

B2G4: A synthetic Data Pipeline for the Integration of Blender Models in Geant4 Simulation Toolkit

Angel Bueno, Felix Sattler, Maximilian Perez Prada, Maurice Stephan, and Sarah Barnes

German Aerospace Center (DLR), Institute for the Protection of Maritime Infrastructures, Germany

Corresponding author: Angel Bueno

Email address: angel.bueno@dlr.de

Abstract

The correctness and precision of particle physics simulation software, such as Geant4, are expected to yield results that closely align with real-world observations or well-established theoretical predictions. Notably, the accuracy of these simulated outcomes is contingent upon the software's capacity to encapsulate detailed attributes, including its prowess in generating or incorporating complex geometrical constructs. While the imperatives of precision and accuracy are essential in these simulations, the need to manually code highly detailed geometries emerges as a salient bottleneck in developing software-driven physics simulations. This research proposes Blender-to-Geant4 (B2G4), a modular data workflow that utilizes Blender to create 3D scenes, which can be exported as geometry input for Geant4. B2G4 offers a range of tools to streamline the creation of simulation scenes with multiple complex geometries and realistic material properties. Here, we demonstrate the use of B2G4 in a muon scattering tomography application to image the interior of a sealed steel structure. The modularity of B2G4 paves the way for the designed scenes and tools to be embedded not only in Geant4, but in other scientific applications or simulation software.

Keywords: muon tomography, Geant4 simulations, synthetic data

DOI: 10.31526/JAIS.2024.476

1. INTRODUCTION

Muon tomography, or MT, utilizes muons to enable noninvasive imaging methods to visualize macroscopic three-dimensional volumes. MT harnesses two main interaction patterns of muons to investigate a volume of interest (VOI): absorption and scattering [1]. The absorption pattern is centered around observing the attenuation of the muon flux as it penetrates the VOI and comparing this flux with the open-sky expectation. The scattering pattern, also known as muon scattering tomography (MST), focuses on quantifying the deflection angles of muons as they traverse the VOI. The deflections depend on the physical properties of the material traversed by the muons and thus serve as the bedrock for tomographic reconstruction. The canonical application of MST aims to quantify material densities in concealed or obscured setups where direct inspection would be costly or not feasible, such as border security applications [2], infrastructure monitoring [3], nondestructive industrial testing [4], or nuclear fuel cask surveillance [5].

Current procedures for MT simulation are often implemented in Geant4 [6], a powerful particle physics simulation software that accurately models particle propagation and their complex interactions with matter within a well-defined setting. Geant4 encapsulates the entire simulation in a single volume, namely, the world volume, in which the physics processes, the geometrical constructs of objects (so-called solids), detectors, and VOIs are specified. The geometrical definition of a VOI and its solids entails native function calls in C++ to the Geant4 application programming interface (API) in order to delineate the VOI and the solid properties. The advanced geometry descriptors within the Geant4 API offer a versatile implementation for constructing approximate and realistic 3D environments and, thus, permit the precise examination and analysis of muon interactions and their subsequent effects [7]. However, creating highly detailed VOIs in Geant4 can be challenging and time-consuming, as it often requires manual coding and much expertise. Therefore, established coding approaches in Geant4 fall short when it comes to efficiently designing and importing complex scenes. This limitation can hinder researchers who need rapid and smooth scene creation and exportation to validate their simulation projects.

This research proposes Blender-to-Geant4 (B2G4), a novel framework that synergizes the power of Blender's 3D modeling tools [8] to enhance scene design and simulations in Geant4. Blender offers a versatile, user-friendly interface to generate intricate geometries, leveraging the power of drag-and-drop placement, solid modeling precision, shape variance randomization, and intuitive material assignments. The suite of functionalities defined in B2G4 transplants the designed scene in Blender into a readable format for Geant4, enabling the automated export and import of scenes with minimal manual effort. The B2G4 framework streamlines the development of simulation scenes in Geant4 for a diverse range of applications. This work demonstrates an end-to-end workflow for the generation and analysis of detailed scenes in muon tomography, using Blender 4.0 and Geant4 v11.1.2.¹

¹Our code is available at github.com/DLR-MI/B2G4

2. RELATED WORK

Scene design in Geant4 with multiple solids is labor-intensive due to manual coding and iterative error rectifications. To speed up the scene design phase, several procedures have been suggested. The *Geometry Description Markup Language* (GDML) was proposed by [9]. The GDML language offers an XML-based schema designed to formulate geometry hierarchies in Geant4, emphasizing the reusability of geometric components to streamline the creation of solids in a simulation scene for Geant4. However, GDML entails manual manipulation of text files and sophisticated XML parsers for integration with Geant4, and notably, it lacks any form of visual feedback mechanism. An alternative approach presented by [10] entails transferring 3D CAD data into GDML through proprietary 3D software like FastRad. This solution introduces the necessity of proficiency in commercial 3D CAD software.

