Layered CAD/CSG Geometry for Spatially Complex Radiation Transport Scenarios ⋆

Elliott Biondo[a,1,*], Gregory Davidson[a,1], Brian Ade[a,2]

[a] *Oak Ridge National Laboratory, 1 Bethel Valley Rd., Oak Ridge, TN 37830 USA*

Abstract

Many spatially complex fission, fusion, and national security Monte Carlo (MC) radiation transport scenarios involve combining computer-aided design (CAD) models with constructive solid geometry (CSG) models. A *layered* geometry method has been implemented in the Shift MC code to address this need. With layered geometry, multiple CAD and/or CSG models can be clipped, translated, rotated, and placed in overlapping layers to form transport-ready geometries. The utility of this method is demonstrated with two problems: (1) a fixed-source simulation with a layered geometry consisting of a LiDAR-generated CAD model of the Combined Arms Collective Training Facility urban environment overlaid with CSG models of a mock hotel and a detector apparatus, and (2) a k-eigenvalue calculation using a layered geometry model of the Transformational Challenge Reactor consisting of CAD fuel elements placed in a CSG core. Tallied particle flux distributions match expectations, but tracking robustness must be improved prior

⋆This manuscript has been authored by UT-Battelle, LLC, under contract DE-AC05-00OR22725 with the US Department of Energy (DOE). The US government retains and the publisher, by accepting the article for publication, acknowledges that the US government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this manuscript, or allow others to do so, for US government purposes. DOE will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (`http://energy.gov/downloads/doe-public-access-plan`).

*Corresponding Author

Email addresses: `veb@ornl.gov` (Elliott Biondo), `gqe@ornl.gov` (Gregory Davidson), `ba7@ornl.gov` (Brian Ade)

[1]HPC Methods for Nuclear Applications Group, Nuclear Energy and Fuel Cycle Division

[2]Research and Test Reactor Physics Group, Nuclear Energy and Fuel Cycle Division

to general-purpose use.

Keywords: Monte Carlo radiation transport, CAD geometry, LiDAR

1. Introduction

The Monte Carlo (MC) method is the de facto standard for radiation transport analysis due to its continuous treatment of position, direction, and energy. During MC simulations, particle motion is tracked through a 3D computational geometry comprised of a collection of *cells*—closed 3D regions containing uniform material properties—representing the physical system. Production-level MC codes generally support geometries in both constructive solid geometry (CSG) and (more recently) computer-aided design (CAD) formats. CSG and CAD differ in their features, limitations, and the typical ways models are produced.

CSG formats are the primary native formats of all production-level MC codes, including Shift [1], OpenMC [2], Serpent 2 [3], and MCNP6 [4]. MC transport analysis has historically been performed on purpose-built CSG models. With CSG formats, models are constructed by combining geometric primitives using Boolean logic operations (e.g., unions, intersections, and differences). Geometric primitives are typically limited to first- and second-order surfaces (e.g., planes, cylinders, and spheres) and solids comprised of these surfaces. While typical light-water reactor (LWR) cores can be reasonably approximated using arrays of cylindrical pins, more complex shapes, including those encountered in fusion neutronics, are difficult to model in CSG. These primitives and corresponding logic operations are specified in text files using modeling languages specific to each MC code. CSG text files are verbose and minimally human-readable, making them tedious to produce and modify.

CAD software offers an intuitive 3D interface which eases geometry construction and modification. CAD formats support higher order (i.e., greater-than-quadratic) surfaces, allowing for arbitrarily complex shapes. Most production-level MC codes support tracking on meshes that are automatically generated from CAD models. Shift, OpenMC, and modified versions of MCNP6 can track particles on triangular surface meshes via the Direct Accelerated Geometry Monte Carlo (DAGMC) library [5]. Serpent 2 can track particles on stereolithography (STL) triangular surface meshes and unstructured polyhedral meshes, and MCNP6 can track particles on unstructured

polyhedral meshes [4]. One limitation of this approach is that curved surfaces must be approximated with a large number of mesh elements, governed by a user-specified tolerance. As a result, these meshes frequently require significantly more computer memory than equivalent CSG models, and likewise, transport on these meshes may be 2.0–3.5 times slower [6] than transport using CSG. This may be a reasonable trade-off if the human effort associated with creating an equivalent CSG model is significant. For brevity, meshes generated from CAD models will be referred to as simply "CAD models" herein.

With both CAD and CSG geometry types, every point within a model's spatial domain must be contained within exactly one of the model's cells. Ill-formed CAD and CSG models may contain gaps, which are regions not contained in any cell, and/or overlaps, which are regions contained in multiple cells. MC particles may become *lost* when encountering these regions, thus preventing the simulation from completing or introducing systematic bias into the simulation. Within some domains, especially fusion, modest lost particle rates are considered acceptable. For example, the official 360° "E-lite" model of ITER loses particles at a rate of 1.3×10^{-6} per source particle when running with void materials [7].

CAD models produced for structural mechanics and thermal hydraulics analysis frequently contain gaps and overlaps because geometric tolerances are less strict in these applications. Although there are some automatic methods for fixing these problematic regions [8], additional labor-intensive manual intervention is often required before the model can be used in MC transport. In addition, any time a CAD model is modified, it must be re-meshed, and this may reintroduce these problematic regions in unpredictable ways.

There are a multitude of spatially complex scenarios across different domains in which it is necessary or convenient to combine CAD and CSG models:

1. Modern fission excore analysis [9] requires combining core and excore models. Core models are almost exclusively modeled in CSG for neutronics analysis, and excore components such as pumps, heat exchangers, and reactor buildings are commonly modeled in CAD [10].
2. Multiphysics simulations of fission cores involve coupling computational fluid dynamics (CFD) with MC neutron transport. Complex spacer grid CAD models are required for CFD analysis, but spacer grids are

often approximated for MC transport [11]. As simulation fidelity expectations increase, it may be necessary to include spacer grid CAD models in CSG cores.

3. The ITER organization has invested considerable effort into MCNP CSG models [7]. Specific analyses may require additional components which may originate in CAD. For example, CAD models of a pellet injector system, a cold valve box, and cooling water pipes were converted to MCNP CSG in order to be integrated with an MCNP CSG model of the ITER tokamak and tokamak complex for shutdown dose rate analysis [12]. It would be convenient to use these CAD models without conversion and integration.
4. Advances in additive manufacturing have expanded the range of feasible shapes for fission fuel elements and other components. The Transformational Challenge Reactor (TCR) [13] uses additively manufactured fuel elements that were designed using CAD and are difficult to model in CSG. Except for the fuel elements, other portions of the TCR core model can be modeled in CSG.
5. Transport has been demonstrated on a CAD urban model semi-automatically generated from data collected by a vehicle-mounted LiDAR unit at the Combined Arms Collective Training Facility (CACTF) [14]. Because LiDAR does not penetrate buildings, it is conceivable that internal details and/or other components might need to be added to these models, ostensibly using CSG.

Several options exist for combining CAD and CSG models. One option is to translate models into the same format. CAD models can be semi-automatically translated to MCNP CSG using tools such as McCAD [15], but this often requires significant manual intervention [9, 12]. MCNP CSG models can be translated to CAD using the MCNP2CAD tool [16], but not all MCNP cell types are supported, and using CAD as the common format might significantly increase memory requirements. Geometry format translations can also be done by hand, but this is extremely labor intensive. Another method of combining CAD and CSG models is the "universe" method. MCNP6 [17] and OpenMC [18] allow CAD universes to be embedded in CSG cells. Serpent 2 allows CAD and CSG universes to be embedded in CAD and CSG cells [19]. Serpent 2 also allows for all of the undefined space in or surrounding a CAD or CSG model to be filled in with a "background universe" [20, 21]. The method described herein is most closely related to

the background universe concept and retains all of the features of these other implementations.

In this work, a *layered* geometry method is proposed, which allows multiple CAD and/or CSG models to be arbitrarily clipped, translated, rotated, and placed in overlapping layers with assigned precedence to form transport-ready geometries. As a proof-of-concept, layered geometry was implemented in the Shift MC code. Section 2 describes how layered geometries are constructed and how MC tracking is performed on layered geometry. Section 3 describes the verification of the implementation of this method within Shift. This implementation is demonstrated in Sections 4 and 5, which address scenarios 4 and 5, respectively. Section 6 provides concluding remarks and planned future work.

2. Methodology

Layered geometries are created by combining pre-constructed models in CAD and/or CSG formats. At this time, the layered geometry feature supports models in Shift's native general geometry (GG) CSG format, the SCALE CSG format [22], and DAGMC CAD. This capability could be expanded to any other geometry format supported by Shift, including the MCNP CSG format and the reactor toolkit (RTK) reactor-aware CSG format. These pre-constructed models are referred to as *constituent* models. The manner in which constituent models are combined to create a layered geometry is illustrated in Figure 1.

Figure 1 shows that the first step is creating *objects* by specifying bounding boxes enclosing any portion of any constituent model. Currently, these bounding boxes must be axis-aligned, but support for oriented bounding boxes could be added in the future. Any materials within the constituent model, denoted by color, may be declared *transparent*. The meaning of transparency will become clear in subsequent steps. Multiple objects can be created from the same constituent model. For example, objects 1 and 2 come from the same constituent model but have different material transparencies. Objects 3 and 4 also come from the same constituent model but are generated using different bounding boxes. The bounding boxes for objects 3 and 4 both truncate portions of the orange cell. Although transparency is currently assigned on the basis of materials, this could be expanded to allow transparency to be assigned on the basis of cells.

