Computer-Aided Design 152 (2022) 103370

Contents lists available at ScienceDirect

Computer-Aided Design

journal homepage: www.elsevier.com/locate/cad

Solid Modelling for Manufacturing: From Voelcker's Boundary Evaluation to Discrete Paradigms

C. Andújar, P. Brunet, A. Chica *, I. Navazo, À. Vinacua

Modelling, Visualization, Interaction and Virtual Reality Group, Computer Science Department, Universitat Politècnica de Catalunya, Barcelona, Spain

ARTICLE INFO

Article history:
Received 21 December 2021
Received in revised form 19 May 2022
Accepted 13 July 2022

Keywords:
Boundary Evaluation
Surface extraction
Topologically consistent isosurfaces
Discrete volume models
Topology optimization
Additive manufacturing

ABSTRACT

Herb Voelcker and his research team laid the foundations of Solid Modelling, on which Computer-Aided Design is based. He founded the ambitious Production Automation Project, that included Constructive Solid Geometry (CSG) as the basic 3D geometric representation. CSG trees were compact and robust, saving a memory space that was scarce in those times. But the main computational problem was Boundary Evaluation: the process of converting CSG trees to Boundary Representations (BReps) with explicit faces, edges and vertices for manufacturing and visualization purposes. This paper presents some glimpses of the history and evolution of some ideas that started with Herb Voelcker. We briefly describe the path from "localization and boundary evaluation" to "localization and printing", with many intermediate steps driven by hardware, software and new mathematical tools: voxel and volume representations, triangle meshes, and many others, observing also that in some applications, voxel models no longer require Boundary Evaluation. In this last case, we consider the current research challenges and discuss several avenues for further research.

1. Introduction

Herb Voelcker is the undisputed pioneer of Solid Modelling. At Rochester, in the 70s, he launched the ambitious *Production Automation Project*, surrounded by a group of collaborators that included Aristides Requicha, Robert Tilove, Jarek Rossignac and many others. Together, they laid the foundations of the mathematical modelling of physical rigid objects, defining regular-sets and regularized set operators. They also noted that models of real-world objects should contain sufficient information to compute any relevant geometric property of the modelled physical entity.

Voelcker highlighted the crucial role of 3D geometry in design and manufacturing activities, observing that "*curiously, these industries' primary means for specifying geometry (2D graphics) has not changed significantly for more than a century*" [1]. However, the development "*of new, computationally oriented schemes for handling mechanical geometry*" was a perfect opportunity that Herb and his team took advantage of. Representations of 3D mathematical objects could thus be devised, and for the first time they could be used to generate computational representations. The basic geometric representation in the Production Automation Project was the Constructive Solid Geometry (CSG), a compact symbolic structure that encoded a tree of regularized set operators (union, intersection, difference) defining the final mechanical part from a predefined set of basic 3D parameterized solid primitives [2].

CAD-related tasks can be broadly classified as *design* (i.e. specify and modify the model), *queries* (e.g. rendering, point classification), *optimization* (e.g. simplification), and *manufacturing automation*. As a representation scheme for solids, CSG proved to provide important advantages for many of these tasks. Some of these advantages include (a) mathematical accuracy for representing solids involving spherical, cylindrical and conical surface patches, (b) immediate guarantee of solid validity (water-tight objects) if the primitives are solids (thus avoiding the need of consistency checks), (c) fast and robust Point Membership Classification (PMC), and (d) easy editing by combining Boolean Operations, and through the parameters describing the primitive objects. Furthermore, CSG trees were extremely compact, saving a memory space that was scarce in those times. For these reasons, CSG was and still is an excellent choice as design representation for mechanical solids.

