Radiation Simulation and Visualization in Nuclear Power Plant Digital Twins

Youndo Do[1], Kai Tan[1], Fan Zhang[1]

[1]Georgia Institute of Technology, Atlanta, Georgia,
fan@gatech.edu

INTRODUCTION

Nuclear energy provides stable clean energy supplies and high energy density. Advanced technologies are current being developed and applied to nuclear field to improve safety, reliability, and efficiency. Robots have been proven to be feasible and efficient in performing radiation surveys in NPPs, which significantly reduces human presence in harsh environments as well as operational costs. With rapidly growing AI technology, robots can be trained facilitate nuclear power plants (NPP) inspection and maintenance. Especially in hazardous environments like high radiation areas, robots are even more powerful tools for their operation and feedback on real-time data.

The integration of robotics in NPPs presents unique challenges, particularly due to the presence of radiation. One of the fundamental issues robots face in this situation is hardware malfunction and wireless communication disruption in high radiation areas. To mitigate these risks, it is essential for robots to navigate while avoiding areas of intense radiation. While most robot training is conducted in virtual environments due to cost-effectiveness and the power of digital twin technology, it is important to accurately model radiation in these virtual simulations.

Traditional methods of radiation modeling often relied on static data and limited real-time feedback. These approaches failed to capture the full complexity of radiation fields. Radiation fields are dynamic and complex – including diverse ray patterns and intricate interactions with surrounding environments. Additionally, there are many components to consider – such as shielding materials, radiation source types, propagation media, geometric configurations, and energy levels. This major discrepancy has hindered the development of robust decision-making capabilities for robotic systems in radiation environments. It is important to model radiation fields as realistically as possible for effective robot training and risk monitoring.

In this paper, bridging the gap between theoretical radiation models and practical applications is investigated, providing a robust framework for radiation visualization that can significantly impact robot simulations and operational plans in nuclear power plants. This project incorporates Unreal Engine, a real-time 3D platform, and OpenMC, an open-source Monte Carlo simulation tool, to model and visualize radiation fields in three-dimensional, interactive formats. This pipeline presents a powerful platform where robotic systems can be simulated and analyzed with radiation effects, thereby improving their safety and performance in hazardous environments.

Related Work

Radiation Simulation

Radiation simulation is crucial in various fields – including medical, physical, nuclear safety, and scientific research, with Monte Carlo methods providing a detailed and stochastic approach to modeling radiation interactions with materials. MCNP, which is renowned for its extensive capabilities in neutron and photon transport [1] and SERPENT, popular in nuclear reactor physics and safety analysis [2], are notable Monte Carlo simulation tools. FLUKA offers a broad energy range for particle transport and interaction simulations, suitable for applications in high energy physics and medical physics [3]. Furthermore, SCALE is used for comprehensive nuclear analysis and design, featuring a blend of Monte Carlo and deterministic methods [4]. Among these tools, OpenMC stands out for its exceptional accuracy and flexibility in simulating neutron, photon, and electron transport which is highlighted by its active community and ongoing developments in computational efficiency [5]. OpenMC is generally utilized to generate different scenarios for neutron distribution systems [6]. Each tool is designed to handle complex geometries and materials, but the selection of OpenMC for this project was primarily influenced by its robust features and open-source accessibility, making it ideal for detailed radiation behavior studies.

Visualization in Digital Twin

In recent years, the use of game engines like Unreal Engine and Unity for research and educational purposes has gained significant interest. Visualizing natural phenomena is a widely studied topic across various fields. In Electrical Engineering, Wolbach et al. [7] utilized the Unity engine to observe the flow of electrical signals, which are non-observable in reality, providing a deeper understanding in fundamental electronic principles. In the geography field, Yu et al. [8] presented a platform called FLV, which integrates multiple software to combine geographic data with a game engine. Their work innovated traditional procedural building techniques, expanding them to create diverse species in forest landscapes. In the field of Marine Engineering, Nie et al. [9]

combined fluid mechanics software with Unity to simulate ocean vehicle movements, accurately modeling water dynamics and vehicle behavior. Notable visualization work has also been conducted in the nuclear field. Researchers at Oak Ridge National Laboratory [10] developed a method for visualizing radiation data using Unreal Engine. Their approach established a straightforward workflow for importing radiation data from SCALE into Unreal, allowing for interactive and immersive visualization of complex radiation fields. This work demonstrated the potential of using game engines to enhance the understanding and analysis of radiation data beyond traditional methods. In this project, open-source software for radiation data is utilized to facilitate easier access for researchers, and a different visualization technique in Unreal Engine is employed for robotic digital twins.