Over the past years, the research community in Geant4 has shifted toward open software solutions. Research work by [11] outlined an open-source workflow to convert STEP files to GDML, but the sophistication of the workflow requires manual coding and is prone to parsing errors across file formats. The proposed software by [12] advocates integrating 3D models into Geant4 by converting standard 3D files to Geant4 code using the TesselatedFacets feature in Geant4. Though this software is a well-established method to export 3D models into Geant4, it does not extend this capability to entire 3D scenes; that is, the geometrical attributes of each solid in a scene, their positions, and logical hierarchies in Geant4 must be manually coded one by one. Further, when adding or modifying a 3D scene, these prior workflows require compilation in Geant4.

Our approach posits a newer direction in scene design for Geant4 using Blender, an open-source 3D modeling suite that is accessible and well-documented and provides a Python interface to program external addons. Figure 1 depicts a scene design setup in Blender (left) and the exported scene in Geant4 (right) using our proposed B2G4 software. Notice that B2G4 permits the design of complete scenes with multiple solids, with precise geometry and material assignments. Further, B2G4 is open source (no proprietary tools are required) and provides enhanced visual feedback with easy scene editing. The dynamical scene exportation in B2G4 allows new scenes to be loaded during run time without the need for recompilation of Geant4, a distinction crucial in scene design. In the following sections, we describe the working principles of B2G4 and explore its application in the context of muon scattering tomography.

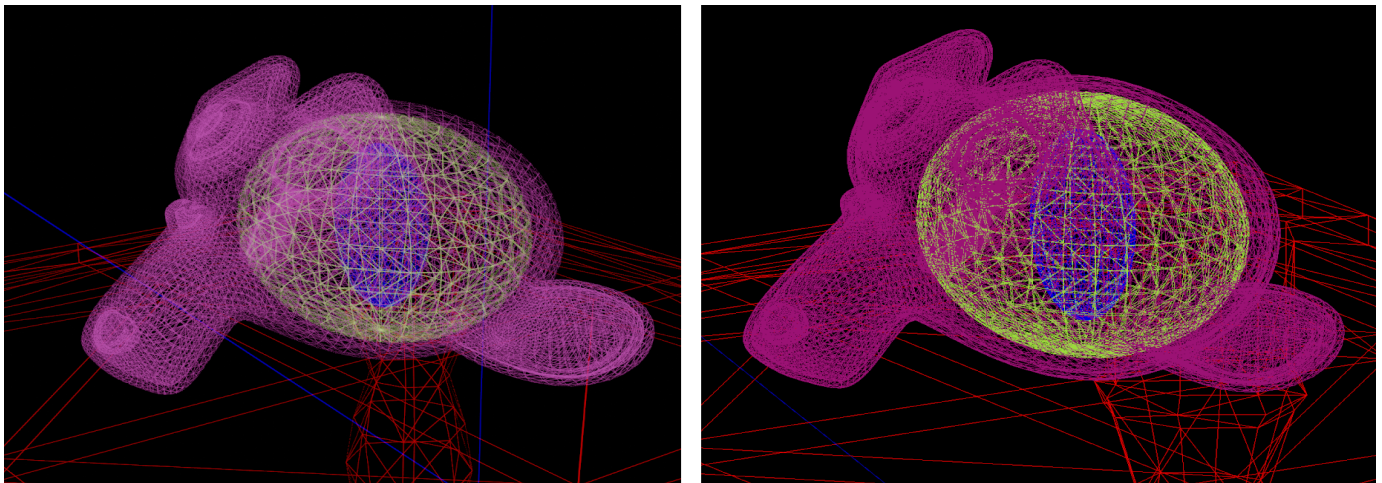


FIGURE 1: Scene setup with three solids in Blender (left) and Geant4 (right). The plastic monkey head (pink) with two ellipsoids (yellow and blue) of lead and uranium inside sits on top of a metallic surface (red) in a cuboid world (blue). The materials used are clearly distinguishable in both Blender and Geant4, demonstrating the finesse of B2G4 in solid and material definition.