However, since CSG does not represent explicitly the boundary of the solid, it does not directly support some important tasks where such a boundary is needed, such as rasterization-based rendering, queries about area and volume properties, and Computer Numerical Control (CNC) i.e. automated control of machining tools such as lathes and mills. These tasks usually require Boundary Evaluation i.e. converting a CSG tree into a Boundary

* Corresponding author.
E-mail addresses: andujar@cs.upc.edu (C. Andújar), pere@cs.upc.edu (P. Brunet), achica@cs.upc.edu (A. Chica), isabel@cs.upc.edu (I. Navazo), alvar@cs.upc.edu (À. Vinacua).

https://doi.org/10.1016/j.cad.2022.103370

Representation (BRep) with explicit faces, edges and vertices. Boundary Evaluation soon became one of the main challenges in Solid Modelling. Voelcker and his team showed that Spatial Localization (the use of spatial locality as mentioned by Tilove and Requicha, [3]) should be at the kernel of the computational solutions to this problem, to avoid quadratic complexities and to have reasonable computational times. This resulted in the PADL-1 and PADL-2 modelling systems [2].

Hierarchical Space Subdivision representations and octrees appeared in parallel [4], as representation schemes that included intrinsic localization. Aristides Requicha and Herbert Voelcker mentioned [2] that octrees are one of the computational options to solve the mathematical problem of evaluating the boundary of regularized trees of operations on solids, while different types of octrees were proposed to efficiently compute regularized Boolean operations between solids [5]. In an extensive survey, Hanan Samet [6] presented different approaches to use octrees as a localization structure in the Boundary Evaluation problem. One example, discussed in [7], used Extended Octrees, a specific class of octrees that are able to encode polyhedral objects in which vertices have three incident faces, as a secondary representation to efficiently localize and compute BReps from a class of CSG trees.

This paper touches on the history and evolution of some ideas that started with Herb Voelcker. Over the last 50 years, we have witnessed important advances in computer hardware (computing power and memory storage) and fabrication technologies (such as the development of additive manufacturing, also known as 3D printing). In parallel, the requirements of CAD-related tasks have evolved in many ways. One of the major factors motivating this evolution is 3D digitization (e.g. photogrammetry, laser scanners), which has been rapidly evolving in terms of speed, accuracy and affordability. Nowadays multiple CAD applications rely on 3D scanned models. For example, in the dental industry, scanned data is used to design surgical guides, aligners, restorations and crowns, with accurate margins and contacts. Altogether, these changes in computing resources, in the nature of the objects being designed (e.g. custom objects such as a dental crown for a specific customer), and in manufacturing technologies, have largely impacted the choice of the most suitable representation for the different CAD-related tasks mentioned above.

In this work, we briefly describe how some of these changes have impacted solid representation schemes and their associated algorithms. Since this paper is a tribute to Herb Voelcker, we chose the Boundary Evaluation problem (one of Voelcker's major concerns, and the field that received many contributions from his team) as the main common thread of the paper. However, we consider Boundary Evaluation from a different point-of-view, in accordance to the contributions of our group, and considering new solid representation schemes that have become extremely popular in the last decades. Instead of dealing with Boundary Evaluation from CSG models, we review the surface extraction problem from discrete volumetric representations such as voxels and octrees.

In particular, we have structured the paper as follows. Section 2 briefly discusses exact representation schemes (e.g. CSG, B-Splines) vs. approximate ones (e.g. triangle meshes, voxels, octrees). The discussion is somewhat motivated by Herb Voelcker's concerns about tolerance schemes for controlling the geometric variability of mechanical objects. Our point here is that, over the last decades, cheaper computer memory (along with computer hardware advances) has allowed for more accurate approximate models (denser triangle meshes, finer voxels, deeper octrees). As a result, such approximate models are accurate enough for an increasing number of applications. At the same time, these representation schemes provide some advantages over their exact counterparts, such as implicit localization (voxels, octrees) and fast rendering (meshes). Section 3 gives a brief historical overview of surface reconstruction from discrete models (focusing on voxels). This problem is the counterpart of CSG Boundary Evaluation, in the realm of discrete volume models. Such a problem is key in a number of applications. In the context of CAD systems, surface reconstruction is used e.g. to extract the boundary from some design representations (F-Rep, implicit surfaces...), and as part of geometry processing operations such as repairing, remeshing and simplification. We consider the extraction of polygonal meshes from voxel data, focusing on topological ambiguity problems, as well as which voxels should be reconstructed to guarantee strict error bounds between the solid's boundary and the reconstructed surface. Section 3.7 deals specifically with the problem of extracting *smooth* surfaces from voxel representations, in the context of some specific fabrication applications. Section 4 presents some applications, such as additive manufacturing, where Boundary Evaluation is often no longer needed, and all tasks related to 3D printing are supported directly and efficiently by a discrete representation, like an octree. Therefore, in the context of *some* fabrication tasks, octrees and voxel representations are now seen as the *final* representation, in the sense that they are the only representation in this final manufacturing phase; no explicit boundary evaluation is needed beyond the design phase.

