Characteristics of 3D Solid Modeling Software Libraries for Non-Manifold Modeling

5 authors, including:

Aikaterini Chatzivasileiadi
Cardiff University
29 PUBLICATIONS 212 CITATIONS

SEE PROFILE

Nicholas Mario Wardhana
Cardiff University
11 PUBLICATIONS 47 CITATIONS

SEE PROFILE

Wassim Jabi
Cardiff University
97 PUBLICATIONS 466 CITATIONS

SEE PROFILE

Robert Aish
University College London
53 PUBLICATIONS 922 CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Re-energising Wales View project

BIM - Blockchain Integration View project

Characteristics of 3D Solid Modeling Software Libraries for Non-Manifold Modeling

Aikaterini Chatzivasileiadi[1] , Nicholas Mario Wardhana[2] , Wassim Jabi[3] , Robert Aish[4] and Simon Lannon[5]

[1]Cardiff University, ChatzivasileiadiA@cardiff.ac.uk
[2]Cardiff University, WardhanaN@cardiff.ac.uk
[3]Cardiff University, JabiW@cardiff.ac.uk
[4]University College London, Robert.Aish@ucl.ac.uk
[5]Cardiff University, Lannon@cardiff.ac.uk

Corresponding author: Aikaterini Chatzivasileiadi, ChatzivasileiadiA@cardiff.ac.uk

ABSTRACT

The aim of this paper is to provide a review of the characteristics of 3D solid modeling software libraries – otherwise known as 'geometric modeling kernels' in non-manifold applications. 'Non-manifold' is a geometric topology term that means 'to allow any combination of vertices, edges, surfaces and volumes to exist in a single logical body'. In computational architectural design, the use of non-manifold topology can enhance the representation of space as it provides topological clarity, allowing architects to better design, analyze and reason about buildings. The review is performed in two parts. The review is performed in two parts. The first part includes a comparison of the topological entities' terminology and hierarchy as used within commercial applications, kernels, and within published academic research. The second part proposes an evaluation framework to explore the kernels' support for non-manifold topology, including their capability to represent a non-manifold structure, and in performing non-regular Boolean operations, which are suitable for non-manifold modeling.

1   NON-MANIFOLD TOPOLOGY

1.1   Definition

Mathematically, Non-Manifold Topology (NMT) is defined as cell complexes that are subsets of Euclidean Space [38]. Practically, topology refers to the spatial relationships between the various

entities in a model and it describes how geometric entities are connected [53]. 'Non-manifold' is a geometric topology term that means 'to allow any combination of vertices, edges, surfaces and volumes to exist in a single logical body' [22]. Such models allow multiple faces meeting at an edge or multiple edges meeting at a vertex. Coincident edges and vertices are merged. Moreover, non-manifold topology models have a configuration that cannot be unfolded into a continuous flat piece and are thus non-manufacturable and not physically realizable [4]. On the contrary, a manifold body without internal voids can be fabricated out of a single block of material [1]. In addition, in a manifold object if one were to draw spheres centered on the points of the object's surface, these would be divided into two pieces; one inside and one outside the object [55]. Examples of manifold and non-manifold geometry are shown in Fig. 1 and further information on the difference between manifold and non-manifold modeling are provided in [10].

Non-manifold supports a form of modeling, which removes constraints traditionally associated with manifold solid modeling forms by embodying all of the capabilities of wireframe modeling, surface modeling and solid modeling forms in a unified representation and extending the representational domain beyond that of the above modeling forms [59]. One of the very first ideas proposed by Kjellberg [30] and his team in the late 1970s was that, having a unified modeling framework with a mix of different representation techniques would allow the user to represent different stages and levels of models. Non-manifold modeling in $\mathbb{R}^3$ can be considered as exhaustively decomposing the $\mathbb{R}^3$ into disjoint sets of elements of zero, one, two, and three dimensional point sets, i.e., vertices, edges, faces, and regions respectively [13], [16].
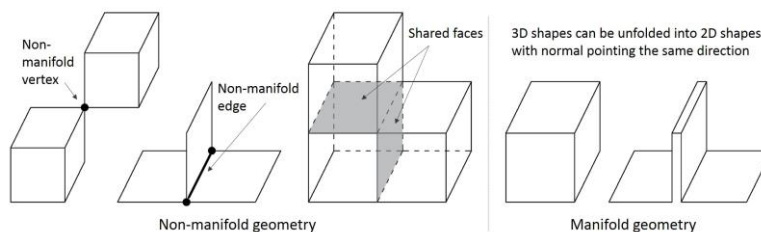

Figure 1: Examples of manifold and non-manifold geometry.

1.2 Applications

NMT has been successfully applied in the ship building industry, the medical field, architectural design, 3D modeling, computer-aided engineering analyses, as well as digital fabrication exploration.

Ship building industry: NMT has been successfully used in the ship-building for 'compartment' design, that is how the overall hull is subdivided into enclosed potentially watertight spaces, and the representation of these subdivision as a complex hull structure [35]. In this field, the use of NMT allows designers to segment a complex overall form into more cellular zones and spaces in a consistent manner.

Medical field: NMT has been successfully used in the medical field to model complex organic structures with multiple internal zones [7], [42]. In these applications, a non-manifold mesh is often generated from a Magnetic Resonance Imaging (MRI) scan of human organs as a basis for further analyses, for example using the Finite Element Method (FEM).

3D modeling: Non-manifold geometric models can maintain additional data, which may not appear in the resultant shape. This is one of their most useful characteristics, as it allows hybrid representation, including characteristics of both Constructive Solid Geometry and Boundary Representation 3D modeling [39].

Computer Aided Design (CAD)/Computer Aided Engineering (CAE)/Computer Aided Manufacturing (CAM): NMT can also be used in engineering analyses, as it can handle cellular structures and abstracted mesh models [34]. Some studies [51], [52] have used non-manifold

topology for structural modeling and finite element analyses and others have demonstrated the partitioning of cells leading to cellular representations which are then used for structural analyses [43]. Moreover, non-manifold spatial models are considered to be suitable for early structural analysis, as horizontal and vertical edges can be used to define beams and columns respectively, while internal or external faces can be used to define floors, roof elements and interior or exterior walls, facades and partitions [1]. The opportunities offered and limitations overcome by non-manifold systems in CAD/CAE/CAM applications have been reported in various studies. For example, Lipson and Shpitalni [36] introduced a topology invariant while providing a basis for a modeling system for sheet metal parts. With this invariant, it is possible to query manufacturing processes, such as number of components and configuration of bend lines and weld lines, using a single qualitative model of the product. This capability is particularly useful in the early stages of the design. Mikchevitch and Pernot [40] studied typical issues when transferring non-manifold 3D models between CAD systems and proposed a methodology for reconstructing 3D partitions that might be lost from exported non-manifold models, which they then successfully validated on academic and industrial models. Vivodtzev et al. [57] introduced a method for topology preserving mesh simplification, which ensures the consistency of CAD/CAM models. Lee [33] established a more integrated environment for the design and analysis of plastic injection moulding parts through the development of a feature-based design system based on a non-manifold modelling kernel supporting feature-based multi-resolution and multi-abstraction modelling capabilities. In that system, the CAD and CAE systems work under a single model in a NMT schema and for design changes, the design and analysis models are modified simultaneously.

Digital Fabrication: the potential of NMT has been investigated in terms of the design and additive manufacturing of conformal cellular structures [25]. It was found that the consideration of topology and more importantly the establishment of topological queries could improve the efficiency of their design.

Architectural design: Considering the above applications, it would be possible to transfer NMT's success from the ship-building and the medical fields to architecture in order to enhance the representation of architectural space [23]. The complexity of a ship, including its scale and its spatial organization, or that of an organic structure, including its multiple internal zones, could be well compared to the complexity of a building. One of the advantages of the NMT technique is the topological clarity that it provides, which allows architects to better design, analyse, reason about, and produce their buildings. The potential of NMT in the early design stages is already acknowledged and research has been undertaken with regard to the advantages of NMT's application for energy analysis in the early design stages [22], [24]. NMT has already been applied together with parametric and associative scripting to model the spatial organization of a building [1]. This information was then used to create different analytical and material models of a building.