3. B2G4 DESCRIPTION

The standard B2G4 workflow consists of several stages: scene creation, parsing, saving, exporting into Geant4, and data interpretation. Figure 2 illustrates the complete workflow of B2G4, adapted for muon tomography data analysis. Note that these stages are grouped into three main modules: the *B2G4 scene exporter*, the *B2G4 data interpreter*, and the *B2G4 muon tomography* module. After the data interpretation step by B2G4, the postprocessed output is forwarded to specific use cases, here, muon tomography data analysis. The implementation of B2G4 into three independent modules enables easy scaling of the scene design process, from local prototyping in Blender to the simulation of randomized scenes at scale with loadable run times in Geant4.

3.1. B2G4 Scene Exporter

The main module for scene creation and exportation is the *B2G4 scene exporter*. This standalone module contains the internal scripts for the creation and saving of the scene in Blender, and the parser to Geant4:

- (1) *Scene creation*: B2G4 follows the code conventions of Geant4 to define the geometric attributes of the object being modeled; that is, each solid object in Geant4 is created by describing its attributes with reference to a logical volume, including shape

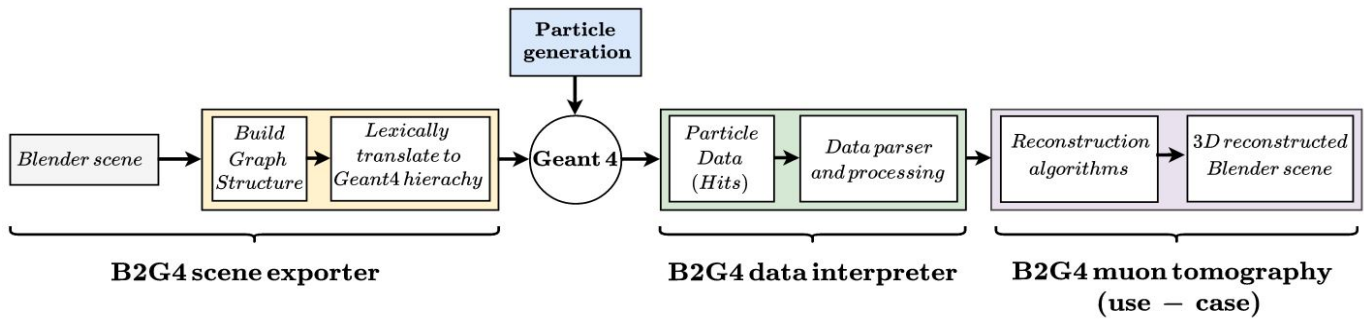


FIGURE 2: *Blender-to-Geant4 (B2G4) workflow*: B2G4 offers a range of tools to bridge Blender and Geant4, with Python scripts for Blender exports and a C++ library for Geant4 imports. The *B2G4 scene exporter* creates a graph structure that mimics the hierarchy of the Geant4 scene structures, allowing the Blender scene to be translated into a format that Geant4 can read. The *B2G4 data interpreter* processes the particle data files generated by the Geant4 simulations and extracts muon hits. The *B2G4 muon tomography* framework reconstructs the Blender scene using muon tomography algorithms from the extracted muon hits. The B2G4 modules are standalone, allowing users to employ this end-to-end workflow tailored to their specific simulation requirements in Geant4.

and physical properties. Hence, the solid design in B2G4 maintains a one-to-one correspondence with the mother-daughter logical relationship in Geant4. This logical hierarchy is implemented using the *Collection* module in Blender, a logical data container that permits the grouping of 3D solids for easier manipulation and organization of 3D scenes. The material assignment is done via the *Material Properties* modifier in Blender. The available materials in B2G4 have been ported from the Geant4 material database and include pure elements, NIST compounds, biological, space, biochemical, high-energy physics, and nuclear materials. B2G4 adopts the ionization potential to aid in material differentiation and enhance visual clarity in analysis through contrasting colors. Every material category in B2G4 has a specific color map based on the normalized ionization potential of the materials specified in the Geant4 material database. We standardize the ionization potentials of materials in a specific category by comparing them to the minimum and maximum ionization potentials. In cases where two materials have the same potential, we use their density to create a color offset for better visual separation. The normalized values are then converted to a color map. Figure 1 depicts this color scheme for different materials for a monkey head lying on a metallic surface. The surface is *G4.STAINLESS-STEEL*, whereas the monkey head is *G4.Plastic*. The monkey brain is *G4.Pb*, and the small cylindrical object inside is *G4.U*.