History is always surprising and unexpected. As we shall see, for some important CAD-related tasks such as 3D printing, the critical issues of "localization and boundary evaluation" have shifted to "localization and printing" with octrees and voxels used as a *final* representation. This path has involved many intermediate steps driven by new hardware, software and mathematical tools: volume representations, triangle meshes, out-of-core algorithms, and many others.

Section 4 also presents a number of relevant issues in applications that no longer use standard geometric representations of object boundaries (surfaces, primitives or triangle meshes), but rely directly on discrete volumetric representations such as voxelizations and voxel hierarchies. We also present a number of research challenges and several avenues for future research in these areas. Although not related to Computer-Aided Design, we have included volume-based techniques from the medical field, to show that voxel-based representations are also becoming a relevant tool for managing complex geometric models in areas beyond Solid Modelling.

2. Surface representations: Exact vs. approximate paradigms

The main challenge of Solid Modelling in Voelcker's time and more precisely in the 70s and 80s, was to encode shapes as complex as possible with compact, low memory footprint representations. Computer memory was limited, and geometric representations necessarily had to accommodate these constraints. This fact led to symbolic representations like Constructive Solid Geometry and to functional schemes including Bézier and Splines surfaces, depending on the application field. Both paradigms were continuous, allowing a theoretical infinite model precision.

The basic problem when dealing with Constructive Solid Geometry (CSG) was boundary evaluation. This was essential to obtain explicit surface representations for manufacturing and other applications. Numerical Control programs for automatic manufacturing soon demanded simple representations of the object surface, the triangle-based STL standard being a clear example, that were also considering tolerances.

Afterwards, geometric representations underwent a rapid evolution during the 90s and 2000s, due to the fast increase in computer memory sizes. Digital 2D images were followed by discrete spacial subdivisions like voxelizations and octrees, and

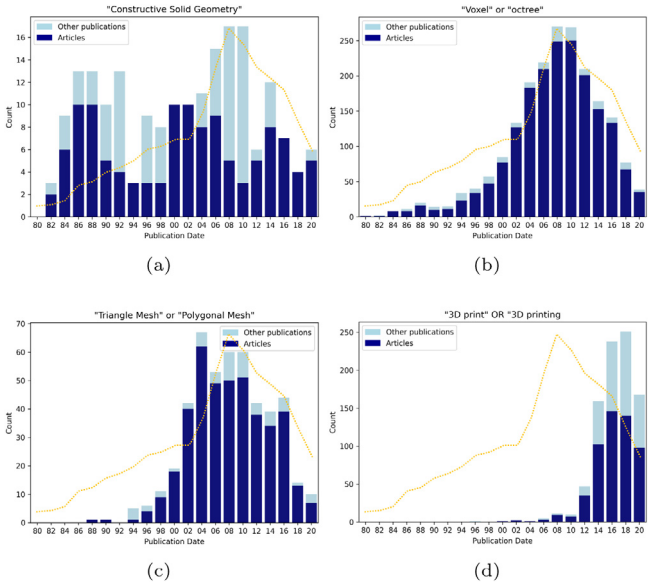


Fig. 1. Evolution of the number of publications with specific keywords in the abstract according to the *ACM Guide to Computing Literature* of the ACM Digital Library: (a) "Constructive Solid Geometry"; (b) "voxel" or "octree"; (c) "triangle mesh" or "polygonal mesh"; (d) "3D print" or "3D printing". We show the results in the 1980–2020 period, grouped every two years. Dark blue bars correspond to entries with "Article" or "Research-Article" publication type, whereas light blue bars refer to the rest of publications (theses, books, chapters, reports etc.). The yellow dotted line shows (in all graphs and in a different scale), the total number of publications in the ACM Digital Library, as this might partially explain the decrease in the number of publications in the last years, observed in all four searches. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