2   TOPOLOGICAL CHARACTERISTICS OF NON-MANIFOLD OBJECTS

2.1   Topological Elements and Data Structures

Topological elements of non-manifold objects are hierarchically interrelated and a lower-dimensional element is used as the boundary of each of several higher dimensional ones [59]. A topological element does not include its own boundary [38]. An example of a hierarchical structure of non-manifold topological elements is presented in Fig. 2(a). The bottom-most element is a vertex, which geometrically equals to a point. A vertex can be detached from other structures, or located at the ends of an edge, which implies a line. This, however, is the end of the similarity with the traditional surface boundary representation, as open and closed loops can be created from the detached as well as inter-connected vertices and edges. Loops can be integrated to produce a face/surface, while faces can be combined to build a shell. It should be noted, however, that a shell may contain isolated vertices, edge, and faces. Then, a volume can be generated from a set of adjacent shells. A complex finally lies at the top of the hierarchy. It can constitute any other elements including volumes, faces, edges, and vertices.
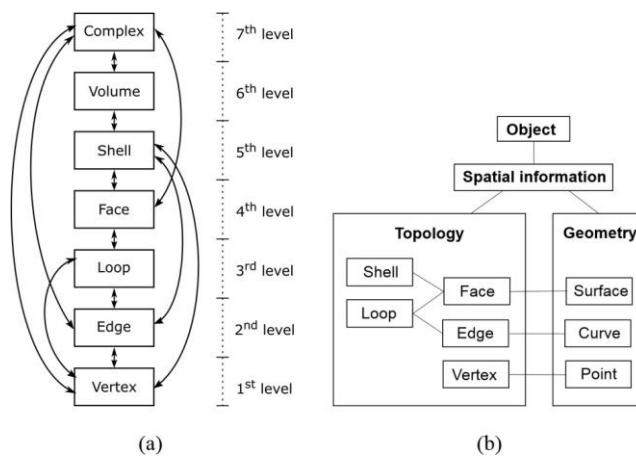
Figure 2: (a) An example of hierarchical structure of non-manifold topological elements [38], (b) Basic data structure, adapted from [56].

Geometric algorithms involve the manipulation of objects, which are not handled at the machine language level. The user must therefore organize these complex objects by means of the simpler data types directly representable by the computer. These organizations are universally referred to as data structures, which are ways to organize information, which, in conjunction with algorithms, permit the efficient and elegant solution of computational problems [46]. A basic data structure is presented in Fig. 2(b). Expanded data structures and topological relationships allow for a richer representation of loci, centrelines, elements, surfaces, volumes and hierarchical structures that are usually found in architectural compositions.

2.2 Boolean Set Operations for Non-Manifold Objects

Boolean set operations are common set operations that are used to combine solids in order to create more complex objects. They are usually applied to two bodies at a time [3]. The main Boolean operations are union, intersection and difference and can be regular or non-regular. In the union operation, the resulting solid occupies the space previously occupied by all the original solids. In the intersection operation, the resulting solid occupies the space previously occupied simultaneously by all the original solids. In the difference operation, the resulting solid occupies the space previously occupied by one of the original solids that the other solids did not occupy [3]. Generally a regular Boolean operation removes any external faces of the input bodies that are within the resulting body, while a non-regular Boolean operation maintains any external faces of the input bodies that are within the resulting body [1]. As a result, regular operations lead to a manifold result, while non-regular operations lead to a non-manifold result. The manifold or the non-manifold property of the output body cannot be informed by the input bodies' property, as manifold and non-manifold inputs can lead to manifold or non-manifold outputs, but not respectively [3]. This is also observed in Fig. 3, which shows the result of different regular or non-regular Boolean operations with manifold inputs. As observed, some operations (union and intersection) are symmetric, while other operations (difference, impose) are asymmetric. With asymmetric operations there is the convention that the inputs are referred to as the 'part' (A) and the 'tool' (B) respectively [1]. As an example, in the impose operation, all parts of A which are within the region of B are removed. Similarly, in the imprint non-regular operation, all parts of B that are not within A are removed. Both these operations lead to non-manifold results.

	UNION	INTERSECTION	DIFFERENCE	MERGE	IMPOSE	IMPRINT	SLICE
MANIFOLD INPUTS							
REGULAR OPERATIONS	1 body			N/A	N/A	only imprint edges on external faces, no cells created	multiple bodies
NON-REGULAR OPERATIONS	N/A	N/A	N/A	1 body with multiple cells		imprint external faces of B within A, creating cells	

Figure 3: The result of different regular or non-regular Boolean operations with manifold inputs (adapted from [1]).

3 METHODOLOGY

The intention of this research was to review academic literature and geometric modeling kernels supporting non-manifold topology. A geometric modeling kernel is a 3D solid modeling software library that provides geometric and topological data structures, as well as algorithms to model an architectural space, a building or an artefact. The study included the investigation of the topological entities used in thirteen data structures, as well as in other proposed class hierarchies suitable for non-manifold modeling, in terms of the levels they support and the terminology used for each level (Section 4.1). In addition, twelve geometric modeling kernels that support NMT have been evaluated (Section 4.2) and they have been assessed according to three characteristics, namely their license types, including whether they are commercial or open source (Section 4.2.1); the topological entities hierarchy they support (Section 4.2.2); and the supported topological operations (Section 6). Inconsistencies were expected to be found regarding the terminology and the supported levels in both the academic research and the kernels, and thus the aim of this research is two-fold; first, to summarize the academic overview and the review of the modeling kernels in a new terminology and class hierarchy standard (Section 5) and then to also propose a testing framework to assess the kernels' support for non-manifold structures based on the new terminology (Section 6). More specifically, the tests, using the Open CASCADE Technology (OCCT) kernel, aim to identify the provided structural representations of a non-manifold structure and operations involving these structures. OCCT 7.2.0 64-bit was used due to its status as one of the most outstanding open source geometry kernels with well-established community base, and its advertisement of non-manifold support. In addition, OCCT was found to provide a richer environment to work in, compared with other open-source kernels, as it provides a higher level of entities, which allows flexibility and versatility in its use.

4   REVIEW ON ENTITIES' TERMINOLOGY IN NON-MANIFOLD MODELING

4.1   Academic Research

The advantages of non-manifold representation have been recognized in various studies, such as [16], [39] and several representation schemes have been proposed for 3D modeling. This section focuses on the review of the research papers whose authors proposed a non-manifold class hierarchy or used an existing one. However, it is acknowledged that some of these frameworks have been based on precursors that are suitable for manifold modeling. The review included the following publications.

- Weiler's [60] radial edge structure
- Rossignac and O'Connor's [50] Simplicial Geometric Complexes
- Gursoz et al.'s [17] vertex-based data structure
- Yamaguchi and Kimura's [62] coupling entities data structure
- Cavalcanti et al.'s [8] Complete Geometric Complexes
- Lee and Lee's [34] partial edge structure
- Karasick's [26] star-edge boundary representation
- Higashi et al.'s [20] cycle structure
- De Floriani and Hui's [14] non-manifold indexed data structure with adjacencies
- Hui and De Floriani's [21] incidence simplicial data structure
- Hachenberger et al.'s [18] SNC (Nef complex) structure,
- Zeng et al.'s [63] Q-complex data structure
- Boguslawski and Gold's [6] dual half-edge data structure

The review also included other researchers, who either proposed their own hierarchy, such as [1], [15], [32], [36], [39], [45], [47], adopted an existing one [5], [11], [22], [32], [38], [41], [45], [49] or proposed extra structures in existing data structures, such as in Luo and Lukacs' [37] work, which removed the ambiguities[1] found in the radial-edge structure. A matrix regarding the entities' terminology used in the above studies can be found in the Appendix A (Tab. 5).

Considering the above, various entity names have been used considering different topological frameworks, which are as follows.
- complex, body, model or group for the higher level of the hierarchy for the 8th level;
- solid, primitive or component for the 7th level;
- volume, region or cell for the 6th level;
- shell, polyhedron or tetrahedron for the 5th level;
- face, dangling face or facet for the 4th level;
- loop, wire, genus or cycle for the 3rd level;
- edge or wire-edge for the 2nd level; and
- vertex for the 1st level.

The number of studies (academic papers) that use each of the aforementioned entity names are presented in Fig. 4. It is seen that vertex is used in all studies, while a set of basic elements [61] including vertex (1st level), edge (2nd level), loop (3rd level), face (4th level) and shell (5th level) is shared in almost every scheme. The higher levels in the topological hierarchy present larger diversity and it seems that region, solid and complex is the preferred terminology for the 6th, 7th and 8th level respectively.

