Towards Imaginative Robots: A Generative Pipeline for Simulated Environments

Christopher May*, Lorenz Suchy, Jörg Franke, Sebastian Reitelshöfer

Institute for Factory Automation and Production Systems (FAPS), Friedrich-Alexander-Universität Erlangen – Nürnberg, Egerlandstr. 7-9, 91058 Erlangen, Germany; *christopher.may@faps.fau.de

SNE 35(1), 2025, 61-70, DOI: 10.11128/sne.35.tn.10720
Selected ASIM SST 2024 Postconf. Publication: 2024-12-10
Received Improved: 2025-03-07; Accepted: 2025-03-09
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna
ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. Autonomous Mobile Systems (AMS) offer significant advantages for industry and private sectors by adapting to diverse and dynamic environments. To train these systems, large amounts of data are required, typically obtained from simulated environments. However, the creation of these environments is often labor-intensive. Here, we propose a generative pipeline that provides a streamlined approach to virtual training and testing while allowing users to apply automated methods including generative AI.

Our pipeline consists of four, partly iterative main steps. The pipeline spans from the creation of individual assets to the utilization of the simulated environments. The pipeline is then implemented for an exemplary scenario, utilizing multiple methods including generative AI.

Furthermore, we propose a novel application of our pipeline to provide robots with the capabilities to “imagine” virtual experiences based on anticipated tasks.

The presented pipeline not only simplifies the process of generating simulated environments, but also resembles a scalable framework for developing increasingly complex AMS.

Introduction

Mobile robots, and in particular Autonomous Mobile Systems (AMS), are transforming the world. While transport robots are already well-established in industry, they have not yet reached their peak. In the coming years companies will expand their fleets and applications with new systems, increasingly powered by AI. [1]

Developing and training AI models requires massive amounts of data to ensure performance in the demanding, large, dynamic, and diverse operating environments of AMS. One solution to reduce the effort associated with collection and annotation of the required data is simulation. In simulated environments, the possibilities to generate synthetic data are virtually unlimited. However, generating data for all kinds of imaginable scenarios, is still related with large human efforts. Recent breakthroughs in generative AI could enable developers of AMS to reduce the needed effort while improving the quality of synthetic data from robotics simulation.

Most of the existing work on the generation of simulated robot training environments focuses on reinforcement learning in small manipulation scenarios [2, 3]. Those do not lie within the scope of our work. One notable exception has been presented by Bonetto et al. Their approach focuses on “Generating Realistic Animated Dynamic Environments for Robotics Research”, abbreviated “GRADE” [4]. GRADE requires an existing set of assets. Bonetto et al. have, among others, proven that synthetic data from simulated environments can be sufficient to train and validate vision-based robots [5, 6]. Another related approach that utilizes generative AI has been presented in the position paper “Towards Generalist Robots: A Promising Paradigm via Generative Simulation” [7]. Their related work “RoboGen” [3] focuses on motion planning for stationary robots. Xian et al. define the term 'generative simulation'. Their concept is supposed to generate scenes with accompanying robot tasks and at the same time include training supervision. Although the authors discuss multiple ideas and claim to be able to generate infinite data for various robots in various environments, at the time of writing this paper, the work of Xian et al. remains primarily a literature review without actual implementation. While an interesting physics simulator, GENESIS is currently missing the promised aspects of ‘generative simulation’. [8]

In this paper, we first identify methods that are relevant in the context of simulated environments for AMS. We then present a generative pipeline for the creation of simulated environments for AMS. The pipeline consists of four main steps, which are partially iterative. In this modular approach, different methods can be employed in different steps of the pipeline. This applies to both conventional methods and generative AI-methods. We furthermore present an exemplary implementation of our pipeline that utilizes several of the methods discussed to create simulated environments, fully populated with AI-generated assets. Finally, we introduce the concept of *imaginative robots* and propose the application of our pipeline to enable robots to prepare for new and unknown situations autonomously.

1 Methods for the Creation of Simulated Environments

Before defining a pipeline for creating simulated environments, it is important to clarify the relevant methods. We identify four general methods that are relevant to simulated operating environments. These methods can incorporate existing models, databases, etc. Although these methods can be used in conjunction with each other, for the purpose of this discussion, we treat them as isolated from one another. We limit ourselves to a rather general evaluation, which is intended to provide general guidance. The presented methods may yield different results when specific approaches are evaluated. In this paper we focus on static, unarticulated environments. Customizability of assets and environments is still a relevant aspect for specific scenarios and with articulated models in mind for future work.