- (2) *Scene saving*: Once the scene has been designed, B2G4 allocates, for each *Collection* in the scene, the mesh of the imported 3D solid, the position, and the assigned material. To achieve that, a dynamic graph is built by traversing the specified *Collections* items. B2G4 defines a graph $G = (N, E)$, whose nodes N contain a *Collection* with the solid created and their attributes. The edges E in this graph preserve the hierarchical logical relationship of the mother-daughter in Geant4 with the root node, the *world volume*. This graph G serves as the basis for the exportation file required for Geant4. Next, B2G4 traverses and accesses the created graph hierarchy G within the Blender environment. For every node (or *Collection*) N within the graph hierarchy G , relevant data are extracted to outline a logical volume, establish a physical placement, and define a solid conforming to the native syntax of Geant4. After all nodes are traversed, the resulting geometry tree encapsulates the data hierarchy of Geant4 and is stored as a JSON file. This file mirrors the Geant4 layout and incorporates pointers to individual solids. Each unique solid undergoes a singular export process to minimize redundancy, being saved as a Polygon File Format (PLY). B2G4 employs the PLY format for object exportation to establish a standardized 3D asset library, given its widespread recognition and adoption as an open-source file standard for sharing 3D triangular mesh data. This approach seeks to ensure compatibility across Geant4 applications.

- (3) *Scene importing*: Here, we describe the process of importing the JSON and PLY files into Geant4. Given that the structure of the JSON file adheres to Geant4's scene hierarchy, parsing the JSON data to its corresponding Geant4 objects becomes rapid and efficient. To allow the dynamic loading of a scene during run time of Geant4, we resort to a single header file that loads and parses both, the JSON and the PLY files. This header file needs to be included only once in the standard Geant4 compilation process, and no external dependencies are required.

For each JSON entry, the required data are systematically extracted. When a reference to a solid appears, the associated PLY files are loaded and constructed with *G4TesselatedSolid*. It is important to note that every solid is created only *once* to save computational overhead, even though it can be referenced several times. The logical volumes (*G4LogicalVolume*) are created by parsing the required data from the JSON file and looking up the referenced *G4Solid* from the already loaded *G4TesselatedSolids*. The resulting *G4LogicalVolume* is stored in an array. The creation of *G4PhysicalVolume* objects resorts to a similar association of the correct transformation data from the JSON file and logical volume, based on the name identifier.

With B2G4, it becomes easy to import scenes of various scales into Geant4—from intricate designs to complex landscapes. Figure 3 depicts two examples of scene design and exportation using B2G4: a rotatory aircraft engine (Figure 3(A)) and a mountain with a golden monkey head (Figure 3(B)). Each solid in the scene is designed using the *Collection* to maintain the one-to-one correspondence with the mother-daughter logical relationship in Geant4, including the *world volume*. The scene of the rotatory

aircraft contains 1.5 million triangles, whereas the scene of the mountain scene contains 4487 triangles. Both are saved in 15.2 and 0.05 seconds, respectively. Once the scene is exported to Geant4, users can implement their custom simulation setup. The B2G4 scene exporter module is a versatile tool that can be seamlessly integrated into other Geant4 simulation pipelines. The adaptable design of the scene exporter makes it suitable for a range of scientific applications beyond muon tomography, such as medical physics, where precise simulation of particle interactions with detailed 3D models and complex scenes is crucial.

