# CMGC: a CAD to Monte Carlo geometry conversion code

Xin Wang[1,2] · Jun-Li Li[3] · Zhen Wu[4] · Shen-Shen Gao[3] · Rui Qiu[3] ·
Li Deng[1] · Gang Li[1]

Received: 12 May 2020 / Revised: 25 June 2020 / Accepted: 29 June 2020 / Published online: 7 August 2020

**Abstract** Automatic conversion from a computer-aided design (CAD) model to Monte Carlo geometry is one of the most effective methods for large-scale and detailed Monte Carlo modeling. The CAD to Monte Carlo geometry converter (CMGC) is a newly developed conversion code based on the boundary representation to constructive solid geometry (BRep → CSG) conversion method. The goal of the conversion process in the CMGC is to generate an appropriate CSG representation to achieve highly efficient Monte Carlo simulations. We designed a complete solid decomposition scheme to split a complex solid into as few nonoverlapping simple sub-solids as possible. In the complete solid decomposition scheme, the complex solid is successively split by so-called direct, indirect, and auxiliary splitting surfaces. We defined the splitting edge and designed a method for determining the direct splitting surface based on the splitting edge, then provided a method for determining indirect and auxiliary splitting surfaces based on solid vertices. Only the sub-solids that contain concave boundary faces need to be supplemented with auxiliary surfaces because the solid is completely decomposed, which will reduce the redundancy in the CSG expression. After decomposition, these sub-solids are located on only one side of their natural and auxiliary surfaces; thus, each sub-solid can be described by the intersections of a series of half-spaces or geometrical primitives. The CMGC has a friendly graphical user interface and can convert a CAD model into geometry input files for several Monte Carlo codes. The reliability of the CMGC was evaluated by converting several complex models and calculating the relative volume errors. Moreover, JMCT was used to test the efficiency of the Monte Carlo simulation. The results showed that the converted models performed well in particle transport calculations.

**Keywords** Monte Carlo · CAD · Geometry converter · Complete solid decomposition

## 1 Introduction

Constructive solid geometry (CSG) representation, which is beneficial to particle tracking, is commonly adopted in Monte Carlo particle transport codes. However, computer-aided design (CAD) systems typically use boundary representation (BRep) to describe solid models. In recent years, the BRep → CSG conversion method has been widely studied to improve the efficiency of large-scale and detailed Monte Carlo modeling. Several model conversion codes have been developed, such as SuperMC [1], McCad [2], and GEOMIT [3]. Great efforts have been made to reduce manual work, increase conversion speed, and improve conversion accuracy when developing these codes.

This work was supported by the National Natural Science Foundation of China (No. 11805017).

✉ Xin Wang
wang_xin@iapcm.ac.cn

[1] Institute of Applied Physics and Computational Mathematics, Beijing 100088, China

[2] CAEP Software Center for High Performance Numerical Simulation, Beijing 100088, China

[3] Department of Engineering Physics, Tsinghua University, Beijing 100084, China

[4] Nuctech Company Limited, Beijing 100084, China

Compared with deterministic methods, the Monte Carlo method has the disadvantage of being more time-consuming. Geometric calculations are frequently executed to determine the track length and particle location during particle tracking. For a highly complex geometry, the geometric calculations may take up most of the transport calculation time. The conversion method has a great impact on the computational complexity of the geometric calculations. Hence, not only the accuracy and speed of conversion but also the impact on geometric calculations should be considered.

A new BRep → CSG conversion code, called the CAD to Monte Carlo Geometry Converter (CMGC), was developed by Tsinghua University. A prototype of the CMGC was presented at the 13th China National Conference on Monte Carlo Method and Applications in 2017. Over the past three years, we have carried out a large number of practical engineering applications of the CMGC, improved the algorithms and functions based on these applications, and released a new version of the CMGC. At present, the CMGC has been successfully applied to the research of the China Fusion Engineering Test Reactor [4] and other large-scale and high-resolution Monte Carlo calculations. This paper provides a detailed introduction to the algorithms, implementations, and applications of the CMGC.

In the CMGC, a complete solid decomposition scheme is designed to split complex solids into as few nonoverlapping simple sub-solids as possible and to reduce the redundancy of auxiliary surfaces; thus, the CMGC can generate a better CSG expression to improve the simulation efficiency. In addition, the CMGC can parse STEP files and provide Monte Carlo geometry conversion functions and material-setting interfaces. BRep → CSG conversion in the CMGC currently supports planes, quadrics (such as spheres, cylinders, and cones), and tori. The CMGC can generate input files for the CSG of MCNP [5], combinatorial geometry of FLUKA [6] in the free format, JMCT [7], and MCShield [8], as well as geometry description mark-up language (GDML) files for GEANT4 [9] applications.

The remainder of this paper is organized as follows. In Sect. 2, we analyze the effect of different solid decomposition methods on particle tracking efficiency, introduce the methods of solid decomposition and auxiliary surface supplementation used in the CMGC, and describe the implementation scheme and main functions of the CMGC. In Sect. 3, validation and application of the CMGC are illustrated in several examples of CAD model conversions and JMCT calculations. In Sect. 4, the conclusions and future work dealing with spline surfaces are discussed.

## 2 Materials and methods

### 2.1 Effect of solid decomposition on particle tracking efficiency

The basic idea of the BRep → CSG conversion algorithm is to split a complex BRep solid into simple sub-solids, whose CSG expressions can be given by Boolean operations of half-spaces or geometrical primitives, and the complex solid can be represented as Boolean operations of these sub-solids. This procedure is often called solid decomposition. Different decomposition methods will generate different CSG expressions for the same solid, and these different expressions have a significant effect on geometric calculation efficiency.

In this study, $\cap*$, $\cup*$, and $-*$ represent regularized set operations of intersection, union, and difference, respectively. Figure 1 illustrates the split of solid $S$ in three ways (2D views are used in this section for simplification). The first decomposition method shown on the left is based on feature recognition technology. This method recognizes $S_1$ and $S_2$ as a slot and a hole, respectively, and represents solid $S$ as $S_{\text{outer}} -^* S_1 -^* S_2$. The second decomposition method shown in the middle uses splitting surfaces and splits solid $S$ into six nonoverlapping sub-solids, so that $S$ can be represented as $S_1 \cup^* S_2 \cup^* S_3 \cup^* S_4 \cup^* S_5 \cup^* S_6$. The third decomposition method shown on the right also uses splitting surfaces, but only two decomposition operations need to be performed, and represents solid $S$ as $S_1 \cup^* S_2 \cup^* S_3 \cup^* S_4$.