1.1 Manual Methods

The first and most obvious class of methods is manual methods. This classification includes all approaches where substantial work is done manually using tools such as Blender and Autodesk Maya [9, 10]. Although manual methods can utilize these tools, they do not involve automation. Users have control and may modify every aspect of their workpiece to fit within the requirements, as long as it is supported by the tools utilized.

While manual methods can produce high-quality handcrafted results, the trade-off is that they are largely time consuming. Therefore, they are not suitable for large-scale simulated environments.

1.2 Automated Reconstruction Methods

Due to the time-intensity of manual methods, the application of automated methods is attractive. A class of automated methods are methods for automated reconstruction. They are proven to be suitable for efficiently reconstructing larger scale outdoor but also indoor environments. [11]

Automated reconstruction approaches are often implemented as photogrammetric methods based on RGB data, but might also incorporate depth data. The gathered data is then combined into photorealistic 3D models that accurately represent their real-world counterpart. [11, 12]

A significant disadvantage of automated reconstruction methods is limited modifiability of the generated models. This hinders the application of photogrammetric methods in the context of generating new data for training and validation of AMS. Possible applications include the reconstruction of individual assets or the reconstruction of empty “base” environments that can be populated later on.

1.3 Procedural Methods

Methods for automated reconstruction cannot create new environments and therefore might be helpful in some aspects, but not to tackle the core problem of new and diverse data. Manual methods can build upon human imagination to create new content - however strongly impeded by the necessary manual labor. Hence, we will now shift towards methods that are able to create entirely new assets and environments with minimal human intervention.

Procedural methods generate content algorithmically within predefined constraints, without the need for manual input after an initial setup. These methods can produce a vast amount of diverse and complex data automatically, both in a deterministic manner but also by incorporating random elements. The absence of a manual input apart from the initial setup is a core feature of those procedural methods.

Procedural methods are well established in computer games, where they are used to generate expansive virtual worlds, such as in commonly known Minecraft. They also find application in robotics simulation: NVIDIA Omniverse includes a "Domain Randomizer", able to alter multiple parameters of a simulated scene randomly [13]. Further procedural approaches in robotics simulation include Cropcraft [14] for generating simulated crop fields or the already mentioned GRADE [4]. [15, 16]

1.4 Generative AI-based Methods

The next class of relevant methods is based on generative AI. Similar to procedural methods, generative AI-methods are able to computationally generate new content. Unlike procedural methods, they are generally not constrained to algorithmically predefined content.

There are several popular approaches to implementing generative AI, such as Generative Adversarial Networks (GANs), Variational Autoencoder (VAEs) or Transformer Models [17–19]. The latter might be the most publicly known type of model for being the basis of LLMs like ChatGPT.

Another relevant approach involves diffusion models. Diffusion models start with random noise and iteratively refine it into a detailed output, guided by a prompt. Inspired by the physical diffusion process, these models reverse noise addition, leveraging conditioning information – like the provided prompt – to shape the noisy base towards the desired content. This approach enables the generation of high-quality outputs. [20, 21]

A further notable approach are Neural Radiance Fields (NeRFs). NeRFs synthesize 3D scenes from 2D images by using deep neural networks to gain a volumetric representation of a scene. They are able to generate high quality scenes, but at the cost of computational inefficiency. [22]

1.5 Summary of Relevant Methods

All of the methods discussed in this chapter are relevant and usable for creating simulated environments. However, each of them has specific advantages and disadvantages. Users have to choose a fitting method based on their specific needs. To summarize the findings of this chapter and to ease the decision-making process, Table 1 provides a generalized comparison of the methods mentioned.

All methods are compared in five categories and rated from -- (worst) up to ++ (best):

- Human Effort involved, less is better
- Quality of results assets
- Customizability of assets for specific requirements, e.g. rigged objects
- Hardware requirements imposed by the method; lower requirements are rated better
- Originality, meaning the capability to generate new content

	Manual	Reconstruction	Procedural	Gen-AI
Effort	--	0	+	++
Quality	++	+	+	-
Customizability	++	-	-	0
Hardware requirements	0	-	-	--
Originality	++	--	0	+

Table 1: The four discussed methods for creating simulated environments are compared in regards to effort, quality, customizability, hardware requirements and originality.

2 Introduction of the Generative Pipeline

In the following we introduce a pipeline which enables its users to create, compose and harness simulated environments. All methods compared in the previous chapter can be applied throughout the pipeline. They may also be combined and different approaches might be used in different steps.

The pipeline shown in Figure 1 consists of four steps, which are explained in a generic manner in this chapter. An exemplary implementation is described in the following chapter.