METHODOLOGY

In this section, we introduce two software tools used for radiation visualization. Subsequently, we will provide a detailed explanation of how the tools are integrated and utilized in our framework.

Unreal Engine

The Unreal Engine has become a key simulation tool in many industries, known for its ability to render photorealistic graphics in real-time and has established a benchmark across industries that demand precision and high-quality renderings - such as smart manufacturing, film production, and gaming industry. With this complete spectrum of abilities, from virtual reality (VR) integration to procedural generation and inverse kinematics, as well as MetaHuman technology, it is clear that Unreal Engine has applications in diverse fields. Even with its extensive abilities, Unreal Engine is designed to be user-friendly. Content can be created using the Blueprints visual scripted system, without being proficient in C++. Unreal Engine 5 is optimized to handle large file sizes, allowing users to deal with high-resolution textures (even 12K) which is crucial for next-generation visual applications.

In this work, Unreal Engine 5.3 was used for radiation visualization. The real-time rendering power of Unreal Engine allows quick visualization of the radiation emission phenomenon. The engine enables importing and visualizing large datasets.

OpenMC

OpenMC is an open-source Monte Carlo simulation tool, which demonstrates the movement of neutrons and photons through arbitrarily defined models. It provides deep analysis of particle behavior in various nuclear systems including nuclear reactors. Compared to deterministic methods, Monte Carlo methods are more suited for complex scenarios since they simulate millions or billions of particles to determine average behaviors. Since OpenMC is open source, unlike most radiation simulation tools which are commercial, it is easy for users to utilize for research and to collaborate with the scientific community. Furthermore, the tool provides a rich Python API, which makes it easy to create and manipulate models programmatically.

In our work, we used OpenMC for nuclear reactor simulation to accurately model neutron behavior and fission reactions. Proper physical modeling in OpenMC ensures that radiation visualization results are accurate and useful for reactor design and radiation shielding analysis.

Integration: Data Pipeline to Visualization

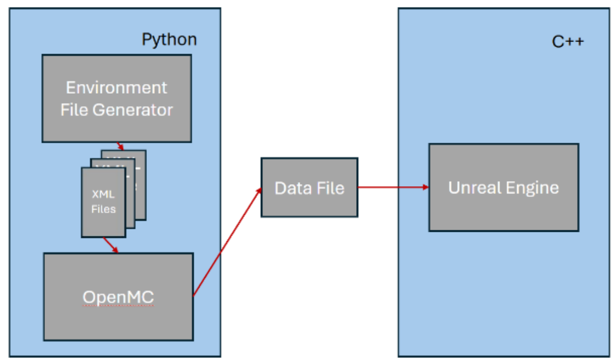

Fig. 1. Radiation Visualization Framework

This paper presents a framework to visualize radiation fields in robotics digital twin. The framework includes the three main parts: an environment file generator, OpenMC, and Unreal Engine. Figure 1 shows the overview picture of the radiation visualization framework. Initially, a map with radiation source is created in Unreal Engine. The geometry, materials, and environment settings (including radiation sources) are selected accordingly during this step.