There are other decomposition methods based on splitting surfaces of solid $S$. Among the decomposition methods based on splitting surfaces, the third decomposition method can split solid $S$ into the least number of sub-solids, but whose number is still greater than that in the decomposition method based on feature recognition technology. It can be seen from this example that feature recognition technology

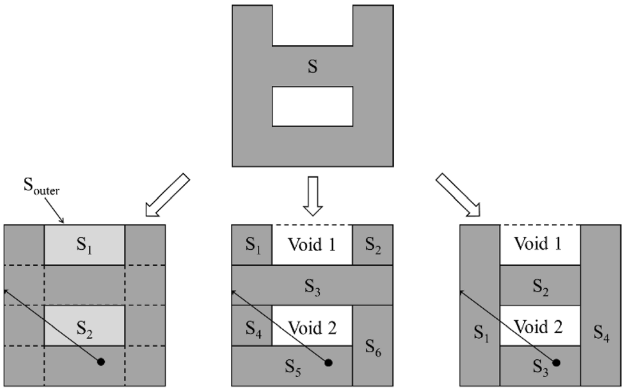


**Fig. 1** An illustration of three different decomposition methods of a solid $S$

can identify and extract features from the topological and geometrical information of the model for solid decomposition, thereby avoiding excessive decomposition of solids, enhancing the readability of CSG expressions, and reducing Boolean operations in the conversion process. In contrast, the decomposition method based on splitting surfaces can use only the union operations of the sub-solids to represent the original solid and will generate more sub-solids. The priorities of splitting surfaces will also affect the number of decomposition operations and sub-solids.

There are mainly two kinds of CSG representations used in Monte Carlo particle transport codes: half-spaces (e.g., MCNP) and geometrical primitives (e.g., GEANT4). The intersection and location calculations are the two most time-consuming geometric calculations during particle tracking. We use Fig. 1 as an example to analyze the effects of different solid decomposition methods on the intersection and location calculations in CSG representations of half-spaces and geometrical primitives. In the second and third decomposition methods, we represent the original solid as a collection of those sub-solids without using the union operations. Assuming that a particle has a collision-less free flight from the position shown by the black point in Fig. 1 to the left boundary of solid $S$, the solid boundaries that the intersection and location calculations should consider are those marked with solid and dashed lines. Solid $S$ has nine boundaries, while all the other solids and void regions have four boundaries. The intersection and location calculations determine the distances and relative positions between the particle and solid boundaries, respectively. We use $N_{\text{intersect}}$ and $N_{\text{locate}}$ to represent the total number of solid boundaries related to the intersection and location calculations along the particle trajectory, respectively. Moreover, the particle tracking efficiency is increased with a decrease in $N_{\text{intersect}}$ and $N_{\text{locate}}$. Assuming that neighbor lists for solids and surfaces have been created in the CSG representations of half-spaces, $N_{\text{intersect}}$ and $N_{\text{locate}}$ are as follows.

1. In the first decomposition method, the particle has a trajectory of $S \rightarrow S_2 \rightarrow S$. The number of solid boundaries related to the intersection calculation is nine, four, and nine, successively; hence, $N_{\text{intersect}}$ is 22. There is only one neighbor each time the particle crosses a boundary; hence, $N_{\text{locate}}$ is also equal to 22.
2. In the second decomposition method, the particle has a trajectory of $S_5 \rightarrow \textit{Void}\ 2 \rightarrow S_4 \rightarrow S_3$. There are two neighbors (i.e., $S_4$ and *Void* 2) to search for when the particle crosses the boundary of $S_5$. Thus, $N_{\text{intersect}}$ is 16 and $N_{\text{locate}}$ is 20.
3. In the third decomposition method, the particle has a trajectory of $S_3 \rightarrow \textit{Void}\ 2 \rightarrow S_1$. $N_{\text{intersect}}$ and $N_{\text{locate}}$ are both equal to 12.

In the CSG representations of geometrical primitives, there are no explicit expressions for void regions. Assuming that the solid $S$ is inside a world box with four boundaries, and an accelerating algorithm (such as smart voxelization [9] or octree) has been used so that only one candidate solid must be searched each time the particle enters a solid from a void region, $N_{\text{intersect}}$ and $N_{\text{locate}}$ are as follows:

1. In the first decomposition method, the particle has a trajectory of $S \rightarrow \textit{World box} \rightarrow S$. The numbers of solid boundaries related to the intersection calculation are nine, nine, and nine, successively; hence, $N_{\text{intersect}}$ is 27. Those related to the location calculation are nine, four, and nine, successively; hence, $N_{\text{locate}}$ is 22.
2. In the second decomposition method, the trajectory is $S_5 \rightarrow \textit{World box} \rightarrow S_4 \rightarrow S_3$, and $N_{\text{intersect}}$ and $N_{\text{locate}}$ are both equal to 16.
3. In the third decomposition method, the trajectory is $S_3 \rightarrow \textit{World box} \rightarrow S_1$, and $N_{\text{intersect}}$ and $N_{\text{locate}}$ are both equal to 12.

Clearly, the third decomposition method has the least $N_{\text{intersect}}$ and $N_{\text{locate}}$. When Boolean operations are more complex and collision events occur, this difference will be more significant. Under the same conditions of the particle tracking algorithm of the Monte Carlo simulation, the best decomposition method to improve simulation efficiency is to split the solid into as few nonoverlapping simple sub-solids as possible so that the CSG expressions of these sub-solids only need intersection operations. Towards this goal, a complete solid decomposition scheme was adopted by the CMGC.

### 2.2 Complete solid decomposition scheme

The complete solid decomposition scheme is based on splitting surfaces and designed to improve the particle tracking efficiency without considering the readability of the CSG expression. The first concern of the scheme is determining and sorting the splitting surfaces. The determination of splitting surfaces is related to whether each sub-solid only needs intersection operations in its CSG expression. The sorting of splitting surfaces determines whether the original solid is excessively decomposed. The second concern of the scheme is that some auxiliary surfaces need to be added to represent the CSG expression of the solid, but auxiliary surfaces also increase the complexity of intersection and location calculations. Searching for the best solutions to solid decomposition and auxiliary surface supplementation are both NP problems; therefore, we use heuristic strategies based on the topological and geometrical information in the BRep data.

(1) *Direct and indirect splitting surfaces.*

In the complete solid decomposition scheme, a solid is first split by the natural splitting surfaces generated from its boundary faces. These splitting surfaces are classified into direct and indirect splitting surfaces in the CMGC. The CMGC determines whether a surface is a direct splitting surface by the splitting edge.

*Splitting edge*: In solid $S$, assuming two boundary faces $F_1$ and $F_2$ are adjoined by an edge $E$, and $SF_1$ and $SF_2$ are the surfaces corresponding to $F_1$ or $F_2$, respectively, if solid $S$ is located on both sides of $SF_1$ or $SF_2$ in the infinitesimal neighborhood of any point on $E$, then $E$ is a splitting edge.