Figure 1: The proposed pipeline for the generation of simulated environments consists of four steps.

The foundation of every virtual environment are its individual components, which we refer to as assets. Hence the first step of the pipeline is the “Creation” step, where assets are generated. Those are 3D models of individual items, e.g., a machine or a table. They should be stored in a standardized and widely compatible format to ensure future usability.

The assets created in step one need to be classified and rated. This is done in step two, “Classification and Rating”. Depending on the method applied for creation of the assets, this step varies in complexity. The goal is to obtain a database of assets, classified at least by type and quality.

An extensive, high quality model database is crucial for a successful implementation of later steps. Users might also incorporate existing and purchasable sets, needing to keep in mind the reduced control over the assets.

Building upon the assets created and classified in the previous steps, we can proceed to the third step of “Composition”. Here the simulated environments are composed from the models in the asset database. This step can vary greatly in complexity, depending on the size and complexity of the desired operating environment of the AMS in question.

The fourth step represents the application or actual use of the simulated environment and does not lie within the scope of our work. Typical applications include the generation of synthetic data, validation of the AMS software or reinforcement learning [5, 23].

Notably, the pipeline shown in Figure 1 does not end here. Instead, an iterative process is started after the application step: The pipeline returns to the environment composition step. Here, a new simulated environment is created and then used for the desired application. This can be done over and over again.

Compared to existing domain randomization approaches in robotics simulators, an entirely new environment can be created with minimal effort. The application can thus benefit from experiences in diverse and virtually unlimited environments. This is a core component of our approach and allows users to take full advantage of the work done in the first two steps.

3 Exemplary Implementation of the generative Pipeline

For the validation of the proposed pipeline, we chose a practically relevant scenario: An electronics production environment, which is to be used for the validation of an autonomous tow truck. In the following, we present an exemplary implementation of the pipeline using various methods.

We chose to focus the application of generative AI on the first step of the pipeline.

The second step is conducted manually due to the nature of the results from the previous step. For step three we present and apply a highly adaptable procedural approach. In this publication the fourth step is limited to a qualitative evaluation of exemplary generated environments.

For implementation we chose – independently from [4] – to use the .usd-format and NVIDIA Isaac Sim as simulation software. NVIDIA Isaac Sim offers significant benefits in regards to graphics and thus evaluation of vision-based algorithms over the established Gazebo simulator [4, 24].

3.1 Creation of Assets through Generative AI

In the first step of asset creation, we apply generative AI. After applying multiple AI-models and optimizing their settings, we settled on using MV Dream and Magic3D [25, 26]. Both were used through the threestudio framework [27].

With the goal in mind of generating models that are as diverse as possible, Magic3D appears to be the better solution. Therefore, depending on the assets to be generated, one has to find a trade-off between higher quality or diverse assets. Generally, both approaches are able to generate 3D-models in usable quality as Figure 2 illustrates. The left section of the figure displays textured renderings, while the right section represents the normals of the meshes.

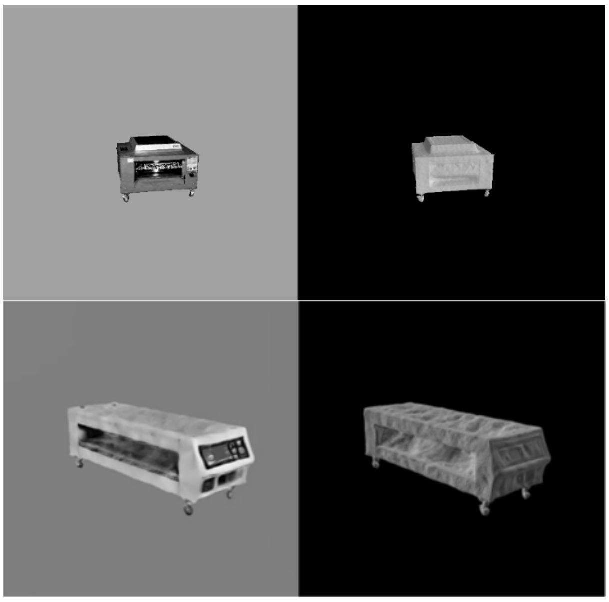

Figure 2: Both 3D models depicted are generated with the prompt “Industrial Reflow Oven”. The upper oven is created by Magic3D, the lower one by MVDream.

To ease the creation of a large number of assets, we use a script that automatically launches the AI model using a list of predefined prompts. The importance of using the right prompt when generating an asset is even more important than in 2D use cases. A prompt like "a pencil" likely won't yield a usable result. A more promising prompt would be "an upright standing pencil".