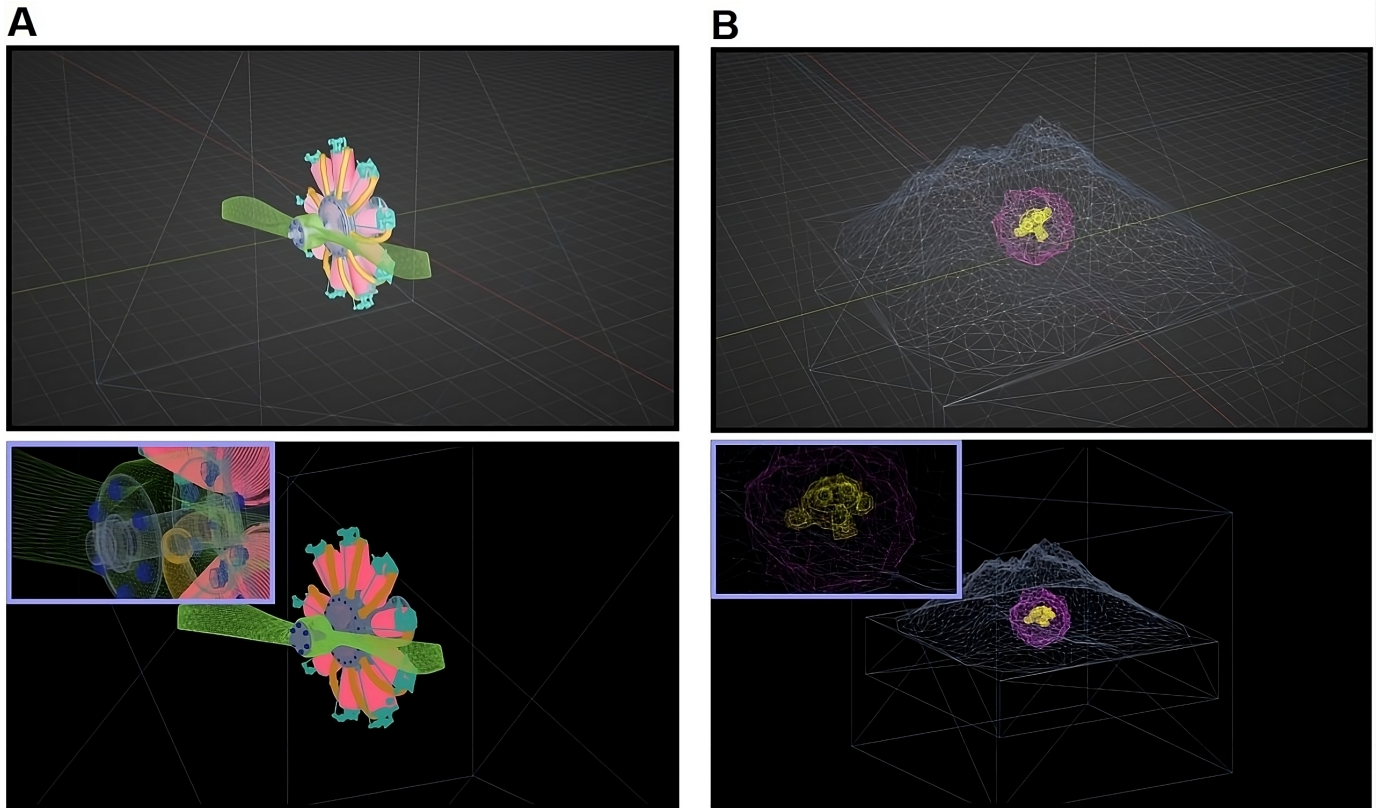


FIGURE 3: A B2G4 example of scene design and parsing from Blender into Geant4: the top portion showcases the solid designed with Blender, while the bottom portion displays its equivalent in Geant4. An inset between the two images highlights the extensive details within each solid, precisely transferred from Blender into Geant4. (A) depicts an aircraft engine with highly detailed bolts and rotatory blades (B) a mountain with a cave and a golden monkey head within. The ionization potential colormap in B2G4 eases the identification of each component in a scene.

3.2. Particle Generation with Geant4

The generation of particles is always customized to suit the Geant4 scientific application. This work uses *Cosmic-ray Shower Library* (CRY) [13], to generate cosmic ray air shower particles. The simulation data is created by recording details of the interactions of atmospheric muons with detector planes. The configuration and characteristics of sensitive detectors have to be defined within Geant4. In this context, we operate under the assumption of ideal detectors; that is, no efficiency losses or resolution effects are considered. For a more detailed review of available particle generation approaches for MT in Geant4, we refer the reader to [2].

3.3. B2G4 Data Interpreter

The B2G4 data interpreter module uses the Python extension of ROOT [14], PyROOT, to parse simulation output from Geant4 into a data structure suitable for data analysis or machine learning workflows. The ROOT software provides a tree data structure that yields faster access to massive data than regular files. PyROOT provides automatic dynamic bindings between Python and C++, making it ideal for converting the Geant4 output tree data structure into a suitable format for further data analysis.

4. MUON SCATTERING TOMOGRAPHY WITH B2G4

Muon scattering tomography (MST) estimates the deflection in the muon trajectories as they traverse the VOI. The magnitude of muon deflection, known as the scattering angle θ , supplies information about the material type and density that the muons traverse: materials with higher densities or atomic numbers yield a broader θ distribution than those with lower densities. Muon trajectories are inferred using tomographic reconstruction techniques to create a 3D scattering density map of the VOI by estimating deflection

angles. The VOI is voxelized, converting the region of interest into a grid of discrete volumetric elements or voxels, determined by the resolution required by the reconstruction algorithms. Then, for an estimated muon path, the reconstruction algorithms estimate the set of voxels most probably traversed and allocate a score based on their scattering angle. All the muon tracks are processed to create a 3D material density map rendering the internal object structure. B2G4 has a module for muon tomography, assuming MST simulations and data interpretation have been carried out. The *B2G4 muon tomography* module takes the processed muon hitmaps as input and produces the 3D reconstructed scene designed in Blender as the output. The B2G4 muon-tomography module includes two standard reconstruction algorithms for muon tomography: PoCA (point of closest approach) and ASR (angle statistics reconstruction).