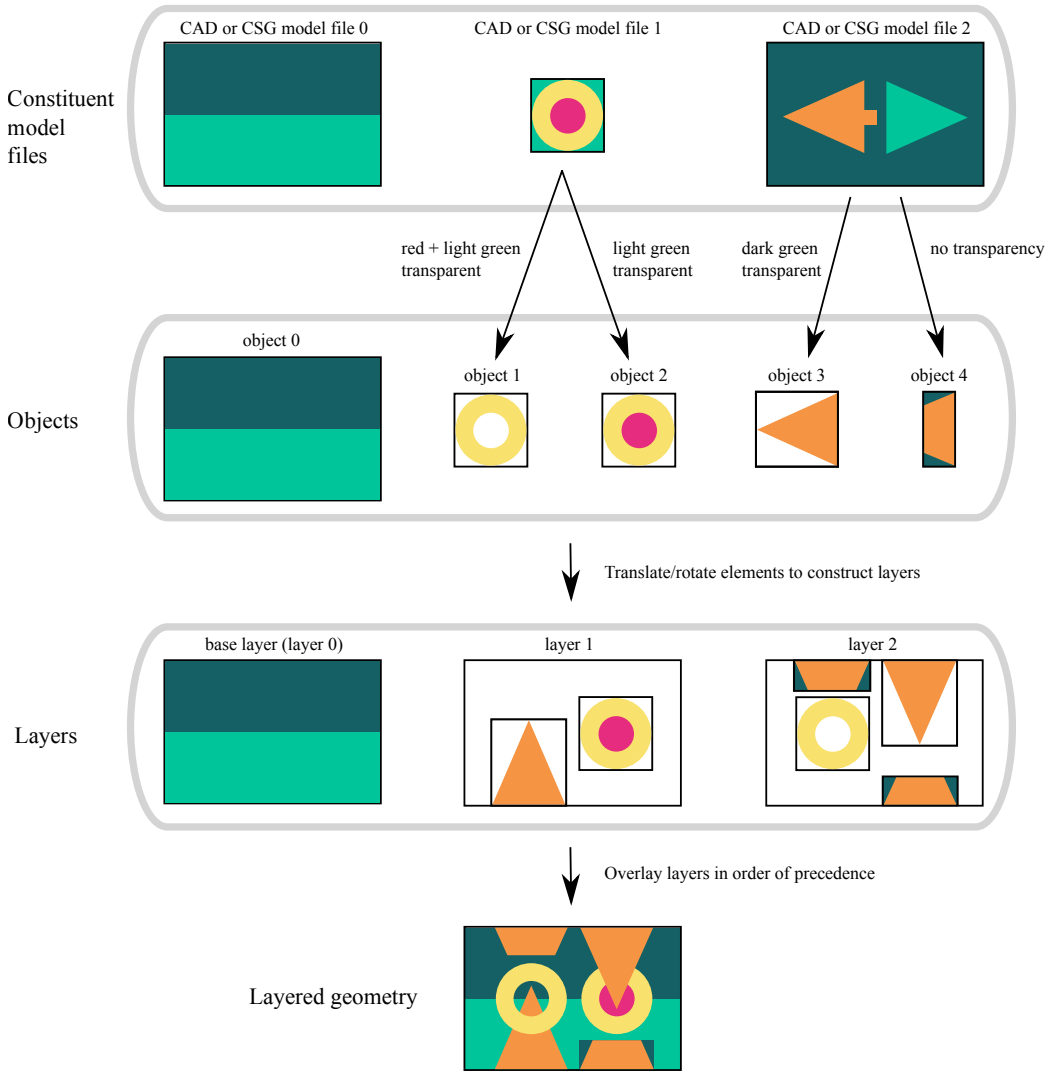


Figure 1: Illustration of the layered geometry construction process. Colors denote material assignments. This 2D example can be extended to 3D without loss of generality.

The next step is creating *layers* from the collection of objects. The layer with the lowest precedence is known as the *base layer*, and must consist of a single object whose bounding box defines the spatial domain of the problem. This ensures that the geometric scene contains no undefined regions. In transport problems in which no constituent model is a convenient base layer, an additional constituent model can be created that consists of a single cell encompassing the full geometry. Future work may relax this requirement, creating a base layer implicitly when necessary. After the base layer, which is numbered layer 0, an arbitrary number of additional layers can be specified with increasing consecutive integers representing precedence. Each non-base layer may contain an arbitrary number of objects which may be translated and/or rotated in any fashion—provided that the bounding boxes of objects on the same layer do not overlap. A single object is permitted to appear multiple times on a single layer, such as object 4 on layer 2 in Figure 1, or multiple times on different layers, such as object 3 on layers 1 and 2. In all cases, each object is stored only once in memory.

The final step is creating the layered geometry itself by superimposing all of the layers in order of precedence, with the base layer on the bottom of the stack, and each subsequent layer stacked on top. Although objects on the same layer cannot have overlapping bounding boxes, objects on different layers are allowed to overlap arbitrarily. Overlapping regions respect the transparency of the specified materials within objects. For example, in Figure 1, the base layer and object 3 can be seen through the transparent center of object 1.

To execute Shift transport on a layered geometry, a user first creates a layered geometry text-based input file. This input file denotes the file paths of the constituent models, corresponding material files, the bounding boxes and transparent materials for each object, and the translations and rotations of the objects on each layer. Transport is then run via Shift's Omnibus interface [23] in the same fashion as any other geometry type. Since constituent models do not have to be modified with the layered geometry approach, this may encourage analysts to maintain libraries of validated models. Complex geometric scenes could then be quickly composed together using these validated model libraries rather than through the creation of a complex monolithic model that must frequently be modified.

Though the focus of this work is on combining CAD and CSG geometries, there are other scenarios where layered geometry may be used for combining pre-constructed models, even if they are in the same geometry format.

Combining multiple CSG geometries may be tedious since surfaces, cells, and materials usually need to be re-numbered. As previously mentioned, combining multiple CAD models requires re-meshing. With layered geometry, no re-numbering would be required when combining CSG models, and no re-meshing would be required when combining CAD models.

2.1. Layered Geometry Tracking

Tracking particles through a layered geometry requires special considerations. The principal strategy is to track particles on all layers independently such that the particle state is updated on all layers for each particle event. Even though a particle is logically tracked on all layers, the particle is always in exactly one cell within the physical system. The layer that contains this cell is the *active* layer. The active layer must be available on demand for cross section lookups and tallies. This can be accomplished using Algorithm 1. This algorithm performs a linear search over the layers in order of decreasing precedence and returns the index of the first layer in which the particle is located within a geometry cell containing a non-transparent material. Future research will determine if tree-based searches (e.g., k-d trees [24] or bounding volume hierarchies) can be used to improve the performance of this algorithm, and subsequent algorithms herein, when layered geometries contain a large number of objects.

The next challenge to address is moving the particle to its next collision site within a layered geometry. With standard MC tracking, this is first accomplished by sampling the number of mean free paths to the next collision (N_c) and then following the procedure shown in Algorithm 2. In this algorithm, a particle is moved within geometry cells and across cell boundaries until it has traveled the distance corresponding to N_c, taking local cross sections into account.

With layered geometry, Algorithm 2 is kept intact, with the `distance_to_boundary`, `move_within_cell`, and `move_across_surface` procedures adapted for tracking on all layers. The layered geometry `distance_to_boundary` procedure is shown in Algorithm 3. This algorithm must consider the distance to boundary on each layer. As an implementation choice, this algorithm returns the minimum distance to boundary across all layers, which might be less than the distance to the next boundary within the physical system. Thus, whenever `move_within_cell` is called, it is guaranteed that a surface crossing will not be encountered on any layer. As a result, the layered geometry `move_within_cell` procedure shown in Algorithm 4

Algorithm 1 Find the active layer for a given position (pos).

```
1: procedure FIND_ACTIVE_LAYER(pos)
2:     for layer_index ∈ [max_layer_index ... 0]
3:         for object ∈ layers[layer_index].objects()
4:             if object.contains_point(pos)
5:                 cell ← object.find_cell(pos)
6:                 if not cell.material.is_transparent()
7:                     return layer_index
8:                 end if
9:             end if
10:        end for
11:    end for
12: end procedure
```

Algorithm 2 Standard MC algorithm for moving a particle to the next collision site, given its current position (pos), direction (dir), and the sampled number of mean free paths to the next collision (N_c).

```
1: procedure MOVE_TO_NEXT_COLLISION_SITE(pos, dir, N_c)
2:     while N_c > 0
3:         xs ← cross_section(pos)
4:         dist_to_collision ← N_c/xs
5:         dist_to_boundary ← distance_to_boundary(pos, dir)
6:         if dist_to_collision < dist_to_boundary
7:             move_within_cell(pos, dir, dist_to_collision)
8:             break
9:         else
10:            move_within_cell(pos, dir, dist_to_boundary)
11:            move_across_surface()
12:            N_c ← N_c - (dist_to_boundary × xs)
13:        end if
14:    end while
15: end procedure
```

simply advances the particle by the same distance on every layer. Likewise, when `move_across_surface` is called, the particle is guaranteed to be on a surface on at least one layer. The layered geometry `move_across_surface` procedure shown in Algorithm 5 loops over all layers and only operates on layers where the particle is currently on a surface.