*Direct splitting surface*: In solid $S$, assuming a boundary face $F$ contains a splitting edge $E$, and $SF$ is the surface corresponding to $F$, if solid $S$ is located on both sides of $SF$, then $SF$ is a direct splitting surface.

The edges in the BRep data are classified into three types according to the angle between two adjacent boundary faces: convex (> 180°), concave (< 180°), and tangent (= 180°). As shown in Fig. 2a, the convex edge is clearly not a splitting edge. The concave edge is a splitting edge, and the two surfaces that correspond to the two boundary faces adjoined by the concave edge are both direct splitting surfaces (dashed lines in this figure represent splitting surfaces).

The tangent edge may be a direct splitting edge. Assuming that edge $E$ is a tangent edge and $SF_1$ and $SF_2$ are the surfaces corresponding to the two boundary faces adjoined by edge $E$, the CMGC determines whether $E$ is a splitting edge and whether $SF_1$ and $SF_2$ are direct splitting surfaces via the following steps:

1. If the sum of the curvatures of $SF_1$ and $SF_2$ is greater than 0, or if the normal directions of $SF_1$ and $SF_2$ on edge $E$ are identical, then edge $E$ is a splitting edge (Fig. 2b–j); otherwise, it is not a splitting edge (Fig. 2k–m).
2. If the normal directions of $SF_1$ and $SF_2$ on splitting edge $E$ are opposite, then both $SF_1$ and $SF_2$ are direct splitting surfaces (Fig. 2d, i, j).
3. If the normal directions of $SF_1$ and $SF_2$ on splitting edge $E$ are identical, then the one with larger curvature is the direct splitting surface (Fig. 2b, c, e–h).

Each splitting edge in a direct splitting surface indicates that the solid is located on both sides of the surface in a connected set of the Euclidean space, and these connected sets must be decomposed. Hence, the complete solid decomposition scheme adopts a heuristic strategy for sorting direct splitting surfaces. Before each decomposition, the complete solid decomposition scheme will first count the number of splitting edges contained in every direct splitting surface. Then, the direct splitting surface containing the most splitting edges will be used. As shown by the third decomposition method in Fig. 1, solid $S$ is decomposed successively by the two direct splitting surfaces containing the most splitting edges to obtain an optimal decomposition result. Then, the CMGC splits the solid with the indirect splitting surfaces.

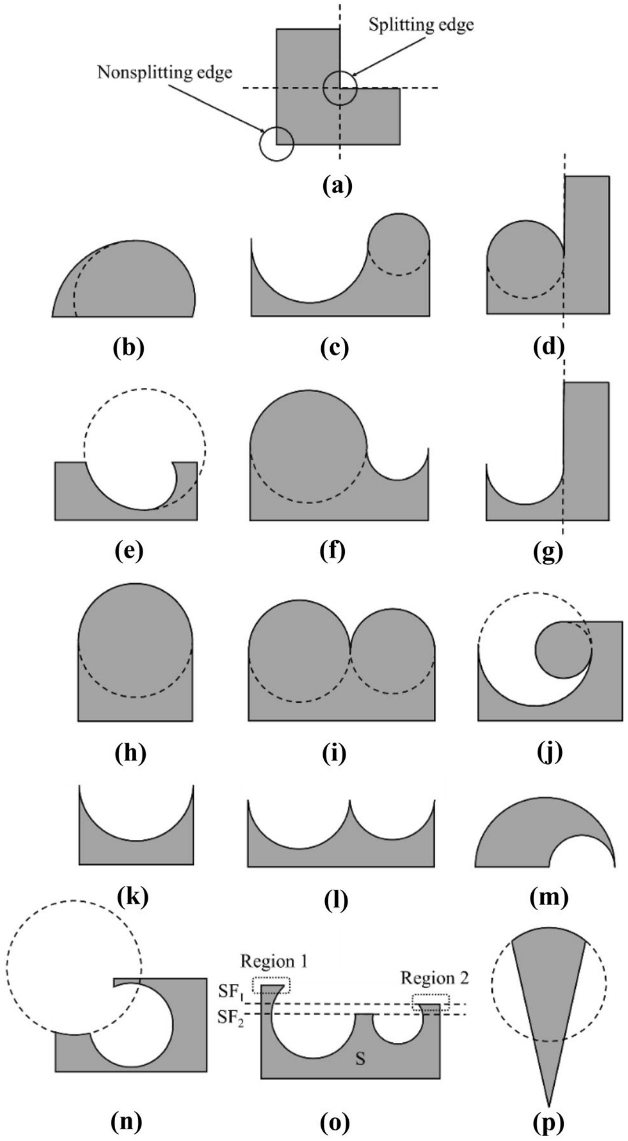


**Fig. 2** An illustration of the concept of splitting edges and natural splitting surfaces

*Indirect splitting surface*: In solid $S$, assuming a boundary face $F$ that does not contain any splitting edges, and $SF$ is the surface corresponding to $F$, if solid $S$ is located on both sides of $SF$, then $SF$ is an indirect splitting surface.

It can be seen that the solid is located on only one side of an indirect splitting surface in the infinitesimal neighborhood of any point on the corresponding boundary face; however, the solid will be located on both sides with the extension of the surface. Figure 2n, o, and p shows several typical indirect splitting surfaces. In the CMGC, the side of a boundary face located outside the solid is defined as the outside of the corresponding surface. An indirect splitting surface is determined by whether any solid vertices exist outside the surface. The priorities of indirect splitting surfaces will also affect the splitting quality. We used a heuristic strategy for sorting indirect splitting surfaces. The greater the number of solid vertices existing outside an indirect splitting surface, the higher the priority given to the surface. For example, there are two indirect splitting surfaces $SF_1$ and $SF_2$ in the solid $S$, as shown in Fig. 2o. Vertices outside of $SF_1$ are all located in region 1, while those of $SF_2$ are located in regions 1 and 2. Therefore, $SF_2$ has a higher splitting priority.

(2) *Auxiliary surface supplementation.*

If a solid contains curved boundary faces, then there are times when the CSG representation cannot be generated according to the natural surfaces corresponding to its boundaries. In this situation, some auxiliary surfaces should be supplemented [10,11]. A sufficient surface set for describing the CSG representation can be constructed by supplementing auxiliary surfaces for each curved boundary face of the solid [10]. However, some of the auxiliary surfaces are not necessary. Figure 3a shows a solid $S$ containing two curved boundary faces $F_1$ and $F_2$, and two auxiliary surfaces $SF_{a1}$ and $SF_{a2}$ are supplemented, respectively. $SF_{a1}$ is necessary because in the CSG representation, $S$ cannot be separated from *S'* by its natural surfaces. The surface set consisting of $SF_{a1}$ and the natural surfaces are sufficient to generate a CSG expression of $S_1$. $SF_{a2}$ is a redundant auxiliary surface that will increase the complexity of the CSG expression and decrease calculation efficiency.