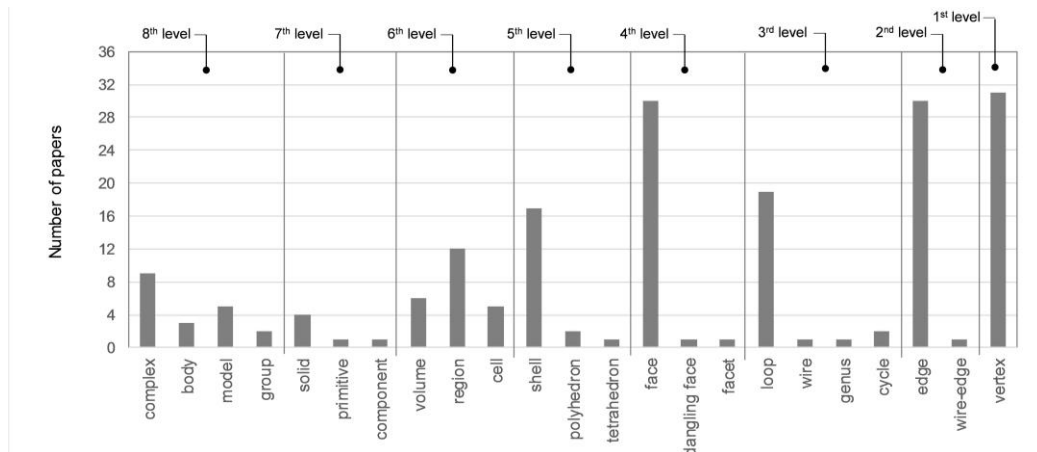[1] More information regarding the ambiguities can be found in [56].

Figure 4: Number of occurrences of entity names in non-manifold academic research (authors' own).

4.2 Non-Manifold Geometry Kernels

The main requirement for the geometric kernel to support conceptual design is to provide a non-manifold topology so that mixed-dimensional geometry can be allowed [28]. Twelve geometric modeling kernels that support non-manifold topology have been put to the test and they are assessed according to three characteristics, namely the topological entities hierarchy they support (Section 4.2.1); their license types, including whether they are commercial or open source (Section 4.2.2); and the offered topological operations (Section 5). A table including the information presented in this section in a concise format is attached in Appendix A (Tab. 6). The following geometric kernels are reviewed in this paper.

- ACIS, by Spatial Corporation
- SOLIDS++, by IntegrityWare
- Parasolid, by Siemens
- ARCHMIND, by Theo Athanasiadis
- BMesh, by various contributors
- Open CASCADE Technology (OCCT), by Open CASCADE SAS
- OpenVolumeMesh (OVM), by RWTH Aachen University
- CGAL, by various contributors
- LibIGL, by the Interactive Geometry Lab of ETH Zürich
- Rhino SDK, by McNeel and Associates
- ASM, by Autodesk
- SMLib, by Solid Modeling Solutions

4.2.1 Topological entities hierarchy

The topological elements used in each kernel vary and so does the hierarchy they use. All kernels support vertices and faces (CGAL uses the term 'facet' instead of 'face'), while all except LibIGL support edges. Compared to the review on the entities terminology from the academic perspective, the geometry kernels make distinct use of loop and wire, with the loop indicating a closed wire. This distinction creates an extra level in the entity hierarchy. Some kernels have the notion of the shell and it seems that the naming from the next level upwards varies, as shown in Tab. 1. For the 7th level, names such as volume, region, cell, lump and solid are being used with region being the most preferred one to signify the space inside a closed shell. In the 8th level, which refers to two closed shells linked by their faces, 'CompSolid' is used by OCCT and 'solid' by ASM and Rhino SDK. As for the 9th level, which includes the collection of any of the lower entities, the dominant terminology is 'body', used by ACIS, Parasolid and ASM. Other naming for this level includes 'compound', 'model'

and 'brep'. The variant number of entities used by each kernel suggests that some kernels provide a richer environment to work in, while others, such as LibIGL, ARCHMIND or BMesh are simply mesh representation libraries.

Level	Entities	Kernels												No of instances
		OCCT	OVM	CGAL	LibIGL	ARCHMIND	BMesh	ACIS	SOLIDS++	Parasolid	ASM	Rhino SDK	SMLib	
9th	body							●		●	●			3
	compound	●												1
	model												●	1
	brep								●					1
8th	CompSolid	●												1
	solid										●	●		2
7th	volume			●										1
	region									●		●	●	3
	cell		●					●			●			3
	lump							●						1
	solid	●												1
6th	shell	●		●				●	●	●	●		●	7
5th	face	●	●	●	●	●	●	●	●	●	●	●	●	12
4th	loop			●			●	●	●	●	●	●	●	8
3rd	wire	●						●						2
2nd	edge	●	●	●		●	●	●	●	●	●	●	●	11
1st	vertex	●	●	●	●	●	●	●	●	●	●	●	●	12
No of entities		8	4	6	2	3	4	9	6	7	8	6	7	

Table 1: Use of entity terminology in non-manifold geometry kernels (authors' own).

4.2.2 Licensing

Whether a kernel is open-source or proprietary has a direct association to the licensing terms under which it is distributed. There is no one license that presents no limitations; it is useful, however, to be aware of the strengths and limitations of each when using open-source tools.

OCCT, OpenVolumeMesh, CGAL, LibIGL, ARCHMIND and BMesh are open-source kernels, which means that they provide a freely available source code. This can be considered an advantage, as open-source tools facilitate interoperability, having the capability for direct integration in various software products. The source code of the open source kernels can be distributed to anyone under various licensing terms and conditions preserving the provenance and openness of the engine. These terms address the freedom to run the engine for any purpose, the freedom to study how it works and to adapt it to one's needs, the freedom to redistribute copies of it and the freedom to modify it to add further capabilities and distribute any improvements to the public [48].

On the other hand, SOLIDS++, Parasolid, ACIS, Rhino SDK, ASM and SMLib are proprietary ones, meaning that under commercial licenses they have closed source codes that are not freely accessible. It should be noted, however, that ACIS offers a University program, under which ACIS can be made accessible for one year with the possibility to renew the contract. More information on the advantages and disadvantages of open-source and proprietary tools can be found on the White

Paper provided by Optimus Information [44]. The above information is also included in Appendix A (Tab. 6).

5 PROPOSAL OF A STANDARDIZED ENTITIES' TERMINOLOGY

As briefly described in Section 1.2, NMT (as well as its manifold counterpart) has benefited various disciplines in creating spatial building models, material building models, structural models, mechanical models or building services models. However, community members across different areas may refer to the same NMT concepts using different terminologies, and may use them for different purposes. For example, in a spatial building model a location could actually be the alias for a vertex, the centre curve the alias for an edge and a path the alias for a wire. The cell could be used as a space or as a thermal zone in a spatial building model, a wall or a floor in a BIM model, as well as a slab or a beam in a structural model.

In light of a more standardized topological framework with regard to the naming and hierarchy of topological elements, a class hierarchy is proposed in Fig. 5. The terminology is proposed to provide a common concept for the diverse discipline-specific terminologies, including the ones for conceptual architectural design, structural design, energy analysis, spatial reasoning and digital fabrication. The terminology is proposed according to the following principles: to reduce ambiguity, to increase distinctiveness, to use simple words, to use words that do not imply a specific discipline, and to use independent descriptors between topological and geometric entities. The entities up to the level of a shell use the currently preferred terminology in academia and in commercial applications. From then on, a cell implies a region of a bounded space that can be either filled (solid) or void. This entity resembles real-life cells such as a biological cell and a prison cell. A CellComplex indicates a series of connected cells and resembles a building complex. A cluster can contain heterogeneous elements, and is a familiar concept in a number of areas including biology (e.g. a cluster of cells), architecture (a cluster of buildings), and set theory (an unordered cluster of objects).
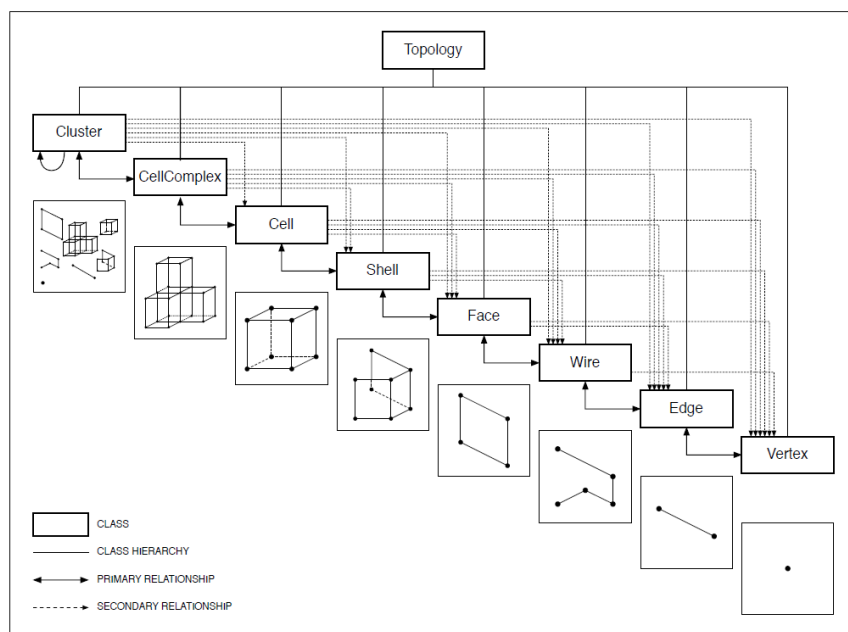

Figure 5: Topological elements class hierarchy with examples (authors' own).