The PoCA algorithm assumes a singular point of scattering for a muon and calculates its minimal separation between incident (in) and emergent trajectories (out) [15]. PoCA employs the incident-measured muon path data in terms of position \mathbf{p} and directions \mathbf{d} as $r_{\text{in}} = (\mathbf{p}_{\text{in}}, \mathbf{d}_{\text{in}})$ and $r_{\text{out}} = (\mathbf{p}_{\text{out}}, \mathbf{d}_{\text{out}})$. Therefore, to determine the exact location of the muon deflection, PoCA solves a system of equations given by the parametric equation of points on straight lines, $P = \mathbf{p} + t\mathbf{d}$, being t , the variation parameter. Formally, this system of equations can be written as follows:

$$\begin{aligned} \mathbf{P}_{\text{in}}(t_{\text{in}}) &= \mathbf{p}_{\text{in}} + t_{\text{in}}\mathbf{d}_{\text{in}}, \\ \mathbf{P}_{\text{out}}(t_{\text{out}}) &= \mathbf{p}_{\text{out}} + t_{\text{out}}\mathbf{d}_{\text{out}}, \\ \overrightarrow{\mathbf{P}_{\text{in}}\mathbf{P}_{\text{out}}} \cdot \mathbf{d}_{\text{in}} &= 0, \\ \overrightarrow{\mathbf{P}_{\text{in}}\mathbf{P}_{\text{out}}} \cdot \mathbf{d}_{\text{out}} &= 0. \end{aligned} \quad (1)$$

Once the PoCA point from both trajectories has been estimated, the muon path is constructed, and voxels crossed by this path are considered as possible candidates influencing the muon trajectory. Then, an information signal (or score) for the muon path, noted as s , proportional to the scattering angle θ , is assigned to the voxel associated with the estimated PoCA point. B2G4 applies noise reduction by rejecting extreme angles and then computes the median for each voxel to generate the 3D scattering density map for all muon tracks. While PoCA is efficient and fast, it oversimplifies the 3D representation by assuming single-point scattering and thus neglects scattering across adjacent voxels. Conversely, ASR assigns values to neighboring voxels by defining a distance D_r , between the center of a voxel c and the incident/emergent muon trajectories [16]:

$$D_r = \max \left(\min_{t_{\text{in}}} \|\mathbf{P}_{\text{in}}(t_{\text{in}}) - c\|, \min_{t_{\text{out}}} \|\mathbf{P}_{\text{out}}(t_{\text{out}}) - c\| \right), \quad (2)$$

where P_{in} and P_{out} are the parametric muon paths from equation (1). The value of D_r is compared with a threshold distance d : voxels that fulfill the condition $D_r < d$ are activated. The threshold value d ensures that at least one voxel for each incoming and emergent track pair is activated as a potential candidate affecting the muon trajectory. The resulting 3D scattering density maps in ASR incorporate contextual information from neighboring voxels, providing a more refined and robust representation compared to PoCA.

Figure 4 illustrates a simplified design of a steel cargo container (*G4_STAINLESS-STEEL*) with a leaden monkey head inside (*G4_Pb*). The container scene comprises 27,504 triangles and is exported in 0.30 seconds. Figure 4(A) showcases a texture-rich model in Blender, displaying the leaden monkey head, while Figure 4(B) depicts the Geant4 import. In Figure 4(C), a side view of the container's tomographic reconstruction in ASR (top) and PoCA (bottom) is presented. Last, Figure 4(D) offers a top-view perspective reconstructed with PoCA (left) and ASR (right). Both methods show detailed facets of the monkey head, including ears, chin, and overall shape. However, while PoCA introduces noise due to its single-point scattering assumption, ASR offers a noise-resistant reconstruction by comprehensively considering more potential voxel interactions in the muon path [17]. This makes ASR a more robust method for detailed imaging, particularly in environments with complex scattering phenomena.