In the complete solid decomposition scheme, each sub-solid is located on only one side of all its natural surfaces before the auxiliary surfaces are supplemented. The auxiliary surfaces are supplemented for only the concave boundary faces contained in the sub-solid, which is sufficient for generating a CSG expression.

Here, we explain the sufficiency of this auxiliary surface supplementation scheme based on the "canonical intersection term" and "describability theorem" proposed by Shapiro and Vossler [10]. We use the symbol $\prod$ to represent a canonical intersection term, $H$ to represent a half-space set, and $h$ to represent a half-space. If the half-space set corresponding to all boundary faces of sub-solid $S$ is $H_{\text{natural}} = \{h_1, h_2, ..., h_n\}$, then the canonical intersection term of set $H_{\text{natural}}$ is $\Pi_k = \widehat{h}_1 \cap *\hat{h_2} \ldots \cap *\hat{h_n}, \hat{h_i} \in \{h_i, \overline{h}_i*\},\ i = 1\ldots h$. There is only one canonical intersection term $\prod_S$ that includes sub-solid $S$, and all other canonical intersection terms are outside $S$, because $S$ is located on only one side of all its natural surfaces. Assuming that the boundary faces of $S$ are all convex faces or planes, the half-spaces in $\prod_S$ are all convex sets. The intersection of convex sets is a convex set, which means that $\prod_S$ is a convex set. $\prod_S$ is included in $S$ because a convex set is connected. Therefore, every canonical intersection term has the same classification with respect to sub-solid $S$, and the describability theorem is satisfied. In other words, $\prod_S$ is a CSG expression of $S$, and there is no need to supplement auxiliary surfaces when the boundaries of the solid only contain convex faces and planes. As shown in Fig. 3b, in the complete solid decomposition scheme, the CMGC will first split $S$ into $S_1$ and $S_2$, and then supplement auxiliary surface $SF_{a1}$ for only the concave boundary face $F_1$.

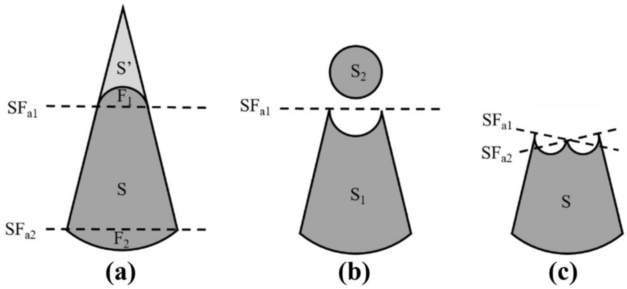


**Fig. 3** An illustration of the concept of auxiliary surfaces

The auxiliary surface may also be a splitting surface, called an auxiliary splitting surface, in the CMGC. Figure 3c shows an example in which two auxiliary splitting surfaces, $SF_{a1}$ and $SF_{a2}$, of a solid $S$ can split the solid. The method for determining auxiliary splitting surfaces is the same as that for indirect splitting surfaces. The outside of an auxiliary surface is defined as the opposite side where the corresponding boundary face is located.

*(3) Decomposition process.*

The process of the complete solid decomposition scheme is as follows. First, the solid is split using the direct splitting surfaces until no direct splitting surfaces exist, and any surface that has been used can no longer be an indirect splitting surface. Second, the solid is split using the indirect splitting surfaces until no indirect splitting surfaces exist; thus, each sub-solid is located on only one side of its natural surface. Third, the auxiliary surfaces are supplemented for only the concave boundary faces (if necessary), and the solid is split by the auxiliary splitting surfaces (if any exist). Finally, all the sub-solids are located on only one side of their natural and auxiliary surfaces. Thus, each sub-solid can be described by intersections of a series of half-spaces or geometrical primitives. If the void regions need to be converted, such as MCNP and FLUKA, then the void regions will be converted into solids by Boolean operations and split as normal solids. Figure 4 shows the decomposition process of the CMGC.

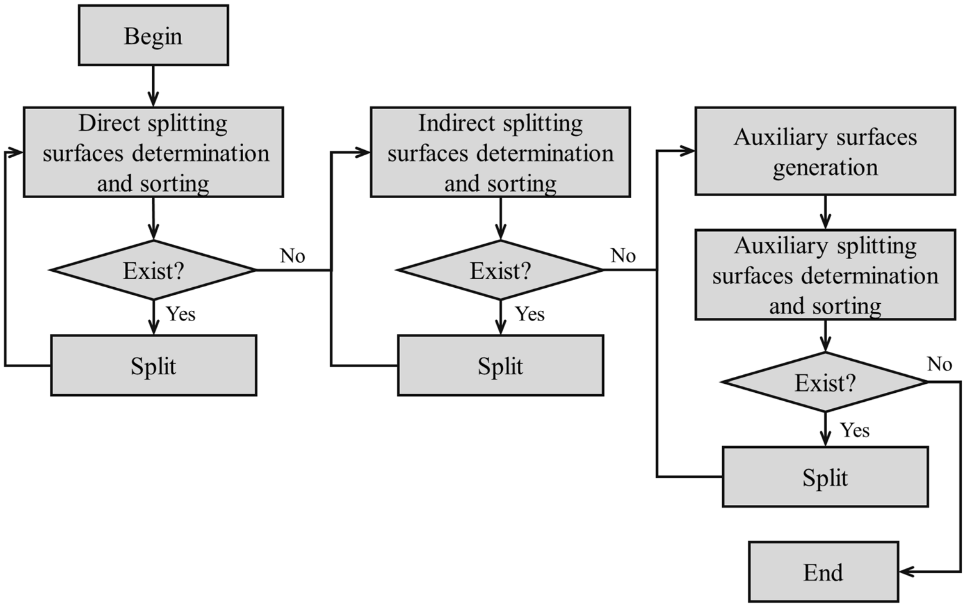

Begin
Direct splitting surfaces determination and sorting
Exist?
No
Yes
Split
Indirect splitting surfaces determination and sorting
Exist?
No
Yes
Split
Auxiliary surfaces generation
Auxiliary splitting surfaces determination and sorting
Exist?
No
Yes
Split
End

**Fig. 4** The decomposition process of the CMGC

*(4) CSG expression generation.*

After the decomposition is completed, the CSG expression for each sub-solid must be generated. Then, the CSG expression of the original solid can be determined according to the relationships between sub-solids. The sub-solids generated using the complete solid decomposition scheme do not overlap with each other, and the relationships between them are all union operations. However, as mentioned previously, we do not explicitly use the union operations for the benefit of particle tracking efficiency. The correlations between the original solid and the sub-solids are expressed by the name of the original solid, which will be introduced in Sect. 2.3. Therefore, only the CSG expression generation method for each sub-solid is needed in the CMGC.