3.2 Manual Classification and Rating of Components

Due to the high hardware requirements of the AI models used in the first step, we were only able to generate a limited number of 300 assets over the course of multiple months. This low number of assets allows us to conduct the second step of the pipeline manually. It is simplified by the fact, that no classification of assets is needed due to the known prompts used for their creation.

However, the quality of the generated assets varies significantly, even within models generated with the same prompt. The models are categorized into three different categories. "Good" are all useable models, "bad" are models where the mesh or texture have significant problems and "failed" for assets where the AI completely failed. Around 30% of the models are rated "good" and thus deemed usable.

The models generated in the first step and rated "good" in this step form the basis for the next step of environment composition. Figure 3 shows a comparison of two models rated "bad" and "good", created with the same prompt.

Additional work is necessary for AI generated assets, since the AI-models we use are not aware of absolute scales. We thus have to scale and rotate the generated assets manually.

3.3 Procedural Environment Composition

For environment composition, we present a procedural approach that uses environment subdivision and provides interfaces to the methods outlined above through a modular approach. For our implementation, we rely solely on our AI-generated model database. The environment composition can be divided into three substeps which are displayed in order in Figure 4.

During layout generation, the available space is defined. A randomly sized rectangle is defined as the base for the layout.

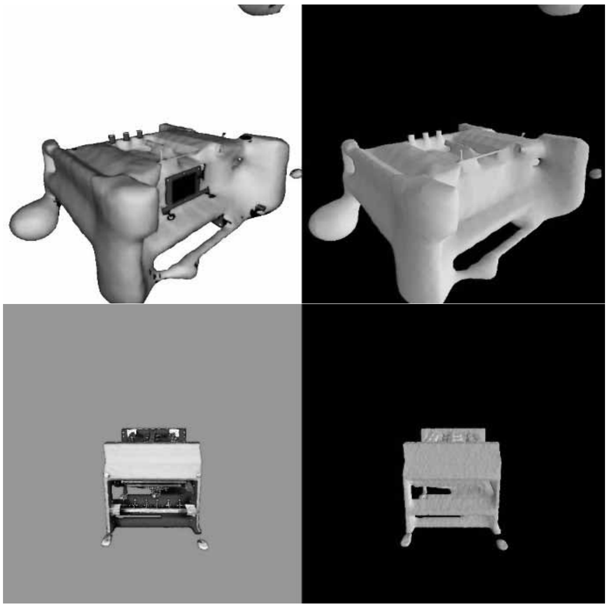

Figure 3: Even with the same prompt, the resulting assets can vary greatly in quality, as illustrated in this comparison of results from Magic3D with the prompt "Pick and Place Machine". The upper model is rated as "bad", the lower one as "good".

Figure 4: The environment composition step can be broken down into the three substeps of layout generation, definition of bounding spaces and asset placement.

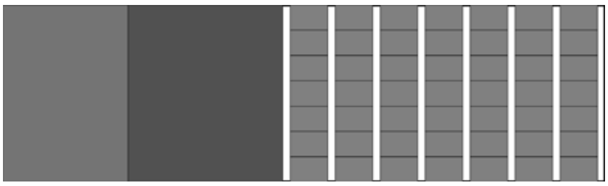

Figure 5: This exemplary procedurally generated floor layout consists of an office space (green), storage space (blue) and multiple production lines (red).

Next, the generated space is subdivided – also randomly – into the available classes of space. For our implementation, those are:

- Office space
- Storage space
- Production space

The latter is further divided into multiple production lines, depending on the size of the plant. An exemplary result of this process is shown in Figure 5.
Subsequently, the defined spaces are further partitioned into bounding spaces. They are defined by their size, position and subtype. An iterative algorithm divides the spaces defined by the layout into smaller rectangular bounding spaces. Their size is chosen randomly within predefined bounds that are dependent on the class of the space. An exception is made for the production lines: To achieve a more realistic, uniform layout, their size is only generated once for each layout and thus identical.

Each bounding space is then equipped with a procedurally generated group of assets. For this purpose, a subfunction is called for each bounding space. This function generates a fitting group of assets within the given space. In our implementation, the function is defined among others for workplaces, storage racks, and production lines. An exemplary, randomly generated production line is shown in Figure 6.

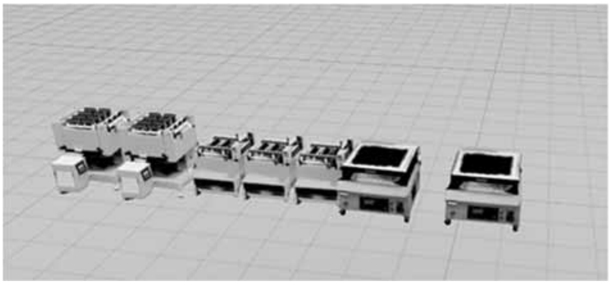

Figure 6: The depicted exemplary production line composed within step three consists of three different AI-generated machines, which are used two or three times.