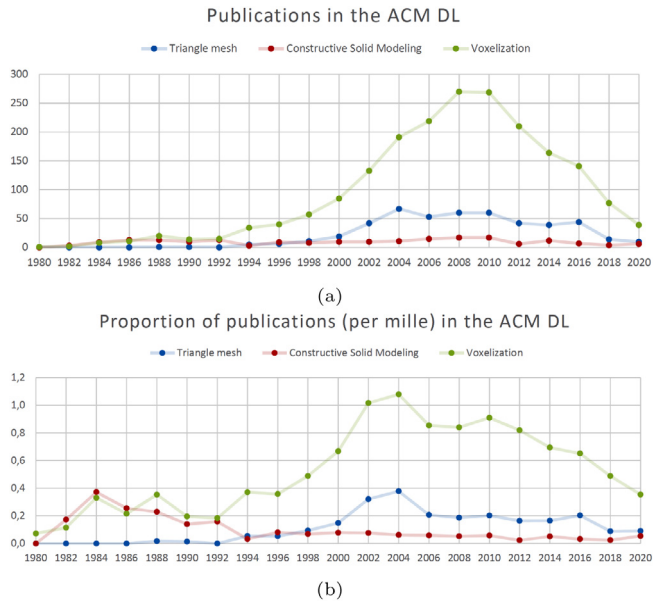


Fig. 2. Comparative evolution of the number of publications with specific keywords in the abstract according to the *ACM Guide to Computing Literature* of the ACM Digital Library. Unlike Fig. 1, we show both absolute number of publications *(a)* and the proportion of publications with respect to the total number of publications in the ACM Digital Library *(b)*.

boundary representations got approximate with the advent of Triangle Meshes. In a certain way, Geometry moved from the exact domain to the approximate, often discrete, domain. With voxels and triangle meshes, geometric and solid representations moved from the initial "infinite model precision", entering an approximate domain with bounded precision, in which the tolerance concept developed by Voelcker and his team was intrinsic.

To show the impact of these changes in Geometric Modelling and in the initial concepts worked out by Herb Voelcker and his group, we have performed an informal analysis of the evolution of the number of research papers in four areas: Constructive Solid Geometry, Voxels and Octrees, Triangle Meshes and 3D Printing. Fig. 1 shows that the number of papers including CSG in the abstract have been decreasing after the 80s, with a peak in the 2000s that could be in part explained by the use of CSG techniques in rendering. On the other hand, the terms "voxel", "octrees" and "triangle meshes" were gradually appearing while becoming standard representations in the research papers. Finally, the term "3D print" has become increasingly popular during the last decade. This fact will be discussed in Section 4. We do not have a complete explanation for the decline of all graphs in recent years, but it could be partially related to the decrease in the total of ACM DL publications, as shown by the yellow dotted line in Fig. 1. Anyway, this Figure shows a move from CSG to the approximate representations and from CNC manufacturing to novel 3D printing techniques. This decrease in the number of CSG-related papers is also visible in Fig. 2, that presents the evolution of the percentage of papers in the different considered areas with respect to the total number of publications in the ACM Digital Library.

Besides discrete models with bounded precision being well-suited for representing solids with tolerances—as already mentioned; they are also extremely useful for localizing the information to accelerate geometry processing. In the next section we will show how they can be used to physically generate the solid boundaries that Voelcker wished to evaluate. In some way, and after 40 years, discrete models are now addressing several of the main historical challenges in Solid Modelling.

