using fastai [7]. Models are trained to navigate the robot from one point in the simulation environment to another.

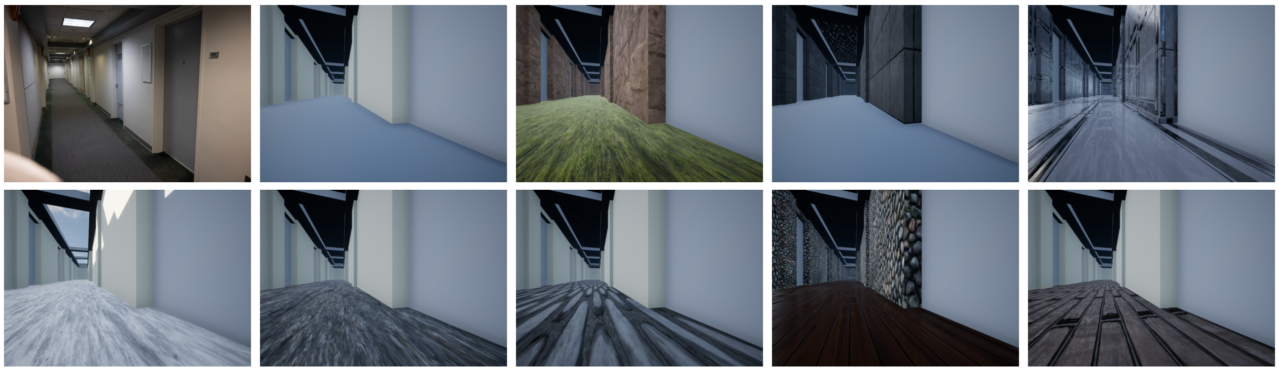
Fig. 1: (Top-Left) Photograph of a hallway in Oldenborg Hall on Pomona College's campus. All other images depict the same hallway in simulation with different randomly selected textures.

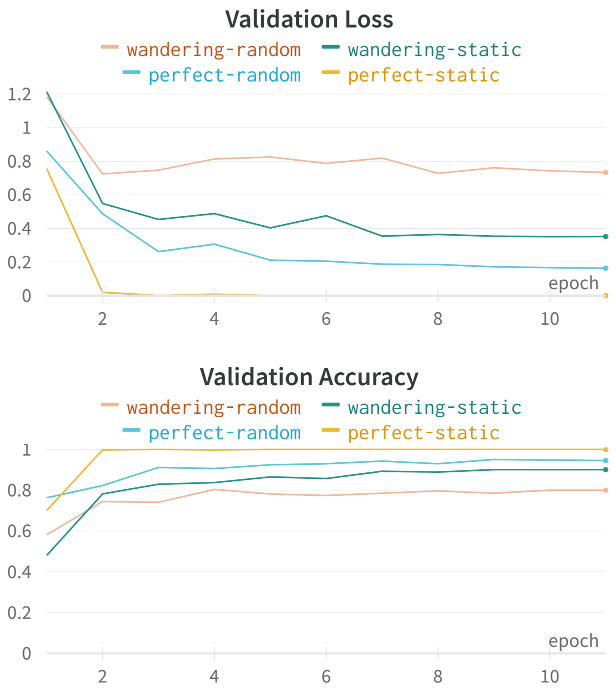

Fig. 2: Validation accuracy and loss for models trained using different datasets.

We explored using both pretrained and non-pretrained models and, unsurprisingly given the small dataset size, found that pretraining drastically improves performance. Models take images as input and output the next action for the robot to take (move forward or rotate left/right). We also explored custom architectures that include additional information as input, such as the output of the previous step (see our previous work for more information [8]). Figure 2 shows the accuracy and loss as calculated on a validation dataset comprising 20% of the complete dataset.

Model accuracy and loss are generally good indicators of performance [9], however, it is important to also evaluate trained model performance on the navigation task. Our evaluation indicates that dataset (1) created using perfect navigator and static textures achieves the best performance in the static environment and the worst performance in the dynamic environment. On the other hand, while models trained using dataset (4) perform particularly poorly in static environments, they do have the best performance in the environment with textures that change randomly. Although using a pretrained network improves performance, it is important for our next experiments to include larger datasets with more variability—both in textures and in the viewpoints represented in the captured images.

The work presented in this abstract will continue primarily in two directions. First, we will explore additional randomized features, such as lighting and dynamic actors. Second, we will evaluate trained models in the real world on our prototype robot.

III. Acknowledgements

The authors would like to thank Elizabeth Johnson and Christiana Marchese for their contributions to this work.

References

[1] W. Guerra, E. Tal, V. Murali, G. Ryou, and S. Karaman, "FlightGoggles: Photorealistic sensor simulation for perception-driven robotics using photogrammetry and virtual reality," *arXiv:1905.11377 [cs]*, 2019.

[2] F. Remondino, A. Karami, Z. Yan, G. Mazzacca, S. Rigon, and R. Qin, "A critical analysis of NeRF-based 3d reconstruction," *Remote Sensing*, vol. 15, no. 14, p. 3585, 2023.

[3] Y. Cao, S. Li, Y. Liu, Z. Yan, Y. Dai, P. S. Yu, and L. Sun, "A comprehensive survey of AI-generated content (AIGC): A history of generative AI from GAN to ChatGPT," 2023.

[4] F. Xia, A. R. Zamir, Z. He, A. Sax, J. Malik, and S. Savarese, "Gibson env: Real-world perception for embodied agents," *arXiv: Artificial Intelligence*, pp. 9068–9079, 2018.

[5] 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.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition."

[7] J. Howard and S. Gugger, "fastai: A layered API for deep learning," *Information*, vol. 11, no. 2, p. 108, 2020.

[8] E. Johnson, S. Heck, K. G. Hernandez, and A. J. Clark, "Searching for problematic simulation conditions," in *Southern California Robotics Symposium*, 2022.

[9] O. Chang, C. Marchese, J. Mejia, and A. J. Clark, "Investigating neural network architectures, techniques, and datasets for autonomous navigation in simulation," in *IEEE Symposium Series on Computational Intelligence*. IEEE, 2021.