For CSG representations of half-spaces, the method is quite simple. We illustrate the method with the CSG of MCNP. For each natural surface and auxiliary surface of a sub-solid, a half-space corresponding to the inside of the surface is created. The intersections of these half-spaces form a CSG expression of the sub-solid. In particular, for a two-sheet surface such as a conical surface, we will add an auxiliary surface to separate one sheet when using an MCNP GQ card.

For CSG representations of geometrical primitives, the method is more complex. We introduce the method with the CSG of GEANT4. Recognition algorithms for geometrical primitives of GEANT4, such as G4Box, G4Sphere, G4Tubs, and G4Para, are established in the CMGC. If a sub-solid is recognized as a geometrical primitive, then it will be directly converted without Boolean operations. Otherwise, a corresponding primitive is generated for each natural surface and auxiliary surface, and the sub-solid is represented by Boolean operations of these primitives. This will inevitably introduce some redundant surfaces and Boolean operations.

## 2.3 Functions and implementation

The conversion method needs to parse BRep data structures and perform a large number of 3D geometric calculations, such as convexity and concavity determinations, intersection calculations, Boolean operations, etc. The CMGC adopted the Open CASCADE Technology (OCCT) [12] to implement these functions. The main focus of this study is the research and implementation of the conversion method. The CMGC can convert Standard for the Exchange of Product Model Data (STEP) files into CSG representations of half-spaces or geometrical primitives and generate geometry input files for several Monte Carlo codes.

The graphical user interface (GUI) of the CMGC was developed based on the ST-Developer from STEP Tools [13]. The GUI has many functions, including STEP file display, assembly structure and attributes display, and material setting and conversion parameter setting for users. Figure 5 shows an accelerator model displayed on the GUI of the CMGC.

Users can define materials (nuclides, densities, etc.) by the material definition interface or import materials from an Excel file in a fixed format. The material assignment interface can then be used to set the correspondence between the solid names, which are parsed with assembly information by the CMGC, and the materials. All the material-setting information will be saved in the CMGC project file before conversion. In addition, since Boolean

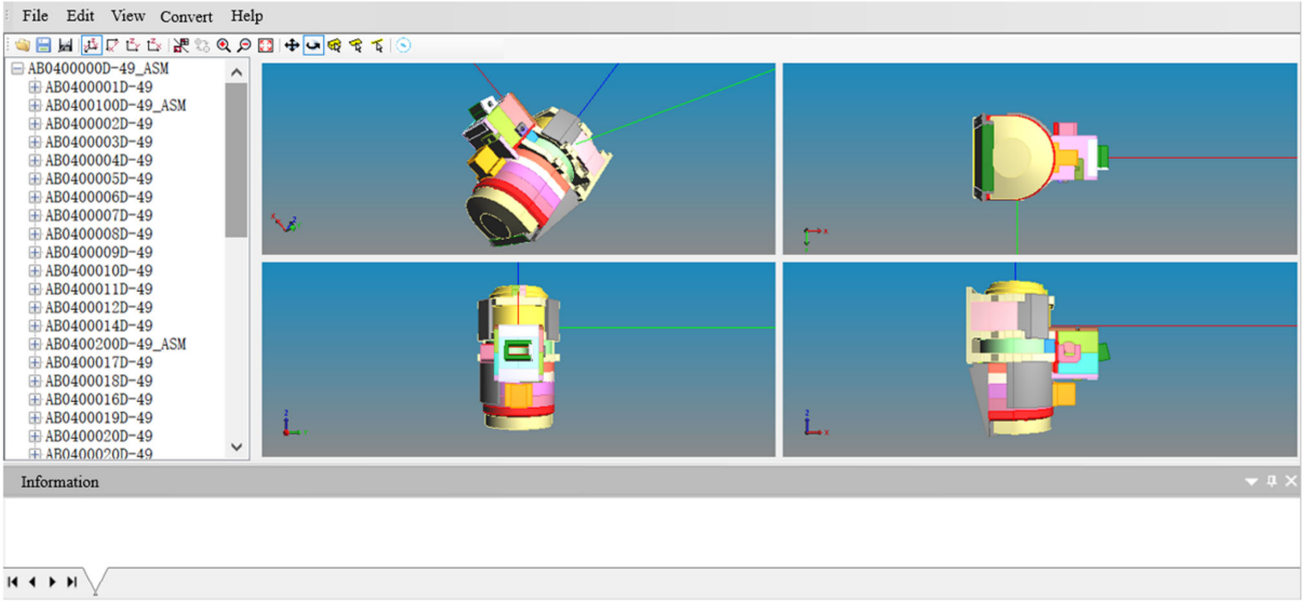


**Fig. 5** (Color online) GUI of the CMGC

operations in OCCT is unstable and sensitive to tolerances, the CMGC allows users to set tolerances for Boolean operations.

During the conversion process, the volume of each solid before and after the conversion and its relative error will be calculated and outputted. The solids with a relative error greater than the threshold, a failed Boolean operation, and successful conversions will be outputted to the corresponding STEP files. The names of the solids remain the same before and after the conversion process in all STEP files and will be outputted to the input files of the Monte Carlo codes. For general ASCII text files, the solid names will be outputted as comments, whereas the solid names will be used as the names of the logical volumes for GDML files. Therefore, after the conversion process is completed, users can set tallies, importances, and other information according to the solid names. Users can also determine the affiliations of the solids before and after the conversion according to the names of the solids. Moreover, the CMGC is parallelized using the shared memory standard OpenMP, which can improve the conversion efficiency when dealing with large-scale models.

The following five steps are taken by the CMGC during model conversion:

1. The assembly structure and attributes of the shapes are analyzed, and then, the materials are defined and assigned.
2. Information on repetitive structures is extracted, and the minimum solid set that must be converted is constructed.
3. Each solid is converted into a series of sub-solids using the complete solid decomposition scheme.
4. The CSG expressions of all solids are generated depending on these sub-solids and repetitive structures.
5. The Monte Carlo geometry input files including material information are outputted.