For the placement of the production lines in our environments, a modification has been made: While the subfunction generally generates a new group of assets for each bounding space, this is not fitting for the production lines. In practice, a production plant often operates several identical production lines. Therefore, a number of types of production lines is randomly chosen after space partitioning. The different lines – such as the one in Figure 6 – are stored separately from the main .usd file. Instead of generating a new production line for each defined space, one is then randomly chosen from the pregenerated lines and placed within the available space including a randomized offset. By adding the modifications for the production lines to our implementation, we are both able to generate random environments and also to obtain areas where a specific structure is necessary.

3.4 Assessment of Generated Environments

In this paper we restrict the application step to a qualitative assessment of environments generated by the pipeline. An advanced application is discussed in chapter 4.2. Figure 7, Figure 8, and Figure 9 represent examples of each kind of area defined in our implementation.

From the exemplary screenshots we conclude that the presented pipeline and its implementation are suitable for the generation of simulated environments for AMS. The generated environments do not yet reach the same level of detail as handcrafted simulated environments. However, while composing an environment by hand would take hours or days, our pipeline is capable of composing environments in minutes on a standard desktop computer. We expect that advances in generative AI and further improvements to the pipeline will make it possible to generate environments and their assets with higher quality and more resource efficient in the near future.

Figure 7: This screenshot from an environment generated by our implementation of the pipeline depicts an office area composed with AI-generated workplaces. There are multiple different desks present, picked randomly from the asset database.

Figure 8: This screenshot from an environment generated by our implementation of the pipeline depicts a storage area with a number of AI-generated storage racks.

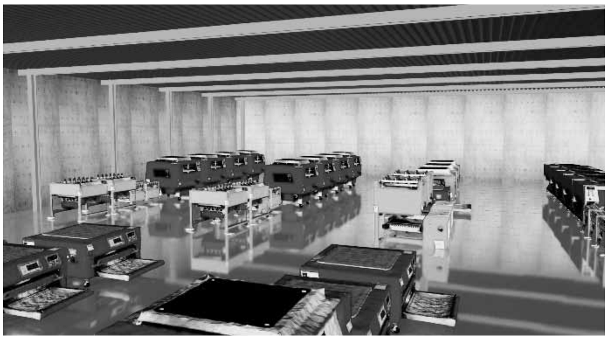

Figure 9: This screenshot from an environment generated by our implementation of the pipeline shows a production area consisting of multiple production lines with AI-generated machines. The lines on the left are identical and have been procedurally composed within step three.

4 Imaginative Robots

Imagination is a key capability that signifies advanced intelligence. Humans, along with few other species, possess the ability to foresee the outcomes of events they have not experienced by mentally simulating them. This ability to mentally simulate novel scenarios is closely linked to high cognitive flexibility and problem-solving capabilities. [28, 29] This translates to robots as well: While humans can anticipate and adapt based on imagined events, autonomous robots can hardly generalize and remain limited to explicitly programmed or learnt behaviours. Based on two well-established definitions of imagination from the Oxford English Dictionary [30] and the Oxford Dictionary of Philosophy [31], we define an *imaginative robot* as follows:

An *imaginative robot* is a robot capable of independently generating new interactive models of environments and situations that the system does not actually perceive, while combining knowledge in novel ways and anticipating possible scenarios.

We argue that building on the capabilities of the presented pipeline, *imaginative robots* could be realized and substantially improve the adaptability and flexibility of AMS. In the following, we lay out a concept and path towards *imaginative robots*.

4.1 Perception

Like all autonomous systems [32], an *imaginative robot* needs to gather knowledge about its environment through perception.

A key requirement of *imaginative robots* is a semantically rich and multimodal environment perception. While a simple Lidar-sensor might be sufficient for basic navigation tasks, modern AMS including humanoid robots need more additional information such as provided by cameras.

Meaningful information needs to be extracted and processed semantically. A key challenge is identifying relevant information and discarding irrelevant data. While modern foundation models such as Grounding-DINO [33] are in theory capable of identifying arbitrary objects, the necessary computing power and storage hinder their widespread and continuous application. [34]

4.2 Memory and Anticipation

The second key component of an *imaginative robot* is its memory. This memory goes beyond classical maps for mobile robots and introduces multiple new components.