6   A TESTING FRAMEWORK FOR NON-MANIFOLD TOPOLOGY: A STUDY CASE WITH OCCT

As a component of the geometry kernels' review written in Section 4.2, this section presents a proposed testing framework to assess the kernels' support for non-manifold structures. The tests particularly aim to identify the provided structural representations of a non-manifold structure and operations involving these structures. Structural representations are examined using construction tests (Section 6.1), in which a structure is created from simpler primitives; and exploration tests (Section 6.2), in which traversals are done between sub-entities of a shape. Operation tests are focused on merge (Section 6.3) and slice (Section 6.4) operations, which are two non-regular Boolean operations supported by OCCT. The discussions cover the correctness of the resulting shapes and the operations' performances. Performance tests were conducted using a machine with Intel Core i7-7600U@2.80 GHz processor, 16 GB of RAM, and Windows 10 Pro. OCCT 7.2.0 64-bit was used and the kernel was built using Visual Studio 2017. Experimentations with other non-manifold kernels regarding kernel capabilities and applications are reported in various studies [2], [12], [19], [31], [33], [54].

6.1   Construction Tests

OCCT provides methods to construct various predefined shapes. A box can thus be created as a cell by using a built-in class and passing, for example, either two extreme corners of the box or one corner and the dimension of the box. Alternatively, a similar box structure can be manually built in a bottom-up manner using the following steps.

1. Create 8 corners of the box
2. Connect the corners into 16 edges
3. Connect every 4 edges on each side on the box into a wire
4. Create the box faces based on the wires
5. Create a shell from the encompassing faces
6. Create a solid from the shell

Fig. 6 shows a performance comparison between the two methods. It can be seen that the built-in class significantly outperforms the manual method. This was expected, as the built-in method pre-allocates memory in the most optimal way, while the manual method reallocates memory along the testing process. It is yet a useful test to explore different ways of creating a topological shape. It also shows the difference in time and thus gives an idea of the time saved when using built-in methods, which can be extrapolated according to the size of the model. The first one requires less than 5 ms even to construct up to 900 cubes, whereas the manual method grows linearly up to around 10 seconds to generate the same number of cubes.
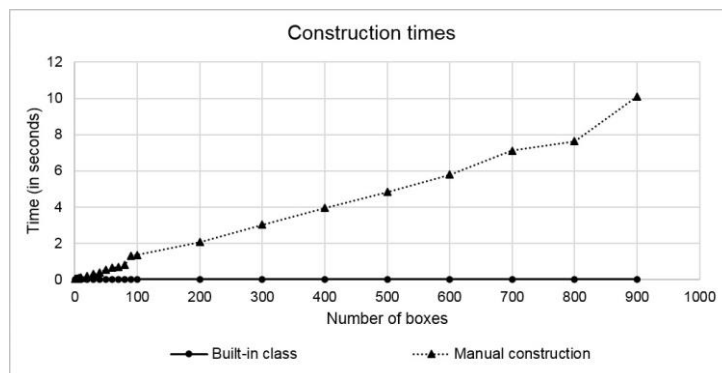

Figure 6: The time complexities of the cubes construction processes using OCCT's built-in class and a manual construction method.

6.2   Exploration Tests

Topological explorations in this paper are divided into three categories:

1. Upward explorations, involving traversals from a higher level to a lower level sub-entity.
2. Downward explorations, involving traversals from a lower level to a higher level sub-entity.
3. Sideways explorations/adjacency queries, involving traversals between sub-entities at the same hierarchical level.

OCCT allows both downward and upward explorations, respectively giving access to a sub-entity's parents and children at any level. Downward explorations can start from any type of sub-entities, and an iterator of the children of the parent sub-entity will be provided. Upward exploration is available; however, it requires a higher level entity which provides context to the navigation, and contains both the input sub-entity and its desired parents. In fact, the implementation of upward navigation performs downward navigation from the higher-level entity to find the sub-entities that have the requested types and are parents to the input sub-entity, rather than relying on an explicit relationship from a child to its parents. Despite these two features, OCCT does not provide a direct means to directly perform sideways explorations. For example, to iterate through a series of connected edges, the parent wire must firstly be examined before the constituent edges can be checked for adjacency. A mechanism that allows traversal between connected edges would therefore be convenient to the library users.

6.3   Merge Operation

The merge operation was examined by uniting two cubes in 11 configurations, according to the various ways they could be linked. These configurations are shown in Appendix B (Tabs. 7 and 8). It was found following this testing that OCCT satisfactorily returned the correct number of sub-entities. In addition, it could be verified that the cube and the tetrahedron in Test 11 shared exactly one vertex, which was the tetrahedron's vertex lying on the cube's face. Shifting this vertex slightly outside the cube will result in no intersection, whereas shifting it inside will slice the tetrahedron and create a new cell, which is the portion of the original tetrahedron that is inside the cube, akin to Tests 7 and 8.

The time complexity of this functionality was tested by performing the merge operation on three arrangements (1D, 2D, and 3D) of overlapping cubes as visualized in Fig. 7. These arrangements were designed such that the same number of input cubes will result in different numbers of sub-entities. It was found, as depicted in Fig. 8, that while all processing times similarly rose polynomially, the most significant rise occurred with the 3D arrangements, which created the most complex structures, followed by the 2D and finally the 1D arrangements.

1D arrangement
Up to 1000 cubes

2D arrangement
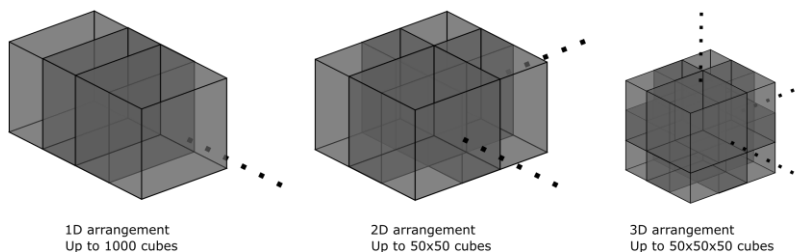Up to 50x50 cubes

3D arrangement
Up to 50x50x50 cubes

Figure 7: Three types of cubes arrangements that were used to assess the merge operation's performance.

6.4   Non-Regular Slice Operation

The non-regular slice operation available in OCCT was assessed by successively slicing a box with parallel finite planes, each regularly arranged within an interval of 1 unit from the other. Fig. 9

shows three arrangements of planes that were designed, and in each arrangement the planes were perpendicular to 1, 2, and 3 axes of the coordinate system. After each iteration, the new shape's topology was checked, and the numbers of sub-entities should abide the equations written in Tabs. 2–4. The derivation of the equations presented in these tables are given in Appendix C (Tabs. 9–11). It can be seen from the same tables that this operation produced the correct numbers of sub-entities.
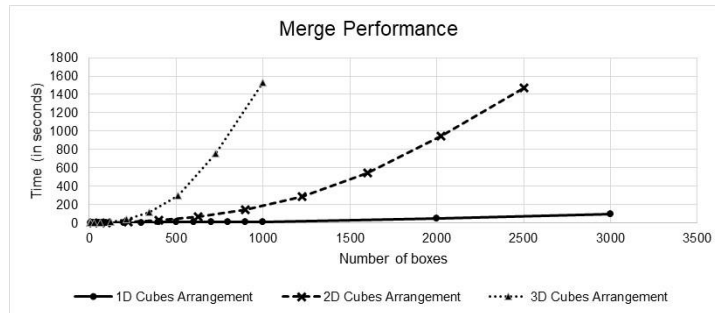

Figure 8: The merge operation's performance given different numbers of cubes in three arrangements.
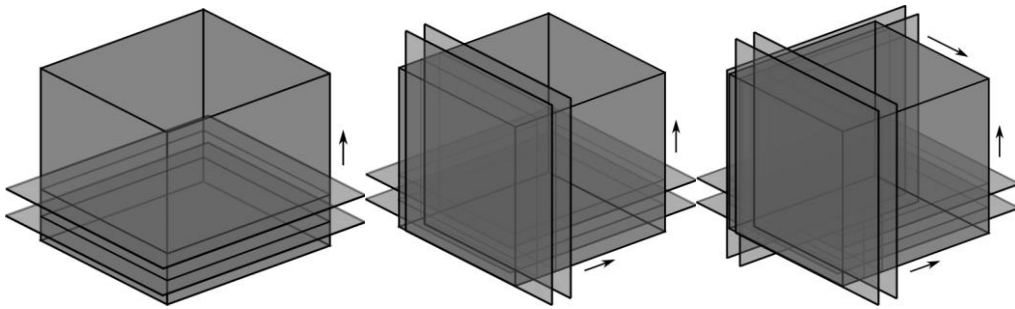

Figure 9: Illustrations of the slice operation evaluation. More slicing planes were inserted to the cube along the directions of the arrows as the evaluation progressed.