Algorithm 3 The `distance_to_boundary` procedure for Algorithm 2, adapted for layered geometry.

```
1: procedure DISTANCE_TO_BOUNDARY(pos, dir)
2:     min_dist = distance_to_outer_boundary(pos, dir)
3:     for layer ∈ layers
4:         current_obj = layer.find_object(pos)
5:         if current_obj is not null
6:             obj_dist ← current_obj.distance_to_boundary(pos, dist)
7:             min_dist = min(min_dist, obj_dist)
8:         else
9:             for obj ∈ layer.objects()
10:                bbox_dist = distance_to_box(obj.bounding_box, pos, dir)
11:                min_dist ← min(min_dist, bbox_dist)
12:            end for
13:        end if
14:    end for
15:    return min_dist
16: end procedure
```

Algorithm 4 The `move_within_cell` procedure for Algorithm 2, adapted for layered geometry.

```
1: procedure MOVE_WITHIN_CELL(pos, dir, distance)
2:     for layer ∈ layers
3:         layer.move_within_cell(pos, dir, distance)
4:     end for
5: end procedure
```

There are other strategies for moving a particle to its next collision site within a layered geometry. For example, another option would be to implement a layered geometry `distance_to_boundary` procedure such that the

Algorithm 5 The `move_across_surface` procedure for Algorithm 2, adapted for layered geometry.

```
1: procedure MOVE_ACROSS_SURFACE()
2:     for layer ∈ layers
3:         if layer.particle_on_surface()
4:             layer.move_across_surface()
5:         else
6:             continue
7:         end if
8:     end for
9: end procedure
```

distance to the next cell boundary in the physical system is always returned. The layered geometry `move_within_cell` procedure would then have to allow surface crossings on non-active layers. The principle advantage of Algorithms 3, 4, and 5 is diagnostic. Moving the particle the minimum distance to boundary on any layer ensures that if a particle is lost on any layer, then an error is raised immediately, allowing the cause of the error to be more easily surmised.

Within each layer, particle tracking is simplified by the requirement that object bounding boxes do not overlap. If a particle is within an object's bounding box, then it must be tracked within the object itself. However, if a particle is not within an object's bounding box, then distance-to-boundary queries are inexpensive point-to-plane calculations. Because tracking is only expensive within objects, the performance of a layered geometry is expected to be closely related to the track density in regions where objects on different layers overlap rather than simply scaling with the number of layers present in a geometry. In the future, performance could be optimized by adopting a tracking algorithm in which tracking is performed only on the active layer.

One challenge of layered geometry tracking is robustness against lost particles. In Section 1, it was mentioned that non-physical gaps and overlaps within a model can cause lost particles. However, there are many other possible mechanisms, some of which are introduced by layered geometry. In both CAD and CSG models, coincident surfaces are typically merged together. Thus, when a particle exits one cell, the adjacent cell it enters is known automatically. This feature remains intact within the constituent models of a layered geometry. However, within the current implementation of layered

geometry within Shift no such merging is done for object surfaces coincident with object bounding boxes, or the cuboidal surfaces of object bounding boxes themselves.

When particles attempt to cross or scatter near these unmerged coincident surfaces, floating point issues can cause the particles to become lost. For example, when two objects on a layer are arranged side-by-side, with their bounding boxes touching, a particle may exit one object's bounding box without logically entering the adjacent object's bounding box. Though the current implementation of layered geometry within Shift attempts to explicitly address these cases, results in Sections 3, 4.2, and 5.2 suggest that further work is necessary to fully resolve these issues.

3. Verification

A verification problem was performed to confirm that transport results obtained using the Shift implementation of layered geometry are consistent with transport results obtained from an equivalent CSG model. The geometry shown in Figure 1 was chosen for this exercise and will be referred to as the "verification geometry." This geometry is a good choice as it employs the full range of layered geometry features, i.e., clipping, translation, rotation, transparency, and multiple overlapping layers. The 2D verification geometry was scaled to 10 cm $\times$ 6.6 cm and extruded 10 cm in z to make it 3D. The light green, dark green, yellow, red, and orange materials in Figure 1 were chosen to be air, water, Portland concrete, aluminum, and stainless steel (SS) 316, respectively.

To construct the layered geometry version of the verification geometry, the three constituent models shown in Figure 1 were first created. Constituent model 0 was created using CUBIT CAD software [25] and exported in the standard DAGMC surface mesh format. Constituent models 1 and 2 were created using Shift's native CSG format. A layered geometry was then created by specifying the file paths to these constituent model files, as well as the parameters necessary to create the objects and layers denoted in Figure 1. Figure 2 shows the resultant layered geometry. This plot, as well as the other 2D geometry slices shown in this document, were created with the Omnibus ray tracer [23]. Because the Omnibus ray tracer uses Shift's tracking routines directly, this tool can be used to verify that a layered geometry has been constructed as expected.

An equivalent CSG model of the verification geometry was created in Shift's native CSG format. This was done using planes, cuboids, and cylinders within a single universe. A transport problem was then set up for the layered geometry and CSG versions of the verification geometry. A standard set of material compositions was used [26]. A mono-energetic 10 keV neutron source was chosen, distributed uniformly through space. A $100 \times 66 \times 1$ uniformly spaced tally mesh was superimposed over the full geometry, with a single energy group spanning 8.0–10 keV. Vacuum boundary conditions were applied on all sides.

Transport was then run using both the layered geometry and CSG versions of the verification geometry. Each of these two trials was run on one node with 48 processes on the Apollo compute cluster at Oak Ridge National Laboratory (ORNL), which has two Intel® Xeon® Gold 5118 CPUs per node. Each trial simulated 10^8 histories with different random number seeds. Flux results for the layered geometry trial are shown in Figure 3. The maximum relative errors in the flux across all mesh volume elements for the layered geometry and CSG trials were 0.1636% and 0.1633%, respectively. The layered geometry trial had a lost particle rate of 7.10×10^{-7} per source particle, whereas no particles were lost in the CSG trial. These lost particles came from the layered geometry tracking routines themselves, not the tracking routines specific to the constituent models.

The difference between the layered geometry and CSG fluxes ($\phi_{\text{diff}} = \phi_{\text{layered}} - \phi_{\text{CSG}}$) and the corresponding standard errors in this quantity (S_{diff}) were calculated for each mesh volume element. The fractions of mesh volume elements with ϕ_{diff} within S_{diff}, $2S_{\text{diff}}$, and $3S_{\text{diff}}$ of zero were found to be 69.47%, 95.58%, 99.77%, respectively. These fractions closely match the expected normal distribution of 68.27%, 95.45%, and 99.73%, respectively. Figure 4 shows the relative differences between the layered geometry and CSG fluxes ($(\phi_{\text{layered}} - \phi_{\text{CSG}})/\phi_{\text{CSG}}$), as well as the number of S_{diff} of difference. From these plots it appears that there are no localized systematic differences between the flux results. Though the lost particles observed during the layered geometry trial indicate that further work is necessary to improve tracking robustness, the fluxes obtained from layered geometry and CSG appear statistically equivalent. This indicates that the implementation of layered geometry within Shift is sufficiently reliable for the demonstrative purposes of this work.

Though this is a simple toy problem, timing results are provided for com-

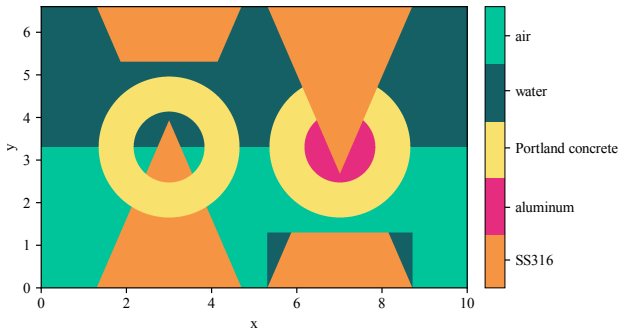


Figure 2: Layered geometry version of the verification geometry, constructed via the procedure shown in Figure 1.

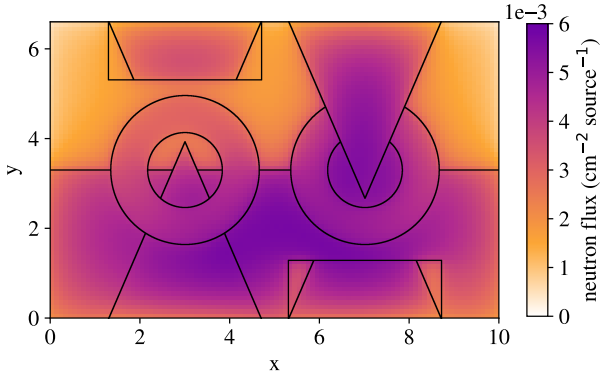
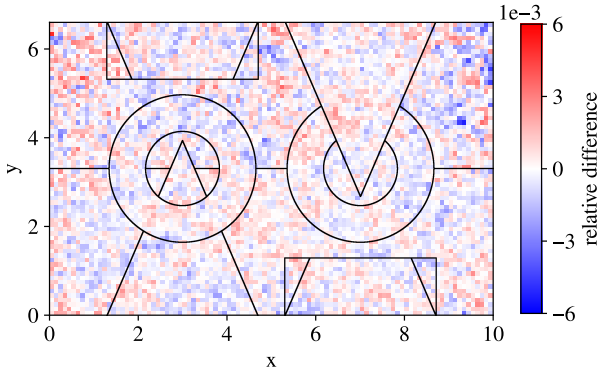
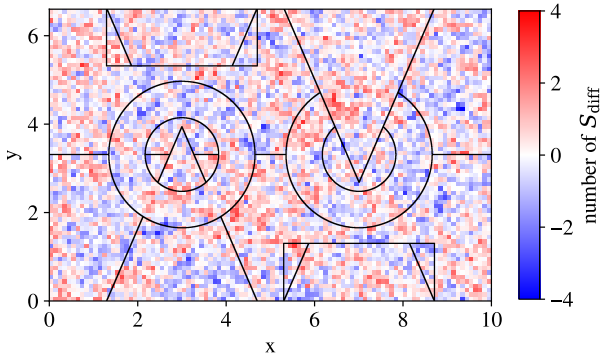


Figure 3: Neutron flux, 8.0–10 keV, layered geometry trial.

pleteness. The simulation with the CSG version of the verification geometry required 0.644 h of wall-clock time to complete (4.31×10^4 histories/(s $\times$ node)), whereas the layered geometry version required 2.58 h (1.07×10^4 histories/(s $\times$ node)), which is 4.03$\times$ longer. Due to the high degree of overlap and the small size of the verification geometry, a significant portion of particle tracks must pass through objects on all three layers. This would account for a factor of three performance penalty. The additional 34.3% performance penalty (i.e., $(4.03-3)/3\times100\%$) can be attributed to the overhead associated with layered geometry routines handling tracking on each layer. Further optimization of the layered geometry capability could likely reduce this performance overhead.