Our concept involves building a comprehensive database of environmental information and contexts extracted from reality on the one hand, and storing the robot's capabilities and related experiences on the other. This memory will need to comprise of a combination of vector- and graph-based databases to facilitate the efficient storage and retrieval of necessary information. It also needs to rank the importance of information and allow for forgetting information, eg. by frequency of occurence as well as impact on the system. This approach forms the basis for a key capability of *imaginative robots*: Anticipation.

Based on past experiences and current information – such as sensory input or a high-level task to solve – an *imaginative robot* can make assumptions about future events and tasks. A crucial element of anticipation is to take action or prepare prior to the expected event [35]. This takes the form of thought experiments based on the pipeline presented in this paper and is the reason for calling this concept *imaginative robots*.

4.3 Thought Experiments

As a thought experiment in the context of *imaginative robots* we define the intelligent generation of simulated environments in preparation for anticipated events. Using the modular generation process described previously, the system can target specific performance or knowledge gaps by constructing scenarios tailored to the robot's needs.

To improve domain adaptation and reduce the Sim2-Real gap, memorized real-world relationships and objects are integrated directly into the simulations.

An *imaginative robot* repeatedly invokes these thought experiments to learn from them, thereby improving task performance. Rather than stopping after a fixed number of trials, the system continuously monitors key metrics to determine when additional simulations are required – a concept related to curriculum learning and active learning [36, 37].

Because a large number of experiments can be run with minimal effort, the robot can focus on areas that need improvement while also testing a wide range of possible outcomes for upcoming scenarios. Once performance in these thought experiments reaches a satisfactory level, the robot stops them and completes the anticipation process. This way, a continually learning and self-improving system could be implemented that closely resembles natural mechanisms for imagination and anticipation.

5 Discussion

The validation of the generative pipeline presented in this paper underscores the pivotal role that generative AI plays in the future development of AMS. The structured and modular approach proves to be essential as it allows for updates in step with advances in AI technology, ensuring that new and more advanced solutions can be seamlessly integrated.

The pipeline can generate diverse and virtually unlimited environments with minimal human input, although it does not fully replace human design expertise. It provides a scalable solution to the data generation challenges encountered in AMS training and validation, and enables rapid synthetic data production.

The current implementation has several important limitations. Environment generation is restricted to strictly rectangular layouts with limited room classifications, while asset placement functions must be manually coded, which limits both variety and realism in the simulated environments.

Furthermore, the manual evaluation process for generated assets, while feasible for the limited scale demonstrated, is likely to be unsustainable for larger-scale implementations.

The computational requirements also hinder wider application, as the generative AI models required approximately 40 GB of VRAM, taking two to three hours per asset on an NVIDIA RTX 6000 ADA graphics card.

The usage of generative AI thus represents significant hardware acquisition and operational costs.

Despite these limitations, the pipeline offers a solid foundation for future work. Its design enables the incorporation of newer AI models and the potential automation of asset evaluation and improvement of layout generation. Such enhancements would reduce manual intervention, improve overall quality, and expand the range of scenarios that can be simulated. The concept of *imaginative robots*, while currently in a preliminary stage, is also supported by this approach, hinting at a future where autonomous systems can generate and adapt to novel virtual experiences based on anticipated tasks.

6 Conclusion and Future Work

In this paper, we introduced a pipeline designed for generating simulated environments for AMS. This pipeline covers the entire spectrum from the creation of individual assets to the generation of complete simulated environments. It enables the rapid generation of large amounts of synthetic data, which is invaluable for robot training and validation.

Special attention was paid to advances in generative AI, which offer significant improvements over traditional methods. To validate our proposed pipeline, we implemented it and successfully generated a wide range of electronics manufacturing environments, populated by AI-generated assets. In addition, we introduced an innovative concept aimed at creating *imaginative robots*.

To exploit the full potential of our pipeline, we anticipate further developments in generative AI, which is advancing at a remarkable pace. Our ongoing efforts will focus on integrating newer AI models, such as LATTE3D or TRELLIS [38, 39]. Initial tests have already shown gains in both efficiency and quality. Additionally, we foresee the application of generative AI at various stages of the pipeline, including asset evaluation and layout generation, thereby broadening the range of scenarios and domains the pipeline can address. In addition, interactive, physically simulated objects would expand the potential applications of the generated environments.

Building on these advancements, we aim to fully realize the concept of *imaginative robots*. Currently, this is achievable to some extent, but as our pipeline evolves to generate new assets and types of environments on the go, its full potential will be unlocked.

Until then, the use of existing assets and predefined environment classifications provides a sufficient interim solution.