3. Boundary evaluation on discrete models

3.1. The rise of triangle meshes and voxelizations

As the availability of computational resources increased, the compactness of the solid representation was no longer critical, and alternative representations for solids gained popularity. Triangle meshes (and in general polygonal meshes) gradually replaced CSG trees and other general boundary representations as the primary representation in some solid modelling applications. Although triangle meshes only provide an approximate representation of the boundary, being exact only for piece-wise linear surfaces, triangle meshes could successfully replace CSG trees whenever the application accuracy requirements did not result in meshes too dense to fit in memory. In parallel, voxel-based representations also gained support because of their conceptual simplicity and their intrinsic spatial localization properties, despite their discrete nature and despite the amount of memory required to represent arbitrary solids under a prescribed tolerance. In what follows, we will use the term **discrete model** to refer to models in which the coordinates of vertices and grid points have been discretized (i.e. we will not consider triangle and tetrahedral meshes as discrete models unless their vertex coordinates have been discretized). We will also refer to discrete models based on a regular space decomposition – including voxels and their hierarchical counterpart, octrees – as **discrete volume representations**

From a computational point of view, triangle meshes and voxel-based representations complement each other very well. Triangle meshes allow for a fast visualization of the solid's boundary in parallel hardware, whereas spatial localization algorithms and Boolean operations for voxel-based representations are robust and straightforward to implement. Due to their mutual

complementarity, the conversion from one representation to the other was increasingly needed to fully benefit from the operational advantages of both representation schemes. Moreover, tetrahedral meshes have been recently used in the context of boundary-to-CSG conversion [8] to accelerate geometric processing for additive manufacturing, as a tool to efficiently convert general meshed objects to non-discrete volumetric aggregate of tetrahedra (e.g. [9,10]), or to support Boolean operations [11] among other applications.

In this section we review, from a historical perspective, algorithms for extracting a polygonal mesh from a discrete solid representation. Notice that this problem is related with that of the Boundary Evaluation, but unlike in the work of Herb Voelcker on CSG trees, our starting point now is a discrete volume representation. The next subsections will focus on voxelizations, but most ideas also apply to hierarchical representations [12].

3.2. Types of voxelizations

There are two fundamental ways to conceptualize a voxelization. Both use a grid subdivision of the space with identical cubic cells. In the first conceptualization, a voxel represents a single sample, or data point, on a 3D grid; cubic cells thus have eight voxels (one for each vertex). From a solid modelling point of view, such voxels can be either inside, outside or on the boundary of the volume. Notice that this definition of voxel as data point allows voxelizations to represent also scalar fields, and thus its use also gained popularity beyond solid modelling, e.g. to represent Computed Tomography (CT) and Magnetic Resonance Imaging (MRI) scan data images. In the second conceptualization, a voxel represents one of the cubic cells of the 3D grid, and therefore the voxel label refers to the classification of such cell with respect to the solid. Such voxels thus can be completely inside, completely outside, or may contain part of the boundary of the solid.

Historically, one of the first algorithms for the extraction of polygonal meshes from a voxelization was the well known Marching Cubes algorithm [13], by Lorensen and Cline, who considered as input a discrete scalar field, i.e. a scalar field sampled on a regular 3D grid. If the scalar value at a voxel was higher than the user-defined isovalue (i.e. inside the solid) then the voxel was set to one (black), while if it was lower (outside), it was set to zero (white). Although Lorensen and Cline were mainly concerned with the efficient visualization of medical CT and MRI data, their original algorithm and the subsequent variants could be used for voxels just encoding their binary in/out classification with respect to a solid, or any other scalar property e.g. Signed Distance Fields (SDFs). Actually SDFs are extensively used today for representing implicitly shapes e.g. for additive manufacturing [14] and for learning generative models of shape families [15,16].

A key issue concerning boundary evaluation algorithms from discrete volume models is the kind of voxel conceptualization they assume (data point vs cubic cell), and the type of information stored at the data points (a binary value, a scalar value) or at the cubic cells. Let us now turn to the different categories of algorithms that perform this boundary evaluation.

3.3. Characterizing voxelization cells

Let V be an arbitrary solid volume, and let $S = \partial V$ be its boundary. Notice that, in general, discrete representations of V are lossy, in the same sense that polygonal meshes can only approximate surfaces with non-planar faces. Cubic cells c of a regular grid R can be classified into three types, depending on their intersection with V and S. A cell is *white* if completely outside the solid, *black* if completely inside, and *grey* otherwise.