## 3 Results and discussion

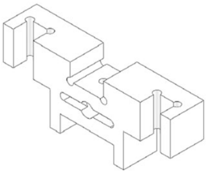
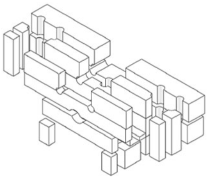
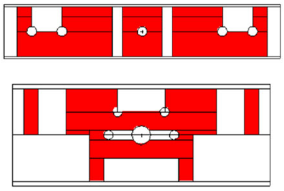
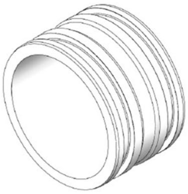
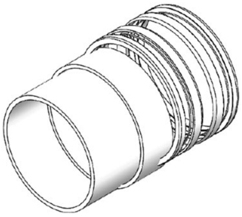
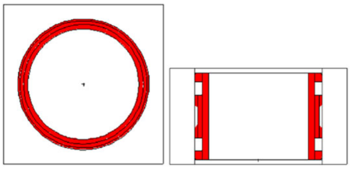
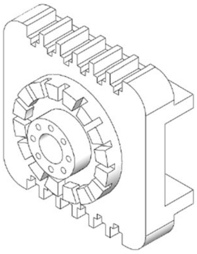
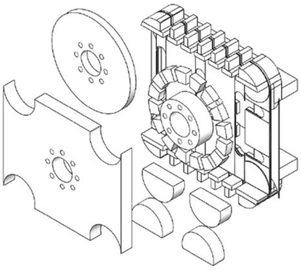
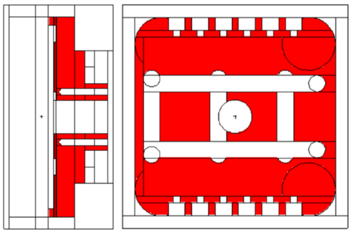
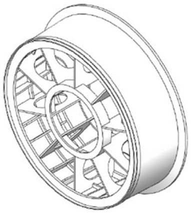
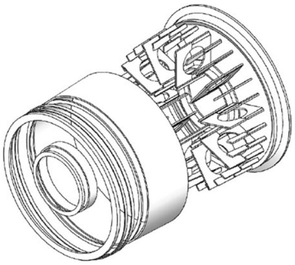
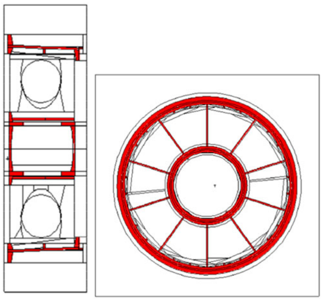
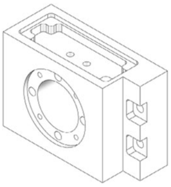
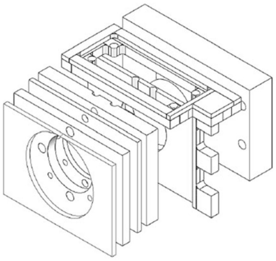
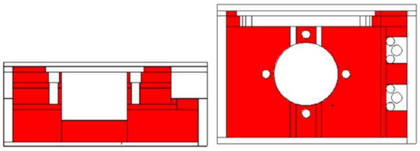
The reliability of the CMGC code was estimated through functional and application tests. Functional tests were performed to verify the conversion algorithm in the CMGC, and the testing STEP files came from ST-Developer. The application test included an accelerator model, optical component model, and a detailed building model. The relative volume error of the total converted models corresponding to the original STEP file was calculated to evaluate the conversion accuracy. Particle transport calculations were performed using JMCT to confirm the properties of the converted models in the application tests.

### 3.1 Functional tests

We chose several CAD models with various surfaces and features for functional tests, and both the models and corresponding void regions were converted. Surface types included planes, spheres, cylinders, cones, and tori. Features included chamfering, slots, and holes. The CAD models contained various splitting edges as well as direct, indirect, and auxiliary splitting surfaces.

Table 1 lists five models in the functional tests. The first column shows the original CAD models, the second

Table 1 Functional tests of the CMGC

| Model before decomposition | Model after decomposition | Section views of the converted MCNP input file |
|---|---|---|
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

column shows the decomposed models, and the third column shows the 2D section views of the converted MCNP input files. The CMGC generated the correct CSG expressions for these models, and the maximum relative volume error was 0.00051% (including void regions). It is obvious from the decomposition results that the heuristic strategies adopted by the complete solid decomposition scheme can prevent solids from being excessively decomposed.

### 3.2 Conversion of an accelerator model

An accelerator model, which consists of 54 irregular solids and various curved surfaces, was converted into input files of MCNP and GEANT4 using the CMGC. As mentioned above, OCCT are unstable and sensitive to tolerance; thus, we tried several different tolerance settings. First, we set a tolerance of 0.0001 for all the Boolean operations of all solids. During the conversion process, a

Boolean operation of a solid in the model failed. We then changed the tolerance of that Boolean operation to 0.001, and all solids were converted successfully. If we set a tolerance of 0.01 for all the Boolean operations of all solids, then two Boolean operations had relative volume errors greater than 5%. A similar situation occurred during practical applications of the CMGC. According to the user manual of OCCT, the tolerance (fuzzy option in OCCT) should be slightly larger than the value of the gap or the embedding depth between the entities in STEP files. Gaps or embeddings sometimes exist in STEP files because of mistakes in CAD modeling, the conversion between different CAD systems, or precision loss. Therefore, users of the CMGC can adjust the tolerance (depending on the CAD models to be converted and their engineering experience) when a failed Boolean operation or a relative volume error greater than the threshold occurs.

Figure 6a shows the 3D view of the accelerator CAD model. After all conversions were successfully performed, the solids of the model were split into 339 sub-solids by the CMGC and the solid conversion time was 39.9 s. The void region was converted into 605 sub-solids with conversion time of 72.4 s. The relative volume error of the conversion was 0.00095%. Figure 6b shows the visualization of the converted GEANT4 GDML file. Figure 6c and d shows section views of the CAD model and the converted MCNP input file at the same location, respectively.

## 3.3 Conversion and calculation of an optical component model

An optical component model was used to evaluate the particle transport efficiency of the converted model and demonstrate the effects of different solid decomposition methods on the particle tracking efficiency. As shown in Fig. 7a, the optical component model consists of 17 solids. The original model was manually built in JLAMT [14] using the method shown on the left in Fig. 1 to reduce the modeling complexity and enhance the readability. For example, a solid of the manually built model illustrated in Fig. 7b was constructed from 20 geometrical primitives using 10 union operations and 9 difference operations. The JMCT input file and STEP file were exported from JLAMT. The STEP file was converted into a new JMCT input file containing 188 sub-solids within 27 s by the CMGC. The relative volume error of the conversion was 0.00024%.

As shown in Fig. 7c, the solid corresponding to Fig. 7b was converted into 42 simple sub-solids by the complete solid decomposition scheme, and most were geometrical primitives without any Boolean operations. As discussed in Sect. 2.1, this decomposition method reduced the frequencies of intersection and location calculations. Furthermore, accelerating particle tracking algorithms by using bounding-boxes or an octree is more effective for a series of simple sub-solids than a single complex solid. Therefore, the converted model is expected to be more efficient than the manually built model for the geometric calculation.