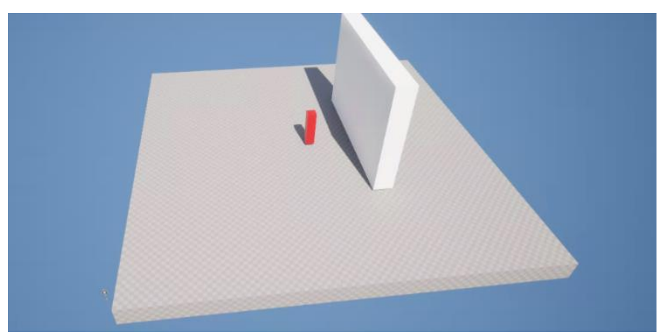

Fig. 2. An environment setup in Unreal Engine

After an initial setup is completed in Unreal Engine as illustrated in Figure 2, the environmental information must be provided to the environment file generator. This information includes the location, materials, and geometries of all the components in the map, the type of radiation sources, and strength of radiation sources. Since OpenMC requires the three files to run simulation - namely geomeries.xml, materials.xml, and settings.xml, the environment file generator creates these files with the given information. All of the files can be created with APIs provided by OpenMC. The successfully created files should be fed into OpenMC to get output data, which show the x,y,z coordinates, flux value at the pixel location, and variance value at the pixel location.

Since we collected the radiation flux data at each pixel point, the next step is to utilize Unreal Engine to visualize each point. As Greenwood et al. discussed [10], visualizing large datasets can be computationally intensive. Although the Materials feature is an efficient method for large-scale rendering, the primary purpose of visualizing radiation fields in this project is for use with robotic simulations. Therefore, while a robot is navigating a terrain, the DebuggingDraw feature is used for temporary visualization during debugging and development phases. During runtime, this DebuggingDraw feature can be turned off to avoid interference with the already computationally heavy robot simulation. When debugging logic, algorithms, or even understanding spatial relationships in nuclear power plants, this feature can help to visualize radiation quickly without worrying about performance optimization. Furthermore, this feature is well-suited for use on small datasets. To avoid computational overhead problems, we have implemented visualization in a way that only the pixels visible to the camera are rendered.

In Unreal Engine, two main functions were created in C++ files. The first function parses data from the OpenMC output text file. The second function utilizes the parsed data to visualize the radiation field for each pixel. Based on the flux data, the color of each pixel is adjusted accordingly.

The integration of Unreal Engine with OpenMC not only provides a visually intuitive representation of radiation data but also allows for dynamic interaction and manipulation within the virtual environment. In robotics simulation environments, a radiation field will be constantly changing. To mitigate the sim-to-real gap, our pipeline can be used to model a constantly changing radiation field to train robots. Furthermore, this framework is essential for radiation modeling in 3D. With large datasets, it is easier to interpret data visually. They can reveal patterns, hotspots or unexpected behaviors that might not be apparent from numerical data alone.

RESULTS

To demonstrate our framework in action, a simple scenario was created. As illustrated in Table, the map is a $100m^3$ cube. The reactor, with stainless-steel shielding, is located at the center of the map and has dimensions of 5x5x10 meters. It is assumed that there is a point source at the core of the reactor. Approximately 10 meters from the reactor in the x-direction, a concrete wall with dimensions of 5x50x30 meters is placed. Figure 3 (a) and (b) provide side-view and top view illustrations from the OpenMC setup. The purple rectangle represents the concrete wall while the red rectangle represents the reactor.

Table I. Configuration of map

Objects	Geometry (m)	Material
Entire Map	100 x 100 x100	Air
Reactor	5 x 5 x 10	Steel Wall
Wall	5 x 50 x 30	Concrete

Fig. 3. (a) Side View of OpenMC environment (b) Top View of OpenMC environment

With this setup, OpenMC provides the 3D results in Figures 4. As a concrete wall is suited for attenuating radiation rays, most radiation fields are reaching out to the left. The same behavior is observed in Unreal Engine as demonstrated in Figure 5, where yellow pixels depict higher radiation flux and purple pixels depict no radiation.