(a)

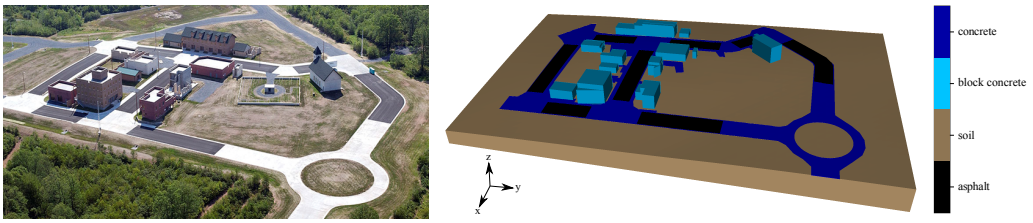


(b)

Figure 4: Discrepancy between layered geometry and CSG fluxes in terms of (a) the relative difference $((\phi_{\text{layered}} - \phi_{\text{CSG}})/\phi_{\text{CSG}})$, and (b) the number of standard errors (S_{diff}) of difference.

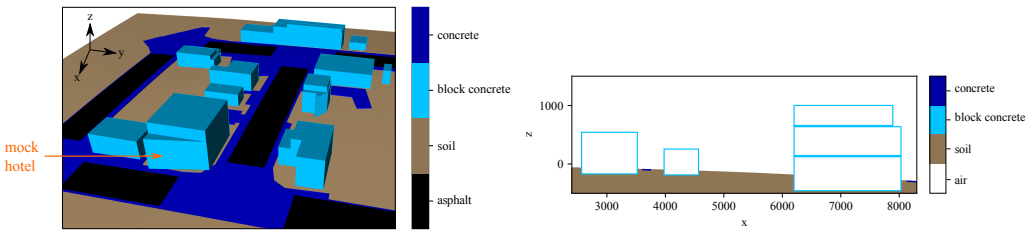
4. Combined Arms Collective Training Facility Demonstration Problem

A principal goal of the Multiagency Urban Search Experiment (MUSE) project [27] was to demonstrate MC transport on CAD geometries generated from vehicle-mounted sensing equipment, thus enabling the rapid prediction of detector responses in urban threat detection scenarios and experiments. The Combined Arms Collective Training Facility (CACTF) [28], shown in Figure 5a, is an urban operations training facility within the Fort Indiantown Gap National Guard Training Center in Lebanon County, Pennsylvania. A CAD model of CACTF, shown in Figure 5b, was generated from unstruc-



(a) Aerial view (b) LiDAR-generated CAD model, 160 m × 260 m × 28.7 m

Figure 5: CACTF aerial view and corresponding CAD model.



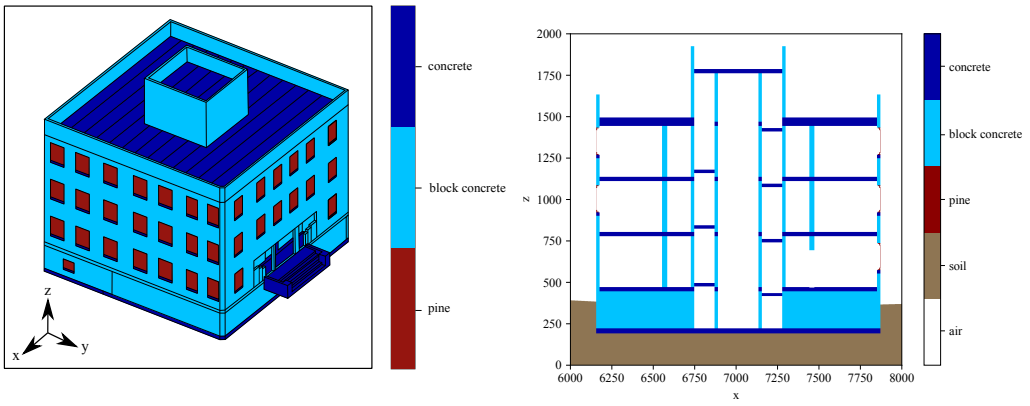
(a) Location of mock hotel (b) x-z slice near the y midplane of the mock hotel

Figure 6: Detailed views of of the CAD model shown in Figure 5b.

tured point cloud data obtained from two vehicle-mounted LiDAR units, with characteristic materials assigned from data collected simultaneously from two spherical digital cameras [14]. This procedure produced approximations of buildings comprised of simple, hollow polyhedra. For example, the mock hotel shown in Figure 6 has no interior detail and a wedge that erroneously extends into the road on the high-x side. For the MUSE project and in this work, this 160 m × 260 m × 28.7 m CAD model was expanded by centering it upon a 1000 m × 1000 m soil platform with a 519 m air volume overhead to properly simulate skyshine [29].

A detailed SCALE CSG model of the CACTF was also created from construction drawings for validation purposes and required "several person-months to complete" [14]. This CSG model is analogous to the CAD model shown in Figure 5b, and it contains detailed building exteriors. Building interiors are not modeled with the exception of the mock hotel. The mock hotel, shown in Figure 7, contains wooden windows, interior walls, a staircase, and an elevator shaft. A SCALE CSG model of a mobile detector apparatus, shown in Figure 8, was also created as part of the MUSE project.

LiDAR-generated CAD geometries may need to be modified to simulate certain scenarios or experiments. For example, a threat source present



(a) Drawing

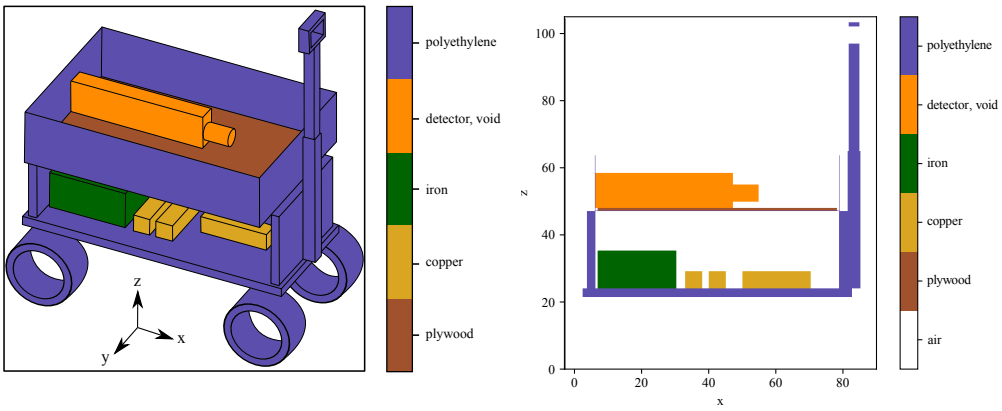
(b) x-z slice near the y midplane

Figure 7: SCALE CSG model the CACTF mock hotel.

within a building would require a detailed interior model. Certain geometry features may not have been present during LiDAR data collection, or they may occupy multiple locations during different simulations. These features may include vehicles, simulated rubble, and detectors, and they may have pre-constructed geometry models in CAD or CSG formats. For the purpose of this demonstration, a scenario was developed in which the detailed CSG mock hotel shown in Figure 7 and the CSG detector apparatus shown in Figure 8 are overlaid upon the LiDAR-generated CAD model shown in Figures 5 and 6.

4.1. Layered Geometry Construction

A layered geometry, shown in Figure 9, was created using the LiDAR-generated CAD model shown in Figures 5 and 6 as a base layer (i.e., layer 0), overlaid with the detailed CSG mock hotel shown in Figure 7 and the CSG detector apparatus shown in Figure 8, both on layer 1. The object bounding box around the mock hotel was chosen to include the air around the mock hotel. The air volume in and around the mock hotel object was intentionally not made transparent, allowing the CSG mock hotel to fully displace the CAD mock hotel, including the erroneous wedge portion of the model that extends into the road. The coordinate system of the CSG model differs slightly from the coordinate system of the CAD model, so the mock hotel object had to be translated 5.75 m in the negative z direction to place it properly. Because the soil volume in the CAD and CSG models differ,



(a) Drawing

(b) x-z slice on the y midplane

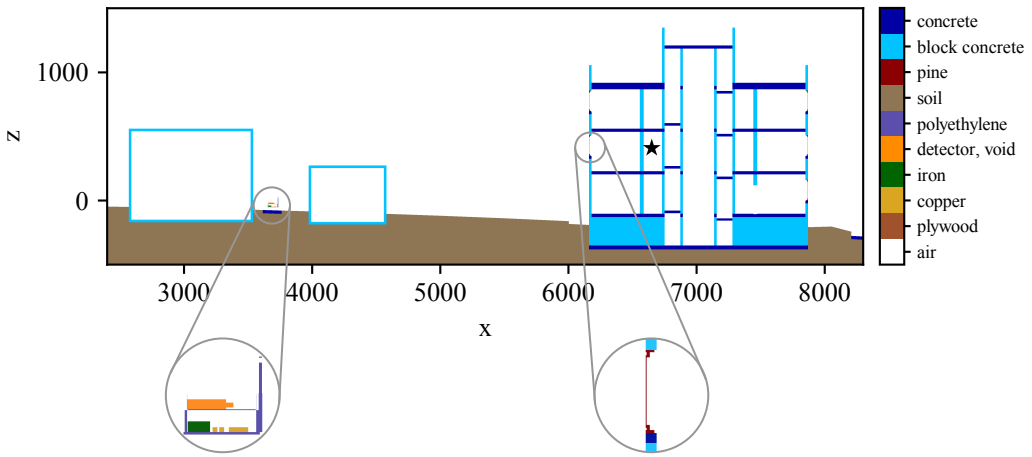
Figure 8: SCALE CSG model of a mobile detector apparatus consisting of a small cart with an NaI detector and other equipment.

there are slight discontinuities in the soil at the CAD/CSG interfaces as seen at $x = 6000$ and $x = 8200$ in Figure 9a.