Acknowledgement

This publication was written as part of the research project "POV.OS – Hardware and Software Platform for Mobile Machinery", funded by the German Federal Ministry for Economic Affairs and Climate Action on the basis of a resolution of the German Bundestag.

Publication Remark.

This contribution is the improced version of the conference version published in
Tagungsband Langbeiträge ASIM SST 2024,
ARGESIM Report AR 47, ISBN ebook: 978-3-903347-65-6, pp 159-166, DOI 10.11128/arep.47.a4708

7 References

[1] Gartner. Gartner Hype Cycle Shows Supply Chain Adoption of Mobile Robots Will Far Outpace Drones Over Next Three Years. Accessed: Mar. 25, 2024. [Online]. Available: https://www.gartner.com/en/newsroom/press-releases/2023-08-17-gartner-hype-cycle-shows-supply-chain-adoption-of-mobile-robots-will-far-outpace-drones-over-next-three-years

[2] Katara P, Xian Z, Fragkiadaki K. Gen2Sim: Scaling up Robot Learning in Simulation with Generative Models. 2023, DOI 10.48550/arXiv.2310.18308

[3] Wang et al. RoboGen: Towards Unleashing Infinite Data for Automated Robot Learning via Generative Simulation. Nov. 2023, [Online]. Available: http://arxiv.org/pdf/2311.01455v2

[4] Bonetto E, Xu C, Ahmad A. GRADE: Generating Realistic Animated Dynamic Environments for Robotics Research. 2023, DOI 10.48550/arXiv.2303.04466

[5] Bonetto E, Xu C, AhmadA. Learning from synthetic data generated with GRADE. 2023. DOI 10.48550/arXiv.2305.04282

[6] Metzner M, et al. "Virtual training and commissioning of industrial bin picking systems using synthetic sensor data and simulation. *Int. Journal of Computer Integrated Manufacturing*, vol. 35, 4-5, pp. 483–492, 2022. DOI 10.1080/0951192X.2021.2004618

[7] Xian Z, Gervet T, Xu Z, Qiao YL, Wang TH, Wang Y. Towards Generalist Robots: A Promising Paradigm via Generative Simulation. May 2023, [Online]. Available: http://arxiv.org/pdf/2305.10455v3

[8] Genesis Authors. Genesis: A Universal and Generative Physics Engine for Robotics and Beyond. Accessed: Mar. 2, 2025. [Online]. Available: https://github.com/Genesis-Embodied-AI/Genesis

[9] Blender Foundation. blender.org - Home of the Blender project - Free and Open 3D Creation Software. Accessed: Mar. 7, 2025, [Online]. Available: https://www.blender.org/

[10] Autodesk. Autodesk Maya. Accessed: Mar. 7, 2025, [Online]. Available: https://www.autodesk.com/de/products/maya/overview

[11] Lieret M, Kogan V, Hofmann C, Franke J. Automated Exploration, Capture And Photogrammetric Reconstruction Of Interiors Using An Autonomous Unmanned Aircraft. In *2021 IEEE International Conference on Mechatronics and Automation (ICMA)*, Takamatsu, Japan, 2021, pp. 301–306. DOI 10.1109/ICMA52036.2021.9512707

[12] Choi S, Zhou QY, KoltunV. Robust reconstruction of indoor scenes. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2015), Boston, Massachusetts, USA, 7-12 June 2015*, Boston, MA, USA, 2015, pp. 5556–5565. DOI 10.1109/CVPR.2015.7299195

[13] NVIDIA. Domain Randomization for RL. Accessed: Mar. 11, 2024, [Online]. Available: https://docs.omniverse.nvidia.com/isaacsim/latest/isaac_gym_tutorials/tutorial_gym_domain_randomization.html

[14] GitHub. Romea/cropcraft: A Procedural World Generator for Robotics Simulation of Agricultural Tasks. Accessed: Mar. 11, 2024, [Online]. Available: https://github.com/Romea/cropcraft

[15] Brewer N. Computerized Dungeons and Randomly Generated Worlds: From Rogue to Minecraft [Scanning Our Past]. *Proc. IEEE*, vol. 105, no. 5, pp. 970–977, 2017. DOI 10.1109/JPROC.2017.2684358.

[16] Togelius J, Shaker N, Nelson MJ. Introduction. In *Procedural Content Generation in Games* (Computational synthesis and creative systems), Shaker N, Togelius J, Nelson MJ, Eds., Cham: Springer International Publishing, 2016, pp. 1–15.