The two JMCT input files were used in the neutron transport calculations by JMCT. A total of 1000 meshes were uniformly distributed in the model space, and the neutron fluxes were tallied. The number of particle histories was $1 \times 10^7$, and the relative deviations of 1000 tallies were all less than 4%. The simulation time of the manually built model was 2908 s, while that of the converted model was 745 s. The durations of the physical calculation and tallying were basically the same, but the geometric calculation time of the converted model was seven times shorter than that of the manually built model. The correspondence between the neutron fluxes and computation time of the physics and tally verified the correctness of the converted model. The significant decrease in the geometric calculation time indicated that the complete solid decomposition

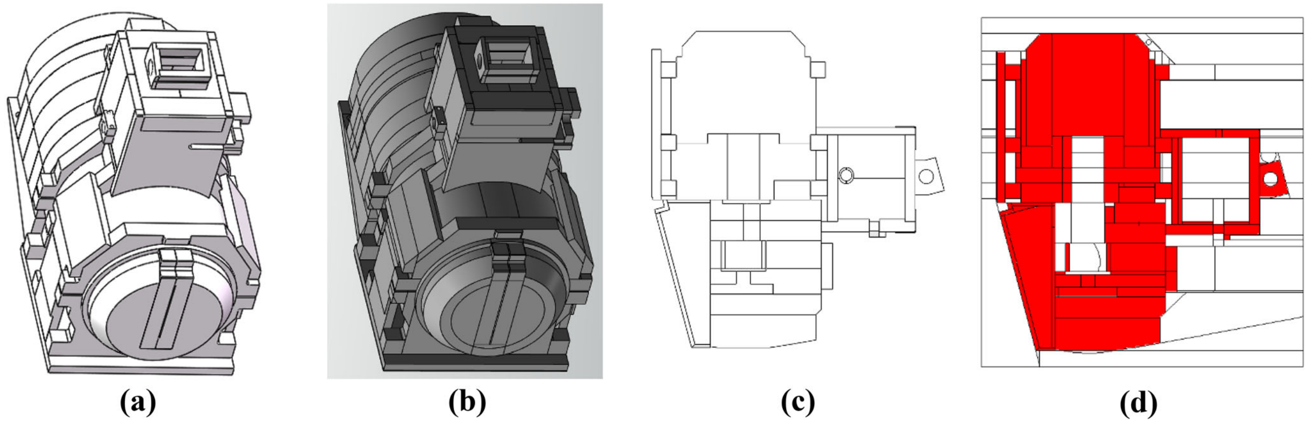


**Fig. 6** (Color online) Accelerator model

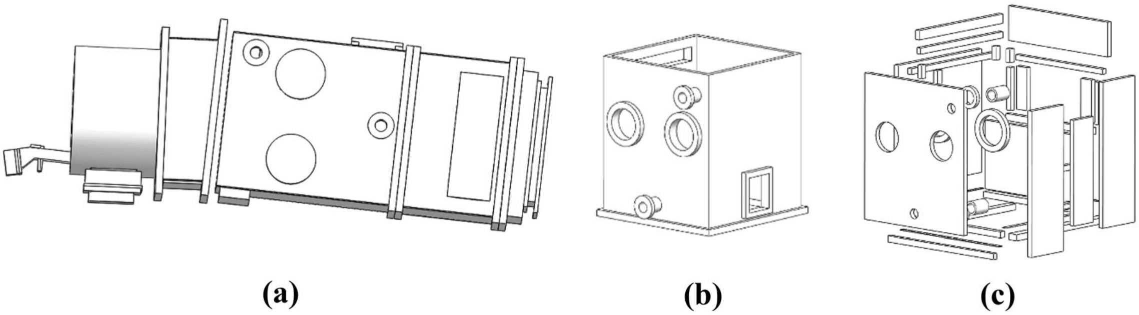


**Fig. 7** Optical assembly model

scheme adopted by the CMGC is better at improving particle tracking efficiency than the method shown on the left in Fig. 1.

### 3.4 Conversion and calculation of a detailed building model

A detailed building model was converted and calculated. Figure 8a shows the external view and Fig. 8b shows the internal structure. The model consists of 2156 solids that can be classified into four materials: brick wall, concrete, metal, and glass. The model was split into 4239 sub-solids and converted into a JMCT geometry input file within 1126 s. The relative volume error of the conversion was 0.00003%. The converted model is shown in Fig. 8c.

Monte Carlo simulation was performed by JMCT using the converted model. The red dot in Fig. 8c shows the location of the particle source. The model space surrounded by the blue lines was uniformly divided into 1 × $10^6$ meshes. The neutron fluxes of all the meshes were tallied, and the number of particle histories was 1 × $10^9$. The simulation used 200 processors on the Tianhe-2 supercomputer and took approximately 10 min. The fluxes of four sections indicated by the green lines in Fig. 8c are shown in Fig. 9. The results were in accordance with the physical analysis.

## 4 Conclusion

The conversion code CMGC was developed to generate an appropriate CSG representation that can improve the Monte Carlo simulation efficiency. A complete solid decomposition scheme was implemented in the CMGC. The goals of this scheme were to split complex solids into as few nonoverlapping simple sub-solids as possible and to reduce the redundancy of auxiliary surfaces. We used the splitting edge and the direct, indirect, and auxiliary splitting surfaces to convert complex solids. We determined that generating auxiliary faces for sub-solids that contain only convex faces and planes is unnecessary. CMGC provides setting interfaces (for materials, tolerances, and other parameters), conversion error analysis, model display, and STEP file processing. These functions are convenient for users who perform conversion monitoring and analysis of the obtained Monte Carlo input file. Several complex models were used for conversion tests and particle transport calculations. The converted models showed good accuracy and calculation efficiency, confirming the reliability of the CMGC.