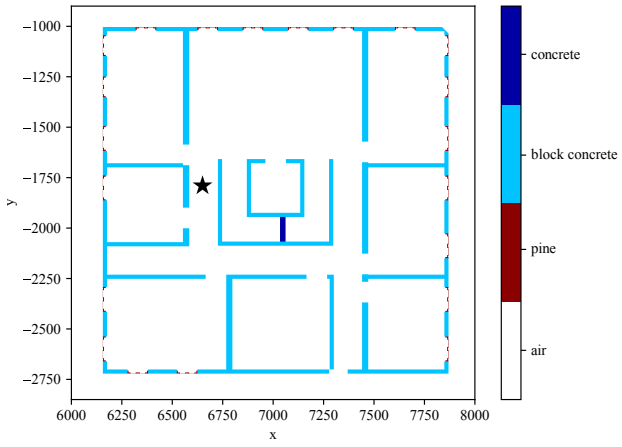
The source and detector locations were chosen to make the problem interesting from a radiation transport perspective. A source was placed in the hallway of the second floor of the mock hotel as indicated by a star in Figures 9a and 9b. From this position, particles can stream through many of the optically thin pine windows on the high y and low x sides of the building after undergoing a small number of scattering events. As a result, there are many viable pathways that particles can take to leave the building and reach the detector. The detector apparatus is centered on the same y plane as the source and is placed between two small buildings in x. The building centered near $x = 4300$ shields the detector from particles streaming out of the mock hotel windows, and the building centered near $x = 3100$ provides a surface near the detector for streaming particles to scatter off of. Although the source and detector are located in a small portion of the CACTF geometry, the geometry was not truncated. The full LiDAR-generated CAD model with additional soil and air skyshine volumes was kept intact for transport.

4.2. Transport Results

Transport was run on the layered geometry shown in Figure 9. The source was chosen to be an 81 μCi ^{137}Cs photon source with 1290 energy bins. An NaI flux-to-count rate response function with 36 bins was applied to the



(a) x-z slice through the source and detector midplane



(b) x-y slice through the source

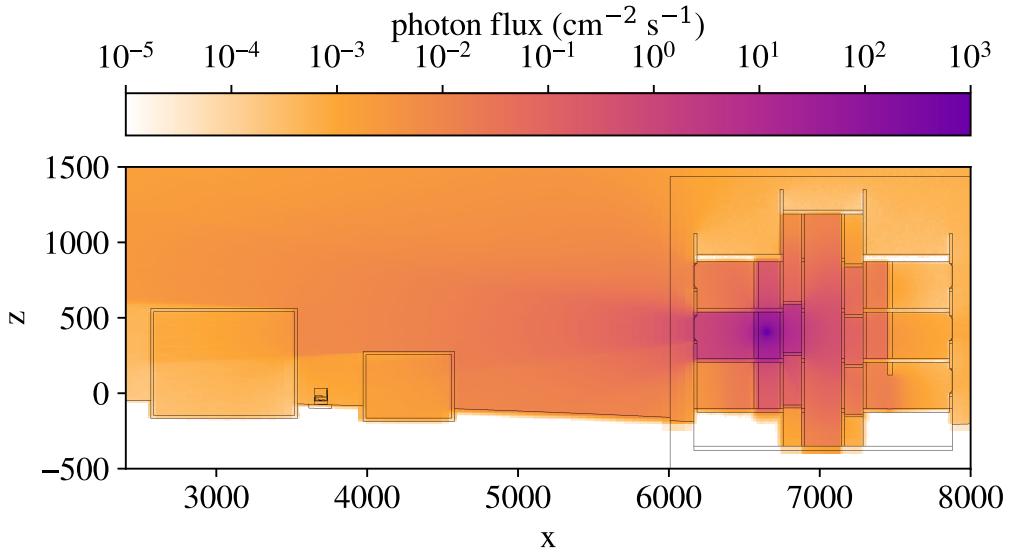
Figure 9: Layered geometry created from the LiDAR-generated CAD model shown in Figures 5 and 6, the CSG mock hotel in Figure 7, and the CSG detector apparatus in Figure 8. The star represents the location of the photon source.

cuboidal detector volume. To obtain results with acceptably low statistical uncertainty, 8×10^9 histories were simulated without variance reduction (other than implicit capture). Ideally, the consistent adjoint-driven importance sampling (CADIS) method [30] would be used for variance reduction. This method can be executed automatically within Shift using the Denovo 3D S_N code [31]. Shift currently does not allow CADIS to be performed on layered geometries, but future support is planned and is expected to be straightforward. The simulation was run on 20 nodes with 48 processes per node on the Apollo compute cluster at ORNL. As mentioned in Section 3, Apollo has two Intel® Xeon® Gold 5118 CPUs per node.

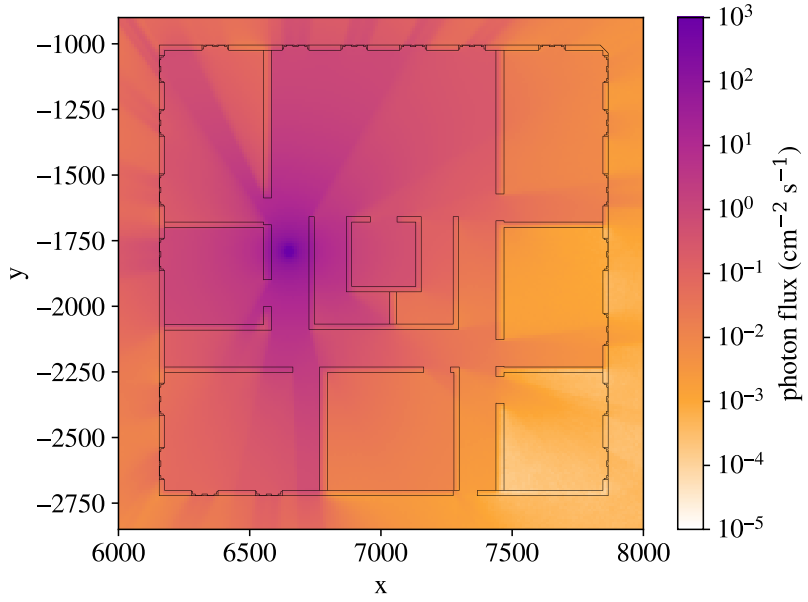
Transport results are shown in Figure 10. As expected, Figure 10b shows that photons exit the mock hotel primarily through the windows. Figure 10a shows that the detector lies within a "shadow" cast by the small building centered near $x = 4300$. Thus, the total detector response was found to be 0.25 ± 0.01 counts/s (3.92% relative error). This simulation required 29.6 h of wall-clock time to complete, with a tracking rate of 3.76×10^3 histories/(s $\times$ node). This tracking rate is 2.85× less than the tracking rate for the layered geometry version of the verification problem in Section 3. Although this problem has fewer layers than the verification problem, this slower tracking rate is expected as there is considerably more geometric complexity on each layer. In addition, the verification geometry is physically small and optically thin, meaning many histories are terminated via leakage. In contrast, the CACTF geometry is much larger and more attenuating, so the average number of tracks per history is likely higher.

A lost particle rate of 1.52×10^{-7} per source particle was observed. The lost particle rate within the tracking routines specific to layered geometry was only 2.50×10^{-10} per source particle (i.e., 0.164% of the total lost particle rate). This is significantly less than the lost particle rate found in the verification problem, as the verification problem likely involves a much larger fraction of particle tracks crossing object boundaries within layers. Another factor could be that this problem consists primarily of planar surfaces, which may be less likely to cause floating point issues compared to the curved surfaces found in the verification problem. The remainder of lost particles were lost within Shift tracking routines specific to SCALE CSG geometry. Of these SCALE CSG lost particles, 88.9% were particles that became "stuck" while tracking, i.e., unable to advance from a particular position.

To further investigate transport, timing, and lost particle results, two additional transport problems where run: a CAD-only simulation, and a



(a) Flux, $x - z$ slice through the source and detector midplane



(b) Flux, $x - y$ slice through the source

Figure 10: Transport results using the layered geometry model shown in Figure 9.

CSG-only simulation. In both cases, the detector apparatus was not included, and instead, a mesh tally with a single cell conformal to the cuboidal portion of the detector was used. Transport results for the CAD-only simulation are shown in Figure 11. Because the entire second floor of the mock hotel is a single cell, none of the nuanced streaming behavior within the mock hotel is captured. The detector response in this case was 0.87 ± 0.02 counts/s (2.25% relative error), which is 3.5 ± 0.2 times higher than the response with layered geometry. In other words, not including the detailed interior of the mock hotel caused a significant overestimate of the detector response. This simulation required only 13.3 h of wall-clock time with a tracking rate of 8.37×10^3 histories/(s $\times$ node). No particles were lost during the simulation.

The CSG-only simulation included the full SCALE CSG CACTF model, not just the mock hotel portion used for layered geometry construction. This simulation did not complete in 37 h, and a lost particle rate 1.50×10^{-7} per source particle was observed, with 81.7% of these particles stuck. This lost particle rate confirms that the vast majority of particles lost in the layered geometry simulation were not the result of the tracking routines specific to layered geometry. The slightly different stuck particle fraction (81.7% vs. 88.9%) may be a result of how stuck particle issues manifest themselves within layered geometry versus pure CSG geometry.