[17] Goodfellow IJ, et al. Generative Adversarial Networks. Jun. 2014. Accessed: Apr. 19, 2024, [Online]. Available: http://arxiv.org/pdf/1406.2661.pdf

[18] Kingma DP, Welling M. Auto-Encoding Variational Bayes. Dec. 2013, Accessed: Apr. 19, 2024 [Online]. Available: http://arxiv.org/pdf/1312.6114

[19] Vaswani A, et al. Attention Is All You Need. Jun. 2017, Accessed: Apr. 19, 2024, [Online]. Available: http://arxiv.org/pdf/1706.03762

[20] Ho J, Jain A, P. Abbeel P. Denoising Diffusion Probabilistic Models. Jun. 2020, [Online]. Available: http://arxiv.org/pdf/2006.11239

[21] Saharia C, et al. Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. May 2022, Accessed: Apr. 19, 2024. [Online]. Available: http://arxiv.org/pdf/2205.11487

[22] Mildenhall B, Srinivasan PP, Tancik M, Barron JT, Ramamoorthi R, Ng R. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. Mar. 2020, Accessed: Apr. 19, 2024, [Online]. Available: http://arxiv.org/pdf/2003.08934

[23] B. Osinski B, et al. Simulation-Based Reinforcement Learning for Real-World Autonomous Driving. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, Paris, France, 2020, pp. 6411–6418. DOI 10.1109/ICRA40945.2020.9196730

[24] Zwingel M, May C, Kalenberg M, Franke J. Robotics simulation – A comparison of two state-of-the-art solutions. In *ASIM SST 2022 Proceedings Langbeiträge*, Breitenecker F, Deatcu C, Durak U, Körner A, Pawletta T, Eds., Jul. 2022, pp. 171–178. DOI 10.11128/arep.20.a2033

[25] Shi Y, Wang P, Ye J, Long M, Li K, Yang X. MVDream: Multi-view Diffusion for 3D Generation. Aug. 2023, Accessed: Apr. 19, 2024, [Online]. Available: http://arxiv.org/pdf/2308.16512.pdf

[26] Lin CH, et al. Magic3D: High-Resolution Text-to-3D Content Creation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[27] Y.-C. Guo YC, et al. threestudio: A unified framework for 3D content generation. Accessed: Mar. 28, 2024, [Online]. Available: https://github.com/hreestudio-project/threestudio

[28] Gilbert DT, Wilson TD. Prospection: experiencing the future. *Science (New York, N.Y.)*, vol. 317, no. 5843, pp. 1351–1354, 2007. DOI 10.1126/science.1144161

[29] Suddendorf T, Corballis MC. The evolution of foresight: What is mental time travel, and is it unique to humans?. *The Behavioral and brain sciences*, vol. 30, no. 3, 299-313; discussion 313-51, 2007. DOI 10.1017/S0140525X07001975

[30] Oxford English Dictionary. *imagination, n*. Oxford University Press, 2024.

[31] *The Oxford Dictionary of Philosophy*. Oxford University Press, 2008.

[32] Siegwart R, Nourbakhsh IR, Scaramuzza D. *Introduction to autonomous mobile robots.* 2nd ed. Intelligent robotics and autonomous agents. Cambridge, Massachusetts, London, England, MIT Press, 2011.

[33] S. Liu S et al. Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection. Mar. 2023, [Online]. Available: http://arxiv.org/pdf/2303.05499

[34] Hofmann C, May C, Ziegler P, Ghotbiravandi I, Franke J, Reitelshöfer S. Automated Individualization of Object Detectors for the Semantic Environment Perception of Mobile Robots. In *Proceedings of the 20th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications,* Porto, Portugal, 2025, pp. 851–862. DOI 10.5220/0013162500003912

[35] Oxford English Dictionary, *anticipation, n*. Oxford University Press, 2024.

[36] Settles B. Active Learning Literature Survey. Rep. TR1648, 2009, Accessed: Mar. 2, 2025, [Online]. Available: http://digital.library.wisc.edu/1793/60660

[37] Bengio Y, Louradour J, Collobert R, Weston J. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning,* Montreal Quebec Canada, Danyluk A, Bottou L, Littman M. Eds., 2009, pp. 41–48. DOI 10.1145/1553374.1553380

[38] Xiang J, et al. Structured 3D Latents for Scalable and Versatile 3D Generation. Dec. 2024, [Online]. Available: http://arxiv.org/pdf/2412.01506v1

[39] K. Xie K, et al. LATTE3D: Large-scale Amortized Text-To-Enhanced3D Synthesis. Mar. 2024, [Online]. Available: http://arxiv.org/pdf/2403.15385