Sub-entity	Topological equation	Actual numbers of sub-entities on after n slices										
		0	1	2	3	4	5	6	7	8	9	10
Vertex	$8+4n$	8	12	16	20	24	28	32	36	40	44	48
Edge	$12+8n$	12	20	28	36	44	52	60	68	76	84	92
Face	$6+5n$	6	11	16	21	26	31	36	41	46	51	56
Wire	$6+5n$	6	11	16	21	26	31	36	41	46	51	56
Shell	$1+n$	1	2	3	4	5	6	7	8	9	10	11
Cell	$1+n$	1	2	3	4	5	6	7	8	9	10	11

Table 2: The expected and the actual topologies of the resulting shapes after the cube is sliced with planes perpendicular to one axis.

Sub-entity	Topological equation	Actual numbers of sub-entities on after n slices on each direction										
		0	1	2	3	4	5	6	7	8	9	10
Vertex	$8+8n+2n^2$	8	18	32	50	72	98	128	162	200	242	288
Edge	$12+16n+5n^2$	12	33	64	105	156	217	288	369	460	561	672
Face	$6+10n+4n^2$	6	20	42	72	110	156	210	272	342	420	506
Wire	$6+10n+4n^2$	6	20	42	72	110	156	210	272	342	420	506
Shell	$(1+n)^2$	1	4	9	16	25	36	49	64	81	100	121
Cell	$(1+n)^2$	1	4	9	16	25	36	49	64	81	100	121

Table 3: The expected and the actual topologies of the resulting shapes after the cube is sliced with planes perpendicular to two axes. The number of slicing planes is thus equal to n multiplied by 2.

Sub-entity	Topological equation	Actual numbers of sub-entities on after n slices on each direction										
		0	1	2	3	4	5	6	7	8	9	10
Vertex	$8+12n+6n^2+n^3$	8	27	64	125	216	343	512	729	1000	1331	1728
Edge	$12+24n+15n^2+3n^3$	12	54	144	300	540	882	1344	1944	2700	3630	4752
Face	$6+15n+12n^2+3n^3$	6	36	108	240	450	756	1176	1728	2430	3300	4356
Wire	$6+15n+12n^2+3n^3$	6	36	108	240	450	756	1176	1728	2430	3300	4356
Shell	$(1+n)^3$	1	8	27	64	125	216	343	512	729	1000	1331
Cell	$(1+n)^3$	1	8	27	64	125	216	343	512	729	1000	1331

Table 4: The expected and the actual topologies of the resulting shapes after the cube is sliced with planes perpendicular to three axes. The number of slicing planes is thus equal to n multiplied by 3.

The same planes arrangements were used to assess the operation's performance. The recorded times are shown in Fig. 10 with respect to the corresponding number of planes and the arrangement types. Similar to the merge's performance, the processing times for the non-regular slice also rose polynomially, with operations involving 3D planes arrangements rising more steeply than the other arrangements due to the resulting shapes' complexity.
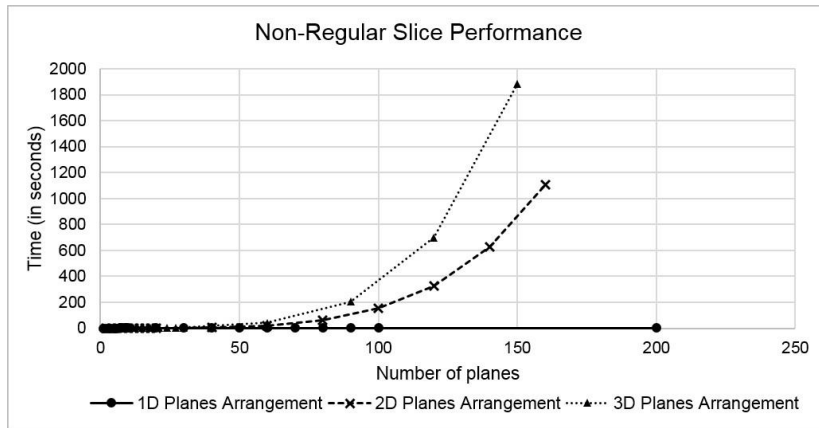
Figure 10: The slice operation's performance given different numbers of planes in three arrangements.

7 ACKNOWLEDGEMENTS

Aikaterini Chatzivasileiadi, http://orcid.org/0000-0001-5413-466X
Nicholas Mario Wardhana, http://orcid.org/0000-0002-7064-2063
Wassim Jabi, http://orcid.org/0000-0002-2594-9568
Simon Lannon, http://orcid.org/0000-0003-4677-7184

REFERENCES

[1]   Aish, R.; Pratap, A.: Spatial information modeling of buildings using non-manifold topology with ASM and DesignScript, in Advances in Architectural Geometry 2012, Vienna: Springer Vienna, 2013, 25–36. https://doi.org/10.1007/978-3-7091-1251-9_1.

[2]   Alleaume, A.: Automatic non-manifold topology recovery and geometry noise removal, in Proceedings of the 18th International Meshing Roundtable, Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, 267–279. https://doi.org/10.1007/978-3-642-04319-2_16.

[3]   de Arruda, M. C.; Martha, L. F. C. R.; Miranda, A. C. D. O.; Lira, W. W. M.: Boolean operations on non-manifold and B-rep solids for mesh generation, in XXVII Iberian Latin American Congress on Computational Methods in Engineering, 2006.

[4]   Autodesk MAYA LT: Two-manifold and non-manifold polygonal geometry, 2016. [Online]. Available:https://knowledge.autodesk.com/support/maya-lt/learnexplore/caas/CloudHelp/cloudhelp/2016/ENU/MayaLT/files/GUID-8E97CEF7-1CFE-4838-B4B7-59F526E21AB2-htm.html. [Accessed: 21-Feb-2017].

[5]   Basanow, J.; Neis, P.; Neubauer, S.; Schilling, A.; Zipf, A.: Towards 3D spatial data infrastructures (3D-SDI) based on open standards - experiences, results and future issues, Advances in 3D Geoinformation Systems, 2008, 65–86. https://doi.org/10.1007/978-3-540-72135-2.

[6]   Boguslawski, P.; Gold, C.: The dual half-edge: a topological primal/dual data structure and construction operators for modelling and manipulating cell complexes, ISPRS International Journal of Geo-Information, 5(2), 2016, 19. https://doi.org/10.3390/ijgi5020019.

[7]   Bronson, J.; Levine, J. A.; Whitaker, R.: Lattice cleaving: a multimaterial tetrahedral meshing algorithm with guarantees, IEEE Transactions on Visualization and Computer Graphics, 2014. https://doi.org/10.1109/TVCG.2013.115.

[8]   Cavalcanti, P. R.; Carvalho, C. P. P.; Martha, L. F.: Non-manifold modelling: an approach based on spatial subdivision, Computer-Aided Design, 29(3), 1997, 209–220. https://doi.org/10.1016/S0010-4485(96)00066-8.

[9]   Cavalcanti, P. R.; Carvalho, C. P. P.; Martha, L. F.: Non-manifold modelling: an approach based on spatial subdivision, Computer-Aided Design, 29(3), 1997, 209–220. https://doi.org/10.1016/S0010-4485(96)00066-8.

[10]   Chatzivasileiadi, A.; Lannon, S.; Jabi, W.; Wardhana, N. .; Aish, R.: Addressing pathways to energy modelling through non-manifold topology, in SimAUD Conference 2018, 2018.

[11]   Crocker, G. A.; Reinke, W. F.: An editable nonmanifold boundary representation, IEEE Computer Graphics and Applications, 11(2), 1991, 39–51. https://doi.org/10.1109/38.75589.

[12]   Damiand, G.; Teillaud, M.: A Generic Implementation of dD Combinatorial Maps in CGAL, Procedia Engineering, 82, January 2014, 46–58. https://doi.org/10.1016/J.PROENG.2014.10.372.

[13]   Dyedov, V.; Ray, N.; Einstein, D.; Jiao, X.; Tautges, T. J.: AHF: array-based half-facet data structure for mixed-dimensional and non-manifold meshes, Engineering with Computers, 31, 2015, 389–404. https://doi.org/10.1007/s00366-014-0378-6.

[14]   De Floriani, L.; Hui, A.: A scalable data structure for three-dimensional non-manifold objects, in Eurographics Symposium on Geometry Processing, 2003, 72–82. https://doi.org/10.2312/SGP/SGP03/072-082.