The fact that the CSG-only simulation did not complete suggests that the layered geometry runtime may be artificially high because of stuck particles within the CSG mock hotel object within the layered geometry. Nonetheless, a factor of 2.23 runtime penalty for layered geometry versus the CAD-only geometry (29.6 h vs. 13.3 h) is not unreasonable, especially considering that the CAD-only geometry does not capture the detailed geometric features of the mock hotel. In this problem, the highest track density region (surrounding the source) is in a region where the CSG and CAD geometry overlap. As a result, the majority of particle tracks must be simulated on both the CSG mock hotel and the LiDAR-generated CAD geometry, which would cause a factor of two performance penalty if tracking rates were equal within both models. This computer-time penalty is modest, considering the significant human effort that is saved by combining these models using layered geometry.

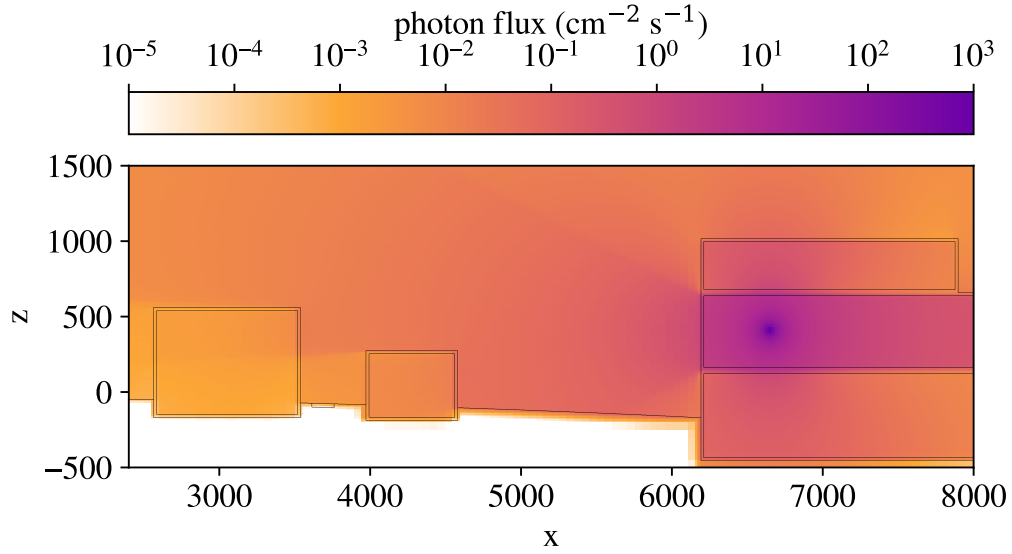


Figure 11: Transport results from the CAD-only simulation using the geometry shown in Figures 5 and 6.

5. Transformational Challenge Reactor Demonstration Problem

The TCR program [13] explored how advancements in additive manufacturing enable complex core designs. A core design was proposed [32], as shown in Figure 12, consisting of cog-shaped fuel elements, as shown in Figure 13. These fuel elements consist of additively manufactured SiC shells loaded with 19.5% enriched uranium mononitride (UN) tristructural isotropic (TRISO) particles in a SiC matrix. Additional integrated reflector elements

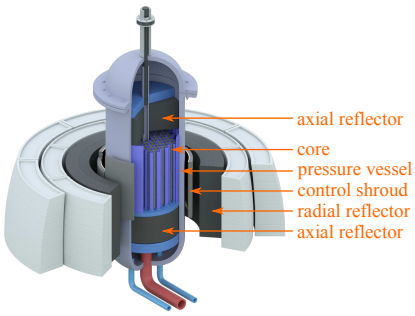


Figure 12: Rendering of the Transformational Challenge Reactor.

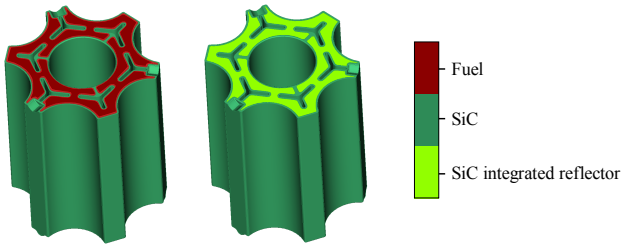


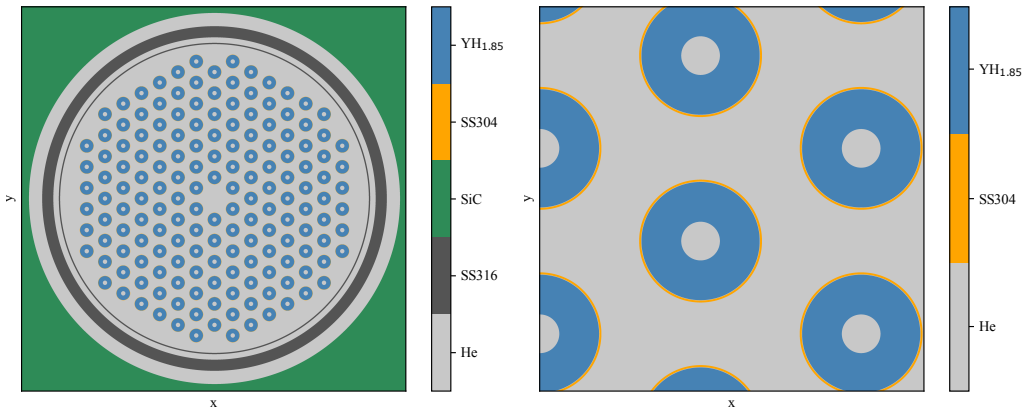
Figure 13: CAD fuel and reflector elements that differ only by material assignments.

consist of the same SiC shell packed only with SiC. These fuel and reflector elements were originally designed in CAD and have wishbone-shaped cooling channels that cannot be analytically represented by quadratic surfaces; therefore, they can only be approximated in CSG. An assembly consists of eight fuel elements and three integrated reflector elements stacked axially with a $YH_{1.85}$ moderator rod through the center. The full core consists of 54 assemblies arranged in a hexagonal grid for a total of 594 fuel/reflector elements. Additional "free" moderator rods occupy the spaces formed between adjacent assemblies. A layered geometry demonstration problem was performed with the TCR geometry as first reported at the International Conference on Physics of Reactors 2022 (PHYSOR 2022) [33].

5.1. Layered Geometry Construction

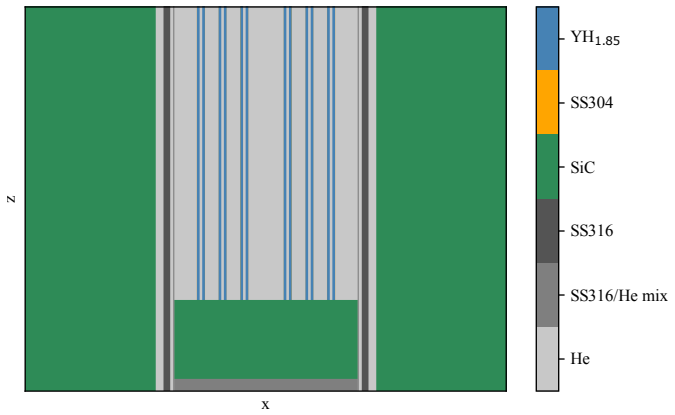
To construct a layered geometry model of TCR, a Shift CSG base layer was first created as shown in Figure 14. This model does not contain a control shroud and is truncated below the upper axial reflector, because neutrons that make it past the integrated reflector elements above the fuel elements are not expected to return to the core. Next, a layered geometry object was created from each of the CAD models shown in Figure 13. The void material around the models was assigned to be transparent. This allows the base layer moderator rods to appear in the center of each element and within the spaces between adjacent elements. The layered geometry implementation in Shift allows the fuel and reflector CAD models to be stored in memory only once, despite appearing a total of 594 times to form the full core.

As mentioned in Section 2, only axis-aligned bounding boxes are currently supported when creating layered geometry objects, and these bounding boxes must not overlap for all objects within a single layer. As a result, multiple layers were created so that fuel/reflector elements could be placed into a



(a) Radial slice (radial reflector truncated)

(b) Radial slice detail



(c) Axial slice

Figure 14: Shift CSG model of the TCR moderator rods and excore.

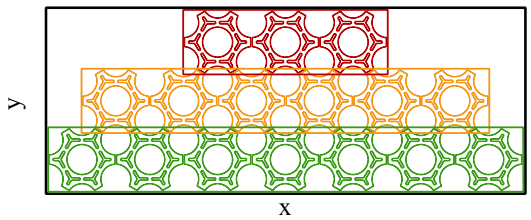


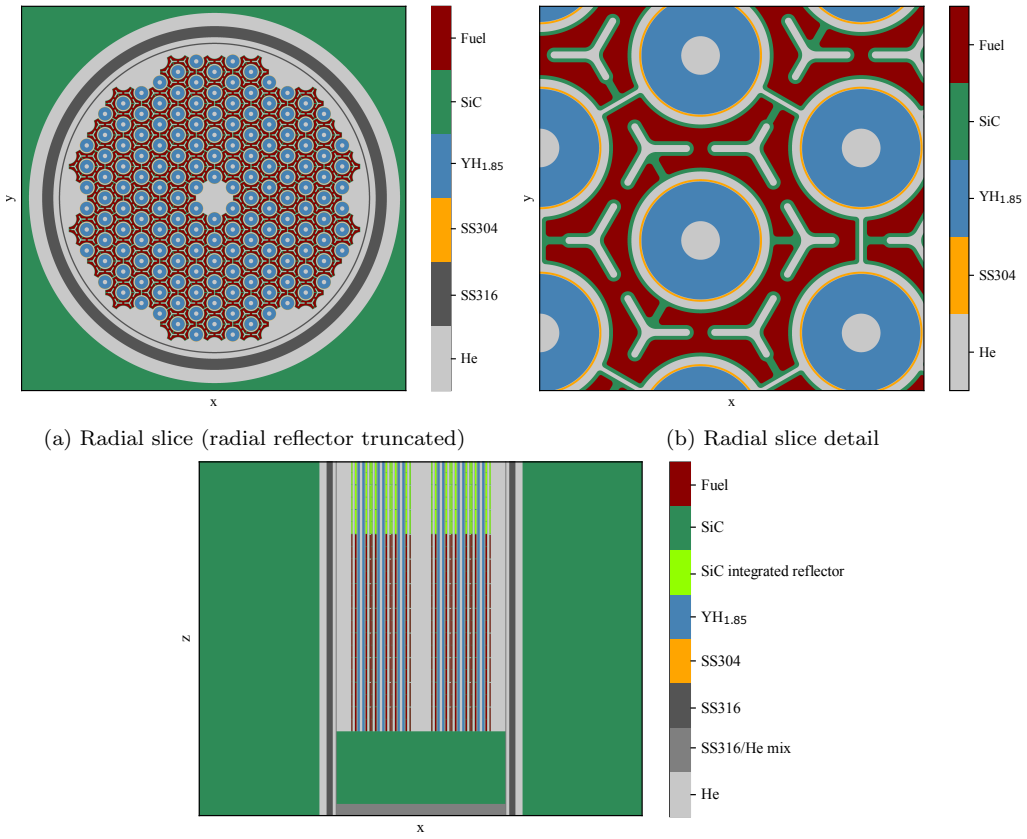
Figure 15: Layering scheme for the first three rows of assemblies. Colors denote layers.