5. CONCLUSIONS

Scene design in Geant4 with precise geometries poses a significant challenge, particularly given the need for manual coding of high-variance scenes. This research work introduces Blender-to-Geant4 (B2G4), a holistic workflow that integrates the 3D design capabilities of Blender with the simulation finesse of Geant4. The proposed B2G4 framework streamlines the creation of corrected and annotated 3D scenes that sidestep the traditionally time-consuming scene generation process. Although B2G4 paves the way toward synthetic data generation and fast scene design, detector creation in Blender and acceleration techniques for solid importation are left for future work. We demonstrated the capabilities of B2G4 in the context of muon tomography for imaging objects within a steel container. A specific module in B2G4 implements standard muon tomography reconstruction algorithms to convert the simulated data into meaningful 3D density maps suitable for further analysis. The scenes created in B2G4 can be adapted to other frameworks or data workflows, whether in the context of muon tomography or other scientific disciplines that require Geant4 simulations.

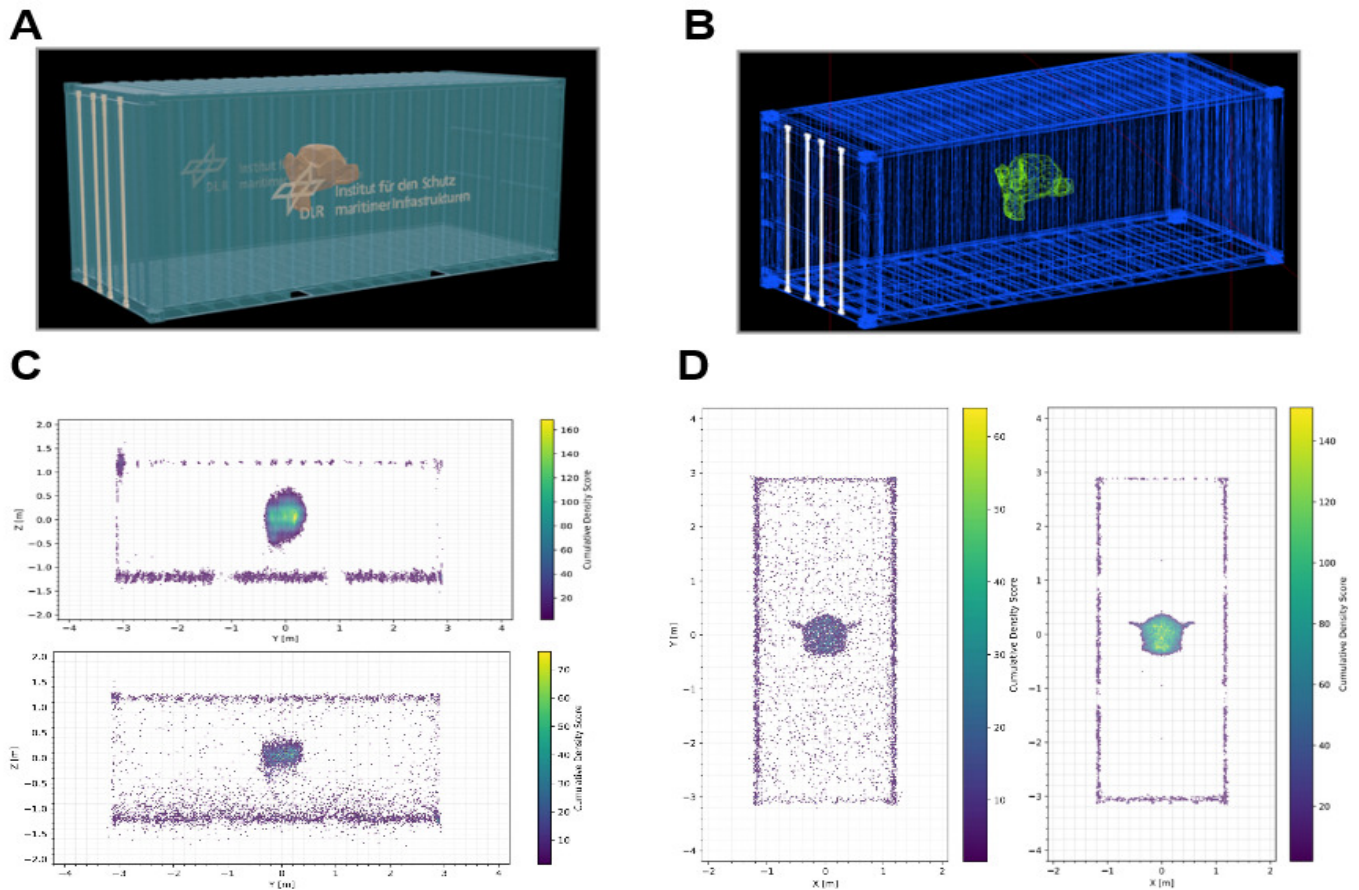


FIGURE 4: Example of muon tomography: A scene with a sealed steel container hiding a monkey head; (A) scene in Blender, with textures (B) exported scene in Geant4; (C) side view of the reconstruction in PoCA (bottom) and ASR (top); (D) top-view of the reconstruction with PoCA (left) and ASR (right).