[15]   Gueorguieva, S.; Marcheix, D.: Non-Manifold Boundary Representation for Solid Modelling, in Proceedings of the International Computer Symposium, 1994.

[16]   Gursoz, E. L.; Choi, Y.; Prinz, F. B.: Non-regularized Boolean set operations on non- manifold b-rep objects, Pittsburgh, PA, 1990.

[17]   Gursoz, E. L.; Choi, Y.; Prinz, F. B.: Vertex-based representation of non-manifold boundaries, in Geometric Modeling for Product Engineering, 1990, 107–139.

[18]   Hachenberger, P.; Kettner, L.; Mehlhorn, K.: Boolean operations on 3D selective Nef complexes: Data structure, algorithms, optimized implementation and experiments, Computational Geometry, 38(1–2), 2007, 64–99. https://doi.org/10.1016/j.comgeo.2006.11.009.

[19]   Hert, S.; Hoffmann, M.; Kettner, L.; Pion, S.; Seel, M.: An adaptable and extensible geometry kernel, Computational Geometry, 38(1–2), September 2007, 16–36. https://doi.org/10.1016/J.COMGEO.2006.11.004.

[20]   Higashi, M.; Yatomi, H.; Mizutani, Y.; Murabata, S.: Unified geometric modeling by non-manifold shell operation, in Second Symposium on Solid Modeling and Applications, 1993, 75–84. https://doi.org/10.1145/164360.164390.

[21]   Hui, A.; De Floriani, L.: A two-Level topological decomposition for non-manifold simplicial shapes, in SPM '07: Proceedings of the 2007 ACM Symposium on Solid and Physical Modeling, 2007, 335–360. https://doi.org/10.1145/1236246.1236297.

[22]   Jabi, W.: The potential of non-manifold topology in the early design stages, in Computational Ecologies: Design in the Anthropocene. Proceedings of the 35th Annual Conference of the Association for Computer Aided Design in Architecture, L. Combs and C. Perry, Eds. University of Cincinnati, 2015, 381–393.

[23]   Jabi, W.: Linking design and simulation using non-manifold topology, Architectural Science Review, 59(4), 2016, 323–334. https://doi.org/10.1080/00038628.2015.1117959.

[24]   Jabi, W.: Parametric spatial models for energy analysis in the early design stages, in Proceedings of the Symposium on Simulation for Architecture & Urban Design, 2014, 17–24. .

[25]   Jabi, W.; Soe, S.; Theobald, P.; Aish, R.; Lannon, S.: Enhancing parametric design through non-manifold topology, Design Studies, 52(C), 2017, 96–114. https://doi.org/10.1016/j.destud.2017.04.003.

[26]   Karasick, M.: On the representation and manipulation of rigid solids, Dept. Comput. Sci., McGill Univ., Montreal, PQ, 1989.

[27]   Karim, H.; Abdul-Rahman, A.; Boguslawski, P.; Meijers, M.; Van Oosterom, P.: The potential of the 3D Dual Half-Edge (DHE) data structure for integrated 2D-space and scale modelling: A

Review, in Advances in 3D Geoinformation Systems, Lecture Notes in Geoinformation and Cartography, A. Abdul-Rahman, Ed. Springer, 2017, 477–493. https://doi.org/10.1007/978-3-540-72135-2.

[28] Kiumarse Zamanian, M.; Fenves, S. J.; Levent Gursoz, E.: Representing spatial abstractions of constructed facilities, Building and Environment, 27(2), 1991, 221–230. https://doi.org/10.1016/0360-1323(92)90024-J.

[29] Kiumarse Zamanian, M.; Fenves, S. J.; Levent Gursoz, E.: Representing spatial abstractions of constructed facilities, Building and Environment, 27(2), 1991, 221–230. https://doi.org/10.1016/0360-1323(92)90024-J.

[30] Kjellberg, T. .: Integrerad Datorstöd for Mänsklig Problemlösning och Mänsklig Kom-munikation inom Verkstadsteknisk Produktion begränsat till Produkt-utveckling, Produktions-beredning, Kon-struktion och Till-verknings-beredning. En systemansats baserad paa Produkt-model, KTH, Stockholm, Sweden, 1982.

[31] Kremer, M.; Bommes, D.; Kobbelt, L.: OpenVolumeMesh - A versatile index-based data structure for 3D polytopal complexes, in 21st International Meshing Roundtable, 2012. https://doi.org/10.1007/978-3-642-33573-0 31.

[32] Landier, S.: Boolean operations on arbitrary polyhedral meshes, Procedia Engineering, 124, 2015, 200–212. https://doi.org/10.1016/j.proeng.2015.10.133.

[33] Lee, S. H.: Feature-based non-manifold modeling system to integrate design and analysis of injection molding products, Journal of Mechanical Science and Technology, 23(5), May 2009, 1331–1341. https://doi.org/10.1007/s12206-009-0407-3.

[34] Lee, S. H.; Lee, K.: Partial Entity Structure: A compact non-manifold boundary representation based on partial topological entities, in Proceedings of the Sixth ACM Symposium on Solid Modeling and Applications, 2001, 159–170. https://doi.org/10.1145/376957.376976.

[35] Lee, S. U.; Roh, M. Il; Cha, J. H.; Lee, K. Y.: Ship compartment modeling based on a non-manifold polyhedron modeling kernel, Advances in Engineering Software, 40(5), 2009, 378–388. https://doi.org/10.1016/j.advengsoft.2007.12.001.

[36] Lipson, H.; Shpitalni, M.: On the Topology of Sheet Metal Parts, Journal of Mechanical Design, 120(1), 1998, 10–16. https://doi.org/10.1115/1.2826661.

[37] Luo, Y.; Lukacs, G.: A boundary representation for form features and non-manifold solid objects, in 1st ACM/SIGGRAPH Symposium on Solid Modeling Foundations and CAD/CAM Applications, 1991.

[38] Masuda, H.: Topological operators and Boolean operations for complex-based nonmanifold geometric models, Computer-Aided Design, 25(2), February 1993, 119–129. https://doi.org/10.1016/0010-4485(93)90097-8.

[39] Masuda, H.; Shimada, K.; Numao, M.; Kawabe, S.: A mathematical theory and applications of non-manifold geometric modeling, in International Symposium on Advanced Geometric Modelling for Engineering Applications, 1989, 89–103.

[40] Mikchevitch, A.; Pernot, J.-P.: Methodology for automatic recovering of 3D partitions from unstitched faces of non-manifold CAD models, Engineering with Computers, 31(1), January 2015, 73–84. https://doi.org/10.1007/s00366-013-0325-y.

[41] Muuss, M. J.; Butler, L. A.: Combinatorial Solid Geometry, Boundary Representations, and Non-Manifold Geometry, in State of the Art in Computer Graphics: Visualization and Modeling, D. F. Rogers and R. A. Earnshaw, Eds. New York: Springer-Verlag, 1991, 185–223. .

[42] Nguyen, T.: Simplifying the non-manifold topology of multi-partitioning surface networks, Washington University in St. Louis, Missouri, 2011.

[43] Nolan, D. C.; Tierney, C. M.; Armstrong, C. G.; Robinson, T. T.: Defining simulation intent, CAD Computer Aided Design, 59, 2015, 50–63. https://doi.org/10.1016/j.cad.2014.08.030.

[44] Optimus Information: Open-source vs. proprietary software: Pros and Cons, Vancouver, Canada, 2015.

[45] Pereira, A. M. B.; De Arruda, M. .; Miranda, A. C.; Lira, W. W. M.; Martha, L. F.: Boolean operations on multi-region solids for mesh generation, Engineering with Computers, 28, 2012, 225–239. https://doi.org/10.1007/s00366-011-0228-8.

[46]   Preparata, F. P.; Shamos, M. I.: Computational Geometry: An Introduction, Corrected. New York: Springer Verlag, 1985.

[47]   Requicha, A. A. G.; Voelcker, H. B.: Boolean operations in solid modeling: boundary evaluation and merging algorithms, Proc. IEEE, 73(1), 1985, 30–44.

[48]   Rosen, L.: Open source licensing: software freedom and intellectual property law. Prentice Hall, 2004.

[49]   Rossignac, J. R.; Requicha, A. G.: Constructive non-regularized geometry, 1991.

[50]   Rossignac, J.; O'Connor, M.: SGC: A Dimension-independent model for pointsets with internal structures and incomplete boundaries, Geometric Modeling for Product Engineering, Proceedings of the IFIP Workshop on CAD/CAM, 1989, 145–180.

[51]   Saxena, M.; Finnigan, P.; Graichen, C. M.; Hathaway, A. F.; Parthasarathy, V. N.: Octree-based automatic mesh generation for non-manifold domains, Engineering with Computers, 11(1), 1995. https://doi.org/10.1007/BF01230440.