However, there are still some challenges in BRep → CSG conversion. First, spline surfaces cannot be converted directly to CSG representation. They can be converted into unstructured meshes, but the accuracy and efficiency of the simulation will be affected, and there may be problems

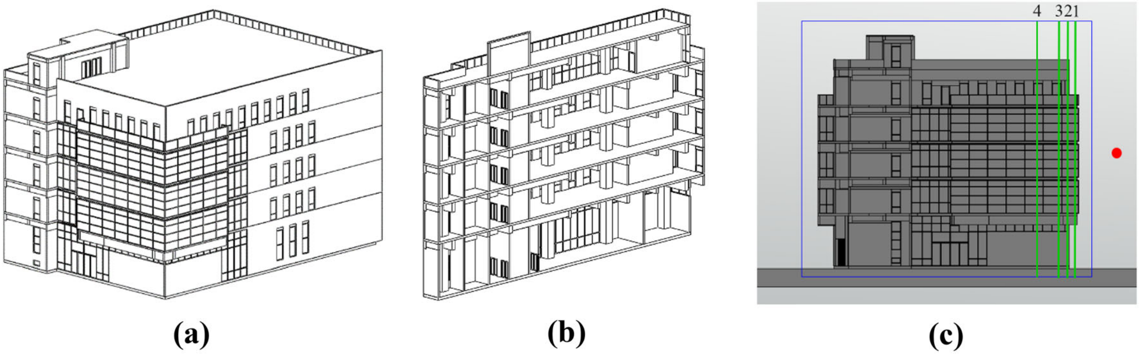
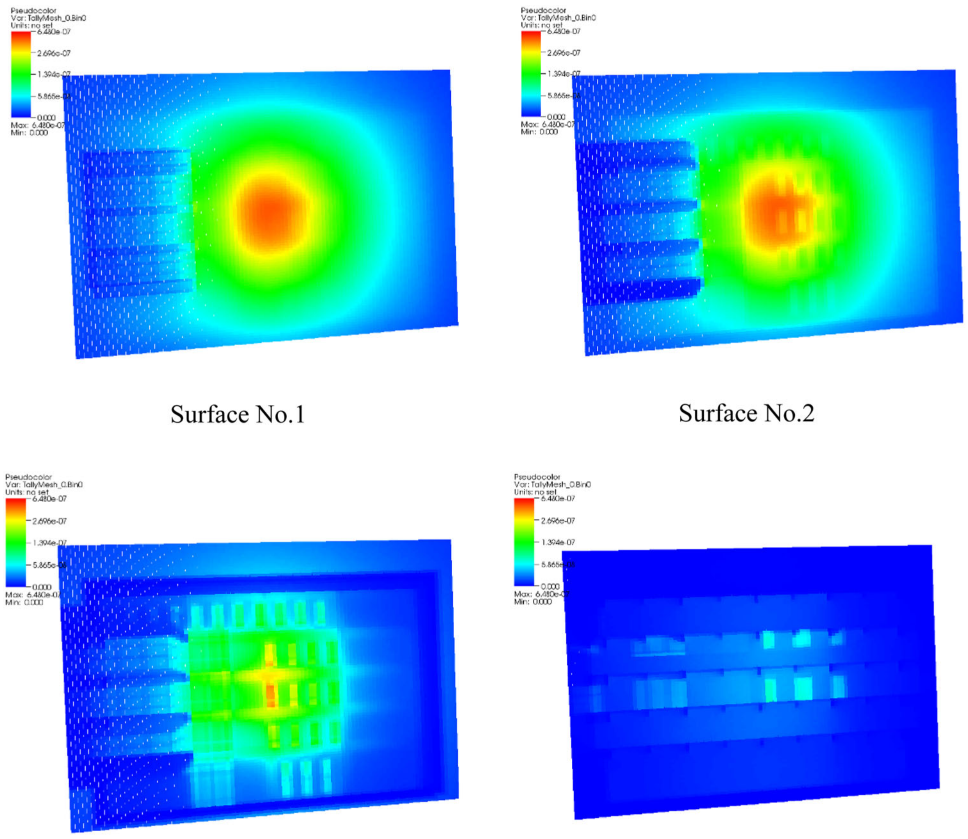


**Fig. 8** Detailed building model



Surface No.1

Surface No.2

Surface No.3

Surface No.4

**Fig. 9** (Color online) Simulation results of the detailed building model

with gaps and overlaps. Second, the Boolean operation functions provided by the modeling engine are unstable. Boolean operations are sensitive to tolerance, and sometimes the proper tolerance to obtain the correct conversion result for a solid cannot be determined. In the future, we will try to contact OCCT to seek further support for stable Boolean operations. We will also focus on reconstructing the CMGC based on JLAMT, which was developed on Siemens NX [15], and on implementing a Monte Carlo modeling method combining automatic conversion and visual modeling.

## References

1. J. Song, G. Sun, Z. Chen et al., Benchmarking of CAD-based SuperMC with ITER benchmark model. Fusion Eng. Des. **89**(11), 2499–2503 (2014). https://doi.org/10.1016/j.fusengdes.2014.05.003
2. L. Lu, U. Fischer, P. Pereslavtsev, Improved algorithms and advanced features of the CAD to MC conversion tool McCad. Fusion Eng. Des. **89**(9–10), 1885–1888 (2014). https://doi.org/10.1016/j.fusengdes.2014.05.015
3. S. Sato, H. Nashif, F. Masuda et al., Progress of conversion system from CAD data to MCNP geometry data in Japan. Fusion Eng. Des. **85**(7–9), 1546–1550 (2010). https://doi.org/10.1016/j.fusengdes.2010.04.038
4. X.M. Shi, X. Wang, G.M. Qin et al., Neutronics conceptual research on a hybrid blanket of China fusion engineering test reactor. Fusion Eng. Des. **156**, 111715 (2020). https://doi.org/10.1016/j.fusengdes.2020.111715
5. T. E. Booth, F. B. Brown, J. S. Bull, et al., MCNP5 1.50 Release Notes, LA-UR-08–2300, Los Alamos National Laboratory (2008).
6. T.T. Böhlen, F. Cerutti, M.P.W. Chin et al., The FLUKA code: developments and challenges for high energy and medical applications. Nucl. Data Sheets **120**, 211–214 (2014). https://doi.org/10.1016/j.nds.2014.07.049
7. L. Deng, Z. Hu, G. Li, et al., 3-D Monte Carlo neutron-photon transport code JMCT and its algorithms. Presented at PHYSOR 2014, Kyoto, Japan, September 28–October 3 (2014).
8. S. Gao, Z. Wu, X. Wang, et al., Development of a radiation shielding Monte Carlo Code: RShieldMC. Presented at International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering 2017, Jeju, Korea, April 16–20 (2017).
9. J. Allison, K. Amako, J. Apostolakis et al., Geant4 developments and applications. IEEE Trans. Nucl. Sci. **53**(1), 270–278 (2006)
10. V. Shapiro, D.L. Vossler, Efficient CSG representations of two-dimensional solids. J. Mech. Des. **113**(3), 292–305 (1991)

11. V. Shapiro, D.L. Vossler, Separation for boundary to CSG conversion. ACM Trans. Graphics **12**(1), 35–55 (1993)
12. Open CASCADE Technology; https://www.opencascade.com/.
13. STEP Tools; https://www.steptools.com/.
14. Y. Ma, Y. Fu, G. Qin et al., Design and development of automodeling tool JLAMT for field application of large-scale models. Chin. J. Comp. Phys. **33**(5), 606–612 (2016). (in Chinese)
15. Siemens; https://www.plm.automation.siemens.com/global/en/products/nx/.