hexagonal grid. The layering scheme is shown in Figure 15. This figure shows that each row of assemblies is placed in a different layer (denoted by color), because the bounding boxes of assemblies in adjacent rows overlap. TCR has nine rows of assemblies, thus nine layers are placed on top of the base layer to form the full layered geometry as shown in Figure 16. An equivalent layered geometry could be created with two layers on top of the base layer, with each layer containing alternating rows of assemblies (i.e., the red and green rows in Figure 15 would be contained in the same layer). However, this scheme is not expected to provide any significant performance advantage, so the simpler scheme was used.

The bottom of each fuel element CAD model contains three hemispherical alignment features that fit into three arch-shaped dimples on the top of each element (as seen in Figure 13). Simulating these alignment features would require the bounding boxes of the fuel/reflector elements to overlap with their axial neighbors, which would require them to be on different layers. For the purpose of this demonstration, these alignment features were truncated via the object bounding box (leaving the dimples unoccupied) to simplify the layering scheme.

5.2. Transport Results

A Shift eigenvalue simulation was performed using the layered geometry in Figure 16. To mimic cold zero power conditions, a temperature of 300 K was used throughout the model. Eigenvalue iteration was performed with 25 inactive cycles and 25 active cycles, with 5×10^6 histories per cycle. Total flux and fission source results were tallied on a $420 \times 420 \times 1$ radial mesh, with a z width extending ± 2 cm about the axial midplane of the fuel elements. An additional $20 \times 20 \times 20$ mesh over the fuel elements was used to assess the Shannon entropy. The simulation was run on the same compute cluster with the same number of nodes/processors as described in Section 4.2.



(a) Radial slice (radial reflector truncated)

(b) Radial slice detail

(c) Axial slice

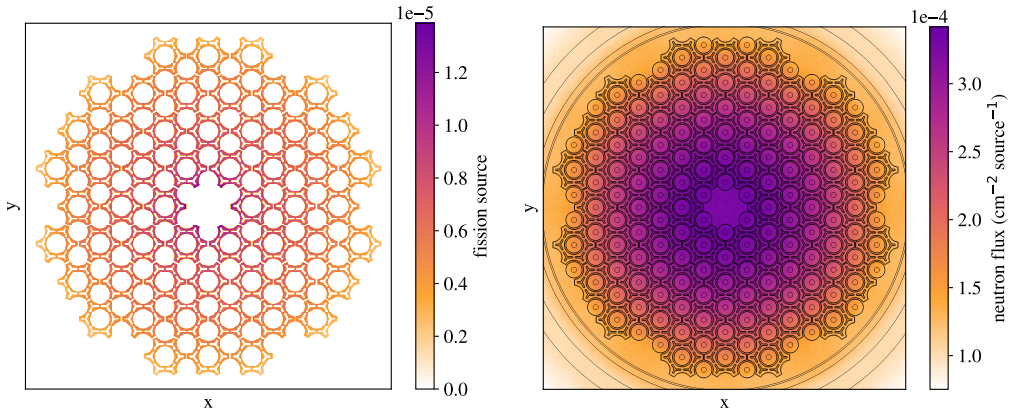
Figure 16: Layered geometry created using the constituent models in Figure 13 and 14.

Transport results are shown in Figure 17. The fission source and flux distribution match expectations, and the k_{eff} and Shannon entropy results show that the simulation achieved a degree of convergence that is acceptable for this demonstration, with a final k_{eff} of 1.0212 ± 0.0005. Transport required 15.7 h of wall-clock time with a tracking rate of 220 histories/(s $\times$ node). This tracking rate is $48.6\times$ slower than the layered geometry version of the verification problem in Section 3. As was the case with the CACTF problem in Section 4.2, this slower tracking rate is explained by the high degree of geometric complexity and the higher number of tracks per history.

Unlike in Section 4.2, the CAD fuel and reflector elements were found to be the major source of lost particles, with a lost particle rate of 6.09×10^{-5} per source particle from DAGMC geometry errors. Particles were also lost via the layered geometry tracking process itself, accounting for an additional lost particle rate of 3.07×10^{-6} per source particle. This is $4.31\times$ higher than the lost particle rate in the Section 3 verification problem, but still $19.9\times$ lower than the lost particle rate from DAGMC errors. It is possible that some of these layered geometry tracking lost particles originated from issues with DAGMC tracking.

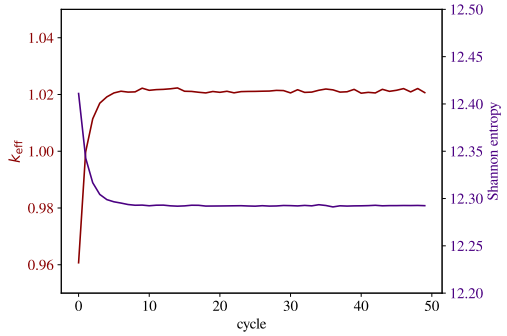
In addition to lost particles, erroneous particles were observed to cause streaks of small but non-zero fission source density through regions not containing fissionable material at an estimated rate of $\sim 4 \times 10^{-6}$ per source particle. It is hypothesized that many of these erroneous particles eventually become lost and are therefore already accounted for in the reported lost particle rate, but more analysis is needed to confirm this. In Shift, source particles are re-sampled if they are born in non-fissionable material, so these rare erroneous particles only slightly bias the fission source distribution.

It is emphasized that although k_{eff}, as well as the fission source and flux distributions match expectations, lost or erroneous particles are not acceptable for production-level eigenvalue calculations, which have much finer tolerances than fixed source shielding problems. Though the majority of particles in the problem were lost within the CAD model, the fact that the tracking routines specific to layered geometry had a lost particle rate of 3.07×10^{-6} per source particle indicates that robustness improvements to the Shift implementation of layered geometry are necessary prior to use with eigenvalue problems.



(a) Fission source distribution

(b) Flux distribution



(c) $\mathbf{k}_{\text{eff}}$ and Shannon entropy

Figure 17: Transport results for the layered geometry shown in Figure 16.

6. Conclusion

The layered geometry feature implemented within Shift provides a versatile mechanism for combining CAD and CSG models arbitrarily to represent spatially complex radiation transport scenarios. The verification problem in Section 3 demonstrated tracking through a layered geometry with clipping, translation, rotation, transparency, and multiple overlapping layers. Due to the high degree of overlap, the tracking rate appeared to be correlated with the number of layers with a modest amount of additional overhead. Though the simulation was not free of geometry errors, these did not appear to have a statistically significant effect on the flux, indicating that the implementation of layered geometry in Shift is suitable for the demonstrative purposes of this work.

With the CACTF demonstration problem in Section 4, CSG models were overlaid upon a LiDAR-generated CAD model. Flux results showed fully resolved streaming paths within the mock hotel which could not be captured using the CAD model alone. Likewise, the tally result of 0.25 ± 0.01 counts/s was 3.5 ± 0.2 times lower than the results from using the CAD model alone. The highest flux region of the problem (surrounding the source) was within the region where the two layers overlapped. Likewise, particle tracking rates scaled roughly with the number of layers; tracking on the layered geometry was 2.23× less efficient compared to tracking on the CAD model alone. During the simulation only 0.164% of lost particles were lost in the layered geometry tracking routines themselves, with the remainder lost within Shift tracking routines specific to the SCALE CSG format. This highlights that tracking on a layered geometry can only be as robust as the format-specific tracking routines of the constituent models. The lost particle rate within layered geometry tracking routines themselves was 2.50×10^{-10} per source particle, which would be considered acceptable for most fixed source shielding calculations.

With the TCR demonstration problem, complex CAD fuel and reflector elements were arranged in 9 layers on top of a CSG base layer to form a 594-element core model, with each fuel and reflector element stored in memory only once. The centers and areas surrounding these elements were declared transparent in order to allow moderator rods to occupy interstitial spaces. The converged fission source and flux distributions, as well as the k_{eff} of 1.0212 ± 0.0005, matched expectations. Slower particle tracking rates (48.6× slower than the layered geometry version of the verification problem) can be

explained by the fact that this problem has more layers, more objects per layer, and highly complex CAD models. As in the CACTF demonstration, the majority of lost particles were lost within a constituent model, in this case the CAD fuel and reflector elements. However, a lost particle rate of 3.07×10^{-6} per source particle within the layered geometry tracking routines themselves is not acceptable for most eigenvalue calculations. This indicates that tracking robustness improvements are necessary prior to using layered geometry for these applications.