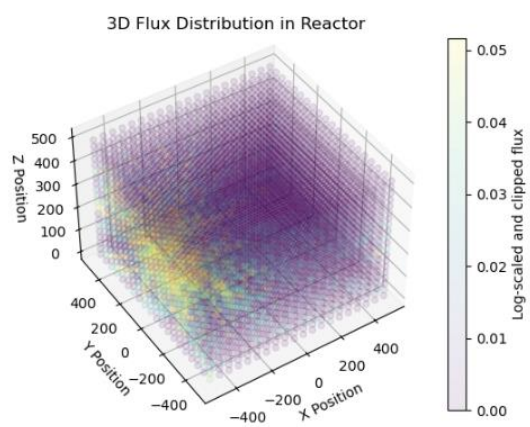

Fig. 4. OpenMC 3D flux distribution in reactor

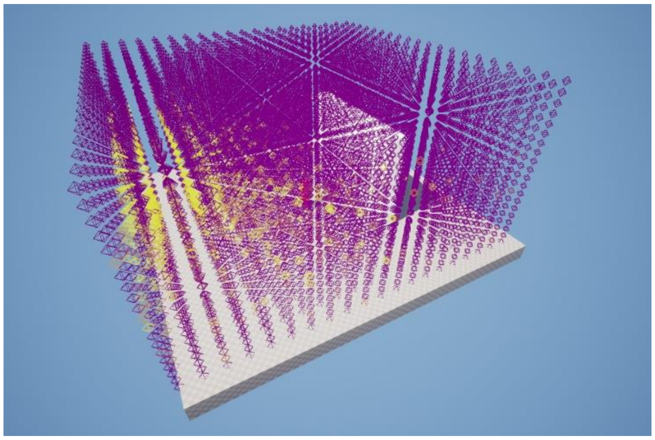
Fig. 5. Radiation Distribution in Unreal Engine

CONCLUSION

The successful integration of OpenMC and Unreal Engine opens new avenues for research and application in radiation safety and robotics. By leveraging the capabilities of these two powerful tools, this approach provides a more comprehensive and interactive method for studying radiation effects, optimizing robotic operations in radiative environments, and enhancing training and preparedness for radiation-related tasks. Future research could explore robot navigation algorithms within digital twin environments, specifically focusing on optimizing routes through low-radiation areas, enhanced exploration of nuclear power plant with radiation visualization, and management of radiation optimization using a swarm of robots.

In summary, this paper presents a novel methodology for visualizing radiation within robotic digital twins, utilizing the combined strengths of OpenMC and Unreal Engine. This integration promises to improve the accuracy, efficiency, and safety of radiation management in various applications, marking a significant advancement in radiation visualization and robotics.

REFERENCES

1. Goorley, T., et al. (2012). "Initial MCNP6 Release Overview." Nuclear Technology, 180(3), 298-315.
2. Leppänen, J., et al. (2015). "The SERPENT Monte Carlo code: Status, development and applications in 2013." Annals of Nuclear Energy, 82, 142-150.
3. Ferrari, A., et al. (2005). "FLUKA: A Multi-Particle Transport Code." CERN.
4. Rearden, B. T., et al. (2020). "SCALE Code System." Nuclear Technology, 206(4), 474-498.
5. Romano, P. K., Horelik, N. E., Herman, B. R., Nelson, A. G., Forget, B., & Smith, K. (2015). OpenMC: A state-of-the-art Monte Carlo code for research and development. Annals of Nuclear Energy, 82, 90-97.
6. Tan, Kai, and Fan Zhang. "Optimizing the Fixed Number Detector Placement for the Nuclear Reactor Core Using Reinforcement Learning." Nuclear Science and Engineering (2024): 1-23.
7. Wolbach, Elliott, Michael Hempel, and Hamid Sharif. "Leveraging VR for the Visualization of Non-Observable Electrical Circuits Principles in Engineering Education." (2024).
8. Yu, Heyuan, Mia M. Wu, and Hong S. He. "Developing platform of 3-D visualization of forest landscape." Environmental Modelling & Software 157 (2022): 105524.
9. Nie, Yunli, et al. "Design of marine virtual simulation experiment platform based on Unity3D." Global Oceans 2020: Singapore–US Gulf Coast. IEEE, 2020.
10. Greenwood, Michael Scott, and Michael B. Smith. Method for Visualizing Radiation Data using Unreal Engine. Oak Ridge National Lab.(ORNL), Oak Ridge, TN (United States), 2020.