[52]   Shimada, K.; Gossard, D.: Bubble Mesh: Automated triangular meshing of non-manifold geometry by sphere packing, in Proceedings of the 3rd Symposium on Solid Modeling and Applications, 1995, 409–419.

[53]   Spatial corporation: Model topology, 2007. [Online]. Available: http://www-isl.ece.arizona.edu/ACIS-docs/PDF/%0AFCG/06TOPO.PDF. 1.

[54]   Sriram, R. D.; Wong, A.; He, L.-X.: Gnomes: an object-oriented nonmanifold geometric engine, Computer-Aided Design, 27(11), November 1995, 853–868. https://doi.org/10.1016/0010-4485(95)00022-4.

[55]   Stroud, I.; Nagy, H.: Solid Modelling and CAD Systems. London: Springer-Verlag London Limited, 2011. https://doi.org/10.1007/978-0-85729-259-9_6.

[56]   Stroud, I.: Boundary representation modelling techniques. London: Springer Verlag, 2006. https://doi.org/10.1007/978-1-84628-616-2.

[57]   Vivodtzev, F.; Bonneau, G.-P.; Le Texier, P.: Topology-preserving simplification of 2D nonmanifold meshes with embedded structures, The Visual Computer, 21(8–10), September 2005, 679–688. https://doi.org/10.1007/s00371-005-0334-y.

[58]   Weiler, K.: Edge-based data structures for solid modeling in curved-surface environments, IEEE Computer Graphics and Applications, 5(1), 1985, 21–40.

[59]   Weiler, K.: Topological structures for geometric modeling, Graduate Faculty of Rensselaer Polytechnic Institute, 1986.

[60]   Weiler, K.: The radial edge structure: A topological representation for non-manifold geometric boundary modeling, in Geometric Modeling for CAD Applications, M. J. Wozny, H. W. McLaughlin and J. L. Encarnacão, Eds., North-Holland, 1988, 3–36.

[61]   Willems, P.: A meta-topology for product modeling, in CIB meeting, 1988, 214–222.

[62]   Yamaguchi, Y.; Kimura, F.: Non-manifold topology based on coupling entities, IEEE Computer Graphics and Applications, 15, 1995, 42–50. https://doi.org/10.1109/38.364963

[63]   Zeng, L.; Liu, Y. J.; Lee, S. H.; Yuen, M. M. F.: Q-Complex: Efficient non-manifold boundary representation with inclusion topology, Computer-Aided Design, 44(11), 2012, 1115–1126. https://doi.org/10.1016/j.cad.2012.06.002.

APPENDIX A   TOPOLOGICAL ENTITIES' HIERARCHY IN ACADEMIC RESEARCH AND IN THE INDUSTRY

Detailed information regarding the review of the topological entities' hierarchy in academic research and in the industry is shown in Tabs. 5 and 6.

Researcher(s)	Data structure (if available)	Extended by	Adopted by	Topological entities hierarchy							Entity no	Further entities
Requicha and Voelcker [47]	-	-	-	body	-	-	-	face	-	edge vertex	4	-
Weiler [58]	Radial-edge structure	Gursoz, Choi, and Prinz [16], Lee and Lee [34] Luo and Lukacs [37]	Crocker and Reinke [11], Muuss and Butler [41], Cavalcanti, Carvalho, and Martha [9]	model	-	region	shell	face	loop	edge vertex	7	-
Masuda et al. [39]	-	-	Masuda [38], Jabi [23]	complex	-	volume	shell	face	loop	edge vertex	7	-
Rossignac and O'Connor [50]	Simplicial Geometric Complexes	-	Rossignac and Requicha [49]	complex	-	-	-	face	-	edge vertex	4	-
Karasick [26]	Star-edge boundary representation	-	-	-	-	-	shell	face	cycle	edge vertex	5	-
Gursoz, Choi, and Prinz [16], [17]	Tri-cyclic cusp structure (Vertex-based data structure)	-	-	-	-	region	shell	face	loop	edge vertex	6	zones, disks
Kiumarse Zamanian, Fenves, and Levent Gursoz [29]	-	-	-	-	solid	-	shell	face	cycle	edge vertex	6	-
Higashi et al. [20]	Cycle structure	-	-	-	-	region	shell	face	loop	edge vertex	6	-
Gueorguieva and Marcheix [15]	-	-	-	complex / object	primitive	volume	shell	face	loop	edge vertex	9	scene
Yamaguchi and Kimura [62]	Coupling entities data structure	-	-	-	-	region	shell	face	loop	edge vertex	6	-
Cavalcanti, Carvalho, and Martha [9]	Complete Geometric complexes	-	-	-	-	region	shell	face	loop	edge vertex	6	-
Lipson and Shpitalni [36]	-	-	-	-	compo-nent	volume	-	face	genus	edge vertex	6	edge, bend, weld
Lee and Lee [34]	Partial edge structure	-	-	model	-	region	shell	face	loop	edge vertex	7	-
de Floriani and Hui [14]	Non-manifold indexed data structure with adjacencies	-	-	-	-	-	tetrahedron	dangling face	-	wire vertex-edge	4	-
de Arruda et al. [3]	-	-	Pereira et al. [45]	group	solid	region	shell	face	wire	edge vertex	8	-
Hui and de Floriani[21]	Incidence Simplicial Data Structure	-	-	complex	-	-	-	face	-	edge vertex	4	-
Hachenberger, Kettner, and Mehlhorn [18]	SNC(Nef complex) structure	-	-	-	-	volume	shell	face/facet	loop	edge vertex	6	-

Aish and Pratap [1]	-	-	-	body	solid	cell	shell	face	loop	edge	vertex	7	-
Zeng et al. [63]	Q-complex data structure	-	-	body	-	region	shell	face	loop	edge	vertex	7	-
Landier [32]	-	-		-	-	cell	polyhedron	face	-	edge	vertex	4	-
Boguslawski and Gold [6]	Dual half-edge data structure	-	Basanow et al. [27]	complex	-	cell	shell	face	loop	edge	vertex	7	-
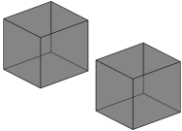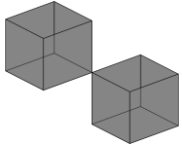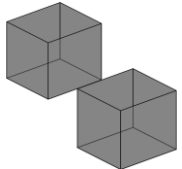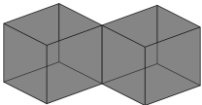
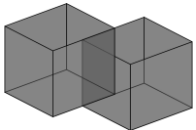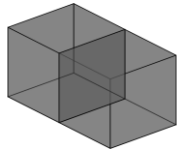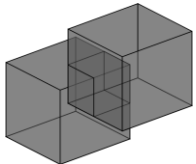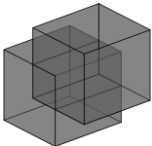**Table 5**: Topological entities' hierarchy found in academic research for non-manifold modeling.

	Developer	Open-source	License	Topological entities hierarchy									Ent no	Extra entities
				9th level	8th level	7th level	6th level	5th level	4th level	3rd level	2nd level	1st level	9	
OCCT	Open Cascade SAS	YES	LGPL	compound	CompSolid	solid	shell	face	-	wire	edge	vertex	8	-
OpenVolumeMesh	Rwth Aachen University	YES	LGPL	-	-	cell	-	face (half-face)	-	-	edge (half-edge)	vertex	4	half-face, half-edge
CGAL	various	YES	Dual license: open source (LGPL, GPL) /commercial	-	-	volume	shell	facet (half-facet)	loop	-	Edge (half-edge)	vertex	6	half-facet, half-edge
LibIGL	ETH Zurich	YES	MPL2 (Mozilla)	-	-	-	-	face	-	-	-	vertex	2	-
ARCHMIND	Theo Athanasiadis	YES	Zlib	-	-	-	-	face	-	-	edge	vertex	3	-
BMesh (Blender)	various	YES	GPL	-	-	-	-	face	loop	-	edge	vertex	4	-
ACIS	Spatial Corporation	NO	commercial	body	-	lump	shell, (subshell)	face	loop	wire	edge (coedge)	vertex	8	subshell, coedge
SOLIDS++	IntegrityWare	NO	commercial	brep object	-	region	shell	face	loop	-	edge	vertex	7	-
Parasolid	Siemens	NO	commercial	body	-	region	shell	face	loop	-	fin, edge	vertex	7	Fin
ASM	Autodesk	NO	commercial	body	-	cell	shell	face	loop	-	edge	vertex	7	-
Rhino SDK	McNeel and Associates	NO	commercial	-	solid	region	-	face	loop	-	edge	vertex	6	-
SMLib	Solid Modeling Solutions	NO	commercial	model	-	region	shell	face (faceuse)	loop (loopuse)	-	edge (edgeuse)	vertex (vertexuse)	7	faceuse loopuse edgeuse vertexuse

Table 6: Characteristics of 3D geometry kernels.