In addition to robustness improvements, several other improvements are planned to enhance the applicability and performance of layered geometry. First, as Shift already has the ability to track particles on MCNP models, extending layered geometry to support MCNP constituent models is expected to be straightforward. This will provide flexibility to analysts that rely heavily on existing MCNP models. Second, support for CADIS/FW-CADIS (which also already exists within Shift for other geometry types) will also be added to layered geometry for use with shielding problems. Finally, performance enhancements such as the use of k-d trees or bounding volume hierarchies will be explored. This could improve performance in cases such as the TCR problem which involve a large number of objects. With these enhancements, layered geometry has the potential to significantly simplify the process of combining existing CAD and CSG models for production-level radiation transport analysis.

Acknowledgments

The authors would like to thank Douglas Peplow for his help with the models used in the CACTF problem in Section 4. This work was supported by the Laboratory Directed Research and Development (LDRD) program of Oak Ridge National Laboratory, which is managed and operated by UT-Battelle LLC for the US Department of Energy (DOE) under contract no. DEAC05-00OR22725.

References

[1] T. M. Pandya, S. R. Johnson, T. M. Evans, G. G. Davidson, S. P. Hamilton, A. T. Godfrey, Implementation, capabilities, and benchmarking of Shift, a massively parallel Monte Carlo radiation transport code, Journal of Computational Physics 308 (2016) 239–272. doi:10.1016/j.jcp.2015.12.037.

[2] P. Romano, B. Forget, The OpenMC Monte Carlo particle transport code, Annals of Nuclear Energy 51 (2013) 274–281. doi:10.1016/j.anucene.2012.06.040.

[3] J. Leppänen, M. Pusa, T. Viitanen, V. Valtavirta, T. Kaltiaisenaho, The Serpent Monte Carlo code: Status, development and applications in 2013, Annals of Nuclear Energy 82 (2015) 142–150. doi:10.1016/j.anucene.2014.08.024.

[4] D. Pelowitz, MCNP6 user's manual version 1.0, Tech. Rep. LA-CFP-13-00634 Rev 0, Los Alamos National Laboratory (2013).

[5] T. J. Tautges, P. P. H. Wilson, J. Kraftcheck, B. F. Smith, D. L. Henderson, Acceleration techniques for direct use of CAD-based geometries in Monte Carlo radiation transport, in: International Conference on Mathematics, Computational Methods & Reactor Physics, 2009.

[6] A. Davis, L. Jacobson, P. Wilson, F. Gallmeier, The use of CAD based radiation transport in support of SNS: DAG-MCNP6 validation, 13th International Topical Meeting on Nuclear Applications of Accelerators 2017 (AccApp 2017) (2018) 72–80.

[7] R. Juarez, G. Pedroche, M. Loughlin, R. Pampin, P. Martinez, M. Pietri, J. Alguacil, F. Ogando, P. Sauvan, A. Lopez-Revelles, A. Kolšek, E. Polunovskiy, M. Fabbri, J. Sanz Recio, A full and heterogeneous model of the ITER tokamak for comprehensive nuclear analyses, Nature Energy 6 (2021) 1–8. doi:10.1038/s41560-020-00753-x.

[8] B. Smith, Robust tracking and advanced geometry for Monte Carlo radiation transport, Ph.D. thesis, University of Wisconsin (2011).

[9] E. E. Davidson, A. T. Godfrey, K. E. Royston, T. M. Pandya, S. C. Henderson, T. M. Evans, Validation of light water reactor ex-core calculations with VERA, Nuclear Technology (2021). doi:10.1080/00295450.2021.1957660.

[10] E. E. Davidson, G. Radulescu, K. Smith, J. Yang, S. Wilson, B. R. Betzler, Reactor cell neutron dose for the molten salt breeder reactor conceptual design, Nuclear Engineering and Design 383 (2021) 111381. doi:10.1016/j.nucengdes.2021.111381.

[11] A. Bennett, G. Bache, N. Martin, J. Senecal, BWR spacer grid modeling using SERPENT 2 / STAR-CCM+ coupling, International Conference on Mathematics Computational Methods Applied to Nuclear Science and Engineering (2019).

[12] J. Yang, S. C. Wilson, S. W. Mosher, G. Radulescu, Integration of the full tokamak reference model with the complex model for ITER neutronic analysis, Fusion Science and Technology 74 (4) (2018) 277–287. doi:10.1080/15361055.2018.1493325.

[13] Oak Ridge National Laboratory, Transformational challenge reactor program, https://tcr.ornl.gov/ (2022).

[14] D. E. Peplow, G. G. Davidson, C. Celik, E. D. Biondo, A. C. Hackett, W. R. Ray, D. E. Archer, J. James M. Ghawaly, A. D. Nicholson, M. J. Willis, B. J. Quiter, M. S. Bandstra, R. E. Meyer, C. H. Chow, I. R. Stewart, J. O. Johnson, Monte Carlo simulation of background and source measurements with CSG and CAD geometries, Tech. Rep. ORNL/TM-2021/2078, Oak Ridge National Laboratory (2021). doi:10.2172/1805000.

[15] D. Grosse, U. Fischer, K. Kondo, D. Leichtle, P. Pereslavtsev, A. Serikov, Status of the McCad geometry conversion tool and related visualization capabilities for 3D fusion neutronics calculations, Fusion Engineering and Design 88 (9) (2013) 2210–2214. doi:10.1016/j.fusengdes.2013.02.146.

[16] P. P. H. Wilson, MCNP2CAD, https://github.com/svalinn/mcnp2cad (2013).

[17] R. L. Martz, The MCNP6 book on unstructured mesh geometry: User's guide for MCNP 6.2, Tech. Rep. LA-UR-17-22442, Los Alamos National Laboratory (2017).

[18] P. Shriwise, DAGMC CAD-based geometry as universes, https://github.com/openmc-dev/openmc/pull/1825 (2021).

[19] J. Leppänen, CAD-based geometry type in Serpent 2–application in fusion neutronics, Joint International Conference on Mathematics and Computation, Supercomputing in Nuclear Applications, and the Monte Carlo Method (2015) 19–23.

[20] A. Jambrina, Re: Background universes, https://ttuki.vtt.fi/serpent/viewtopic.php?f=3&t=4015#p13620 (2022).

[21] A. Talamo, Y. Gohar, J. Leppänen, SERPENT validation and optimization with mesh adaptive search on stereolithography geometry models, Annals of Nuclear Energy 115 (2018) 619–632. doi:10.1016/j.anucene.2018.01.012.

[22] B. Rearden, M. A. Jessee, Eds., SCALE code system, version 6.2, Tech. Rep. ORNL/TM-2005/39, Oak Ridge National Laboratory, Oak Ridge, TN (2016). doi:10.2172/1424483.

[23] S. R. Johnson, T. M. Evans, G. G. Davidson, S. P. Hamilton, T. M. Pandya, K. E. Royston, E. D. Biondo, Omnibus user manual, Tech. Rep. ORNL/TM-2018/1073, Oak Ridge National Laboratory, Oak Ridge, TN (2020). doi:10.2172/1649625.

[24] J. L. Bentley, Multidimensional binary search trees used for associative searching, Communications of the Association for Computing Machinary 18 (9) (1975) 509–517. doi:10.1145/361002.361007.

[25] T. D. Blacker, S. J. Owen, M. L. Staten, W. R. Quadros, B. Hanks, B. W. Clark, R. J. Meyers, C. Ernst, K. Merkley, R. Morris, C. McBride, C. Stimpson, M. Plooster, S. Showman, CUBIT geometry and mesh generation toolkit 15.2 user documentation, Tech. Rep. SAND2016-6850R, Sandia National Laboratory (2016).

[26] R. J. McConn, C. J. Gesh, R. T. Pagh, R. A. Rucker, R. Williams, III, Compendium of material composition data for radiation transport modeling, Tech. Rep. PNNL-15870, Pacific Northwest National Laboratory, Richland, WA (2011).

[27] A. D. Nicholson, I. Garishvili, D. E. Peplow, D. E. Archer, W. R. Ray, M. W. Swinney, M. J. Willis, G. G. Davidson, S. L. Cleveland, B. W. Patton, D. E. Hornback, J. J. Peltz, M. S. L. McLean, A. A. Plionis, B. J. Quiter, M. S. Bandstra, Multiagency urban search experiment detector and algorithm test bed, IEEE Transactions on Nuclear Science 64 (7) (2017) 1689–1695. doi:10.1109/TNS.2017.2677092.

[28] Fort Indiantown Gap, Combined arms collective training facility (CACTF), https://www.ftig.ng.mil/Training/Urban-and-Collective-Training-Sites/Combined-Arms-Collective-Training-Facility (2022).

[29] G. A. Sandness, J. E. Schweppe, W. K. Hensley, J. D. Borgardt, A. L. Mitchell, Accurate modeling of the terrestrial gamma-ray background for homeland security applications, in: 2009 IEEE Nuclear Science Symposium Conference Record (NSS/MIC), 2009, pp. 126–133. doi:10.1109/NSSMIC.2009.5401843.

[30] A. Haghighat, J. C. Wagner, Monte Carlo variance reduction with deterministic importance functions, Progress in Nuclear Energy 42 (1) (2003) 25–53. doi:10.1016/S0149-1970(02)00002-1.

[31] T. M. Evans, A. S. Stafford, R. N. Slaybaugh, K. T. Clarno, Denovo: A new three-dimensional parallel discrete ordinates code in SCALE, Nuclear Technology 171 (2010) 171–200. doi:10.13182/NT171-171.

[32] B. R. Betzler, et al., Transformational Challenge Reactor preliminary core design report, Tech. Rep. ORNL/TM-2020/1718, Oak Ridge National Laboratory (2020).

[33] E. Biondo, G. Davidson, B. Ade, Layered CAD/CSG geometry for neutronics modeling of advanced reactors, in: International Conference on Physics of Reactors, 2022. doi:10.13182/PHYSOR22-37768.