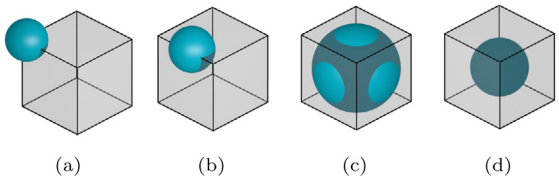


Fig. 3. Different types of grey cells depending on the solid volume shown in blue: (a) cell in $G_0 \dots G_3$, (b) cell in $G_1 \dots G_3$, (c) cell in G_2 and G_3, (d) cell in G_3. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

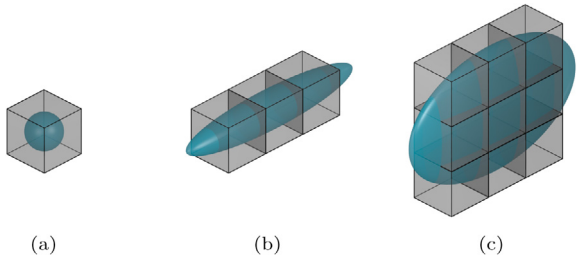


Fig. 4. Solid components (in blue) that result in different grey cells: (a) cell in $G_3 \setminus G_2$, (b) cells in $G_2 \setminus G_1$, and (c) cells in $G_1 \setminus G_0$.

Grey cells have a non-empty intersection with S, which can be used to define multiple subsets of grey cells (Fig. 3). G_0 **cells** are those cells having at least one vertex inside the solid V, and at least one outside the solid. G_1 **cells** are those cells having at least one edge intersected by S. Similarly, G_2 **cells** have at least one face intersected by S. Finally, G_3 **cells** are those intersecting S, and therefore all grey cells are G_3.

These subsets form a hierarchy, $G_0 \subset G_1 \subset G_2 \subset G_3$, as we shall consider grid components (edges, faces and cells) as closed sets, [17]. This hierarchy corresponds in turn to a hierarchy of surface reconstruction algorithms, to which we now turn our attention.

3.4. Characterization of surface reconstruction algorithms

Given a discrete representation of an arbitrary volume V, a fundamental problem is the computation of a surface $\tilde{S}$ that approximates the (unknown) boundary S of V. From now on we will refer to this problem as *Surface Reconstruction* or *Surface Extraction*. Although the surface reconstruction problem is quite similar to the *boundary evaluation* problem studied by Voelker, we will use the latter only when dealing with representations (e.g. CSG) that provide an exact representation of the underlying volume.

From a surface reconstruction point of view, it is important to pay attention to those cells containing part of the solid boundary (thus in G_3) but not in the other subsets. In particular, cells in $G_3 \setminus G_2$ contain isolated portions of S in its interior (Fig. 4(a)). Similarly, cells in $G_2 \setminus G_1$ contain pipe-like portions of S that enter and exit through faces of cells without disturbing any edges (Fig. 4(b)), and cells in $G_1 \setminus G_0$ contain thin portions of S that intersect edges and faces without enclosing any grid vertex (Fig. 4(c)).

The subsets above allow us to classify surface reconstruction algorithms. According to the input, a surface reconstruction algorithm is a G_k algorithm if it only sees (takes as input) the cells in G_k, but not in G_m, for any $m > k$. Obviously, G_k algorithms with $k < 3$ will miss some small portions of the solid, as those depicted in Fig. 4, and will not be able to offer a complete reconstruction unless the cell-size ℓ is chosen so small that the model contains no G_3 cells.

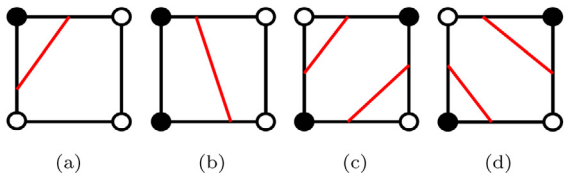


Fig. 5. Unambiguous faces (a, b) generating a single edge of the triangular mesh and the two possible choices of edges in an ambiguous face (c, d).