CONFLICTS OF INTEREST

The authors declare that there are no conflicts of interest regarding the publication of this paper.

ACKNOWLEDGMENTS

This research was funded by the SilentBorder project under the grant agreement ID 101021812 of the EU Horizon 2020 Research and Innovation Programme.

References

- [1] L. Bonechi, R. D'Alessandro, and A. Giammanco. Atmospheric muons as an imaging tool. *Reviews in Physics*, 5:100038, 2020.
- [2] S. Barnes, A. Georgadze, A. Giammanco, M. Kiisk, V. A. Kudryavtsev, M. Lagrange, and O. L. Pinto. Cosmic-ray tomography for border security. *Instruments*, 7(1), 2023.
- [3] L. F. Thompson, J. P. Stowell, S. J. Fargher, C. A. Steer, K. L. Loughney, E. M. O'Sullivan, J. G. Gluyas, S. W. Blaney, and R. J. Pidcock. Muon tomography for railway tunnel imaging. *Physical Review Research*, 2(2):023017, 2020.
- [4] P. M. R. Arbol, P. G. Garcia, C. D. Gonzalez, and A. O. Alonso. Non-destructive testing of industrial equipment using muon radiography. *Philosophical Transactions of the Royal Society A*, 377(2137):20180054, 2019.
- [5] D. Poulson, J. Bacon, M. Durham, E. Guardincerri, C. L. Morris, and H. R. Trelue. Application of muon tomography to fuel cask monitoring. *Philosophical Transactions of the Royal Society A*, 377(2137):20180052, 2019.
- [6] S. Agostinelli, J. Allison, K. Amako, J. Apostolakis, H. Araujo, P. Arce, M. Asai, D. Axen, S. Banerjee, G. Barrand, et al. GEANT4—a simulation toolkit. *Nuclear instruments and methods in physics research section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 506(3):250–303, 2003.
- [7] M. Perez Prada, S. Barnes, and M. Stephan. Analysis of secondary particles as a complement to muon scattering measurements. *Instruments*, 6(4), 2022.
- [8] Blender Foundation. Blender—a 3D modelling and rendering package, 2018.

- [9] R. Chytrcek, J. McCormick, W. Pokorski, and G. Santin. Geometry description markup language for physics simulation and analysis applications. *IEEE Transactions on Nuclear Science*, 53(5):2892–2896, 2006.
- [10] M. Constantin, D. E. Constantin, P. J. Keall, A. Narula, M. Svatos, and J. Perl. Linking computer-aided design (CAD) to Geant4-based Monte Carlo simulations for precise implementation of complex treatment head geometries. *Physics in Medicine and Biology*, 55(8):N211, mar 2010.
- [11] M. Pinto and P. Gonçalves. GUIMesh: A tool to import STEP geometries into Geant4 via GDML. *Computer Physics Communications*, 239:150–156, 2019.
- [12] C. M. Poole, I. Cornelius, J. V. Trapp, and C. M. Langton. A CAD interface for GEANT4. *Australasian Physical and Engineering Sciences in Medicine*, 35(3):329–334, Sep. 2012.
- [13] C. Haggmann, D. Lange, and D. Wright. Cosmic-ray shower generator (CRY) for Monte Carlo transport codes. In *2007 IEEE Nuclear Science Symposium Conference Record*, volume 2, pages 1143–1146, 2007.
- [14] R. Brun and F. Rademakers. ROOT—An object oriented data analysis framework. *Nuclear instruments and methods in physics research section A: accelerators, spectrometers, detectors and associated equipment*, 389(1-2):81–86, 1997.
- [15] W. Zeng, M. Zeng, X. Pan, Z. Zeng, H. Ma, and J. Cheng. Principle study of image reconstruction algorithms in muon tomography. *Journal of Instrumentation*, 15(02):T02005, feb 2020.
- [16] M. Stapleton, J. Burns, S. Quillin, and C. Steer. Angle statistics reconstruction: a robust reconstruction algorithm for muon scattering tomography. *Journal of instrumentation*, 9(11):P11019, 2014.
- [17] S. Riggi, V. Antonuccio-Delogu, M. Bandieramonte, U. Becciani, A. Costa, P. La Rocca, P. Massimino, C. Petta, C. Pistagna, F. Riggi, et al. Muon tomography imaging algorithms for nuclear threat detection inside large volume containers with the muon portal detector. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 728:59–68, 2013.