APPENDIX B   TWO-BOX CONFIGURATIONS FOR THE MERGE OPERATION

The eleven two-box configurations used in testing the merge operation are shown in Table 7.

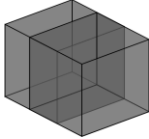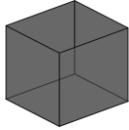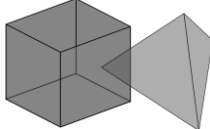| No.	Configuration	Description	Expected Topology		Actual Topology	
1		Two disjoint boxes	V: 16 E: 24 W: 12	F: 12 S: 2 C: 2	V: 16 E: 24 W: 12	F: 12 S: 2 C: 2
2		Two boxes sharing a vertex	V: 15 E: 24 W: 12	F: 12 S: 2 C: 2	V: 15 E: 24 W: 12	F: 12 S: 2 C: 2
3		Two boxes with partially overlapping edges	V: 16 E: 25 W: 12	F: 12 S: 2 C: 2	V: 16 E: 25 W: 12	F: 12 S: 2 C: 2
4		Two boxes sharing a whole edge	V: 14 E: 23 W: 12	F: 12 S: 2 C: 2	V:14 E: 23 W: 12	F: 12 S: 2 C: 2
5		Two boxes with partially overlapping faces	V: 16 E: 26 W: 13	**F: 14** S: 2 C: 2	V:16 E: 26 W: 13	**F: 13** S: 2 C: 2
6		Two boxes sharing a whole face	V: 12 E: 20 **W: 11**	F: 11 S: 2 C: 2	V: 12 E: 20 **W: 12**	F: 11 S: 2 C: 2
7		Two boxes with partially overlapping edges, faces, and cells	V: 22 E: 36 W: 18	F: 18 S: 3 C: 3	V: 22 E: 36 W: 18	F: 18 S: 3 C: 3
8		Two boxes sharing a whole edge, and with partially overlapping edges, faces, and cells	V: 20 E: 34 W: 18	F: 18 S: 3 C: 3	V: 20 E: 34 W: 18	F: 18 S: 3 C: 3

#		Description	Expected				Actual			
9		Two boxes sharing a whole edge and face, and with partially overlapping edges, faces, and cells	V:16 E: 28 W: 16	F: 16 S: 3 C: 3			V:16 E: 28 W: 16	F: 16 S: 3 C: 3		
10		Two co-located boxes	**V: 8** E: 12 W: 6	F: 6 S: 1 C: 1			**V:16** E: 12 W: 6	F: 6 S: 1 C: 1		
11		A tetrahedron with its apex touching a box's face	V: 12 E: 18 W: 6	F: 6 S: 2 C: 2			V: 12 E: 18 W: 6	F: 6 S: 2 C: 2		

Table 7: Eleven two-box configurations with their expected and actual topologies. Initial descriptions: V = Vertices, E = Edges, W = Wires, F = Faces, S = Shells, C = Cells. Mismatches are in bold.

APPENDIX C   DERIVING THE TOPOLOGICAL EQUATIONS FOR THE SLICE OPERATION

Slicing a cube with a series of planes in different arrangements yield new topologies as a result of intersecting and splitting the cube's sub-entities with the slicing planes. This section is dedicated to explain the number of those topological sub-entities created by one-axis, two axes, and three-axes slicing planes arrangements. The set of planes parallel to the Y- and Z-axes (in other words, perpendicular to the X axis) are called YZ-planes. The other two sets of planes are referred to as XY-planes and XZ-planes. The one-axis plane arrangement (Tab. 8) entirely uses YZ-planes. On the other hand, the two-axes arrangement (Tab. 9) employs XZ- and YZ-planes, whereas in the three-axes arrangement (Tab. 10) all sets of planes are in action.

In the aforementioned tables, it is assumed herein, without any loss of generality, that the cube is axis-aligned. The edges parallel to the X, Y, and Z axes will respectively be referred to as X-edges, Y-edges, and Z-edges. The faces parallel to the XY, YZ, and XZ planes are called XY-faces, YZ-faces, and XZ-faces (similarly, XY-wires, YZ-wires, and XZ-wires for the wires).

Sub-entity	Description	Quantity
Vertex	The cube's original vertices	8
	The intersections between the planes and the cube's X-edges	$4n$
	Total	$8+4n$
Edge	The cube's original Y- and Z-edges, left untouched by the planes	8
	The cube's original 4 X-edges each split into n+1 new X-edges	$4(n+1)$
	The planes' edges merged into the topology	$4n$
	Total	$12+8n$
Wire/Face	The cube's original YZ-wires/faces	2
	The cube's original 4 XY- and XZ-wires/faces each split into 1+n new wires and faces	$4(1+n)$
	The planes themselves, merged into the topology	n

	Total	6+5n
Shell/Cell	The original cube's shell/cell	1
	The new shells/cells as a result of the planes slicing the cube	n
	Total	1+n

Table 8: The numbers of sub-entities as a result of slicing a cube with a one-axis arrangement of n YZ-planes.

Sub-entity	Description	Quantity
Vertex	The cube's original vertices	8
	The intersections between the XZ-planes and the cube's Y-edges	4n
	The intersections between the YZ-planes and the cube's X-edges	4n
	The intersections between the XZ- and YZ-planes on the cube's XY-faces	$2n^2$
	Total	$8+8n+2n^2$
Edge	The cube's original Z-edges, left untouched by the planes	4
	The original 8 X- and Y-edges each split into (1+n) edges	8(1+n)
	The planes' edges on the cube's XY-faces, merged into the topology. The edges in turn split each other, and each of the plane's edge is split into n+1 edges. There are n initial edges from the XZ-planes and another n edges from the YZ-planes, so as many as 2n×(n+1) edges are created on each XY-face.	4n(1+n)
	The plane's edges on the cube's initial XZ- and YZ-faces, merged into the topology. A total of n edges are created on each face.	4n
	The internal intersections of the XZ- and YZ-planes inside the cube	n^2
	Total	$12+16n+5n^2$
Wire/Face	The cube's XZ- and YZ-faces each split into 1+n faces	4(1+n)
	The cube's 2 XY-faces each split into $(1+n)^2$ faces.	$2(1+n)^2$
	The internal intersections between the XZ- and YZ-wires/planes. There are a total of 2n wires/planes, each split into 1+n wires/faces.	2n(1+n)
	Total	$6+10n+4n^2$
Shell/Cell	The initial shell/cell split each n times into n+1 new shells/cells, for both the X- and Y-axes. This includes the shells/cells bounded by the internal intersections between the slicing XZ- and YZ-planes.	$(1+n)^2$
	Total	$(1+n)^2$

Table 9: The numbers of sub-entities as a result of slicing a cube with a two-axis arrangements of n XZ-planes, n YZ-planes, and n XZ-planes.

Sub-entity	Description	Quantity
Vertex	The cube's original vertices	8
	The intersections between the slicing planes and the cube's edges. There is always an intersection between every pair of edge and slicing plane.	$12n$
	The intersections between the plane's edges on the cube's faces, n 2 vertices inside each face (that is, excluding the intersections on the cube's edges which are written above)	$6n^2$
	The internal intersections between the planes inside the cube (i.e. excluding the vertices on the faces as written above). A vertex is created at every location where three perpendicular planes.	n^3
	Total	$8+12n+6n^2+n^3$
Edge	The original 12 edges each split into (1+n) edges	$12(1+n)$
	The planes' edges on the cube faces. There are 2n of these edges on each face, each split by other edges into n+1 edges.	$12n(1+n)$
	Edges created by the internal intersections of the slicing planes inside the cube. Every inter section edge between two perpendicular planes is further split into 1+n edges. There are a total of $3n^2$ of such intersection edges.	$3n^2(1+n)$
	Total	$12+24n+15n^2+3n^3$
Wire/Face	Each of the cube's wire/face is split into (1+n) 2 wires/faces.	$6(1+n)^2$
	Each of the slicing planes is divided by other intersecting planes into (1+n) 2 wires/faces. There are in total 3n planes.	$3n(1+n)^2$
	Total	$6+15n+12n^2+3n^3$
Shell/Cell	The initial shell/cell split each n times into n+1 new cells for all axes. This includes the shells/cells bounded by the internal intersections between the slicing planes.	$(1+n)^3$
	Total	$(1+n)^3$

Table 10: The numbers of sub-entities as a result of slicing a cube with a three-axis arrangement of n XZ-planes, n YZ-planes, and n XZ-planes.