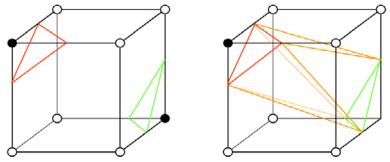


Fig. 6. The two possible topologies arising from an X-cube.

Concerning the output, we say that a surface reconstruction algorithm provides a *k-reconstruction* iff the output surface is completely contained in the union of G_k cells and the output volume intersects *all* cells in G_k. For example, G_3 algorithms may still miss some of the small portions in Fig. 4 if they do not guarantee a 3-reconstruction. Actually, only 3-reconstructions guarantee a strict bound on the Hausdorff distance between the volume V and the reconstructed volume $\tilde{V}$ [17]. In the next subsections we review surface reconstruction algorithms according to the classification above.

3.4.1. 0-reconstruction algorithms

Most of the surface reconstruction algorithms from voxel data [18–26] are variations of the original Marching Cubes algorithm [13]. When seen from the point of view of solid modelling (most of them assume discrete scalar fields that are binarized through an isovalue) they have a common set of characteristics: the input information is the classification of the grid vertices with respect to the volume (they only detect G_0 cells), and the output surface is a triangle mesh reconstructed locally for each G_0 cell. Each triangle of the final mesh belongs to a unique cell, and the output surface $\tilde{S}$ intersects only once each black-white edge of the grid. These properties guarantee that the output surface stabs all G_0 cells, so all these algorithms provide 0-reconstructions.

The main difference among these algorithms lies in the method they use to perform the local reconstruction of the triangle mesh M. There are two decisions to consider: the selection of the local topology of M in a cell and the triangulation of the resulting connected components (sheets). The only configurations giving choices to control the local topology [26] are those that correspond to cells having ambiguity faces (also called *X-faces*, see Fig. 5), or having only two diagonally-opposite *black* vertices (also called *X-cubes* (Fig. 6). Fig. 7 shows some representative cases for the different cell types.

We can group the Marching Cubes-based algorithms in three families according to how they resolve the local ambiguities: those using a tetrahedral subdivision, those using a Single-entry Look-Up Table (LUT), and those using (implicitly or explicitly) a Multiple-entry LUT. Single-entry LUTs only use the black/white classification of the grid vertices, while Multiple-entry LUTs use the scalar values at grid vertices and thus assume non-binary voxel data.

Tetrahedral subdivision approaches [21–23] perform a conformal subdivision of the G_0 cells. These methods yield directly a valid two-manifold surface, but the tetrahedral subdivision determines the topology inside each cube.

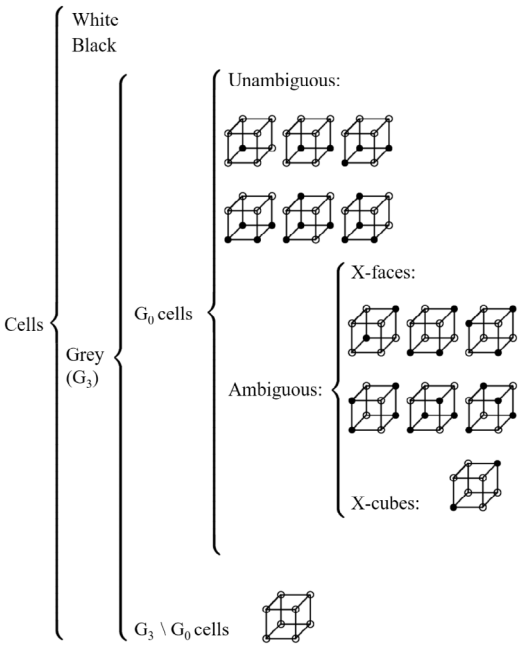


Fig. 7. Representative configurations of cubic cells for 0-reconstruction algorithms.

In Single-entry LUT approaches, the local topology of a G_0 cell is univocally defined by the classification of its vertices. Some proposals do not compute the LUT explicitly but they take an equivalent decision algorithmically [18]. All these methods can achieve consistent topologies on ambiguous faces. The triangulation of a configuration can be done by always locating the mesh vertices in the black-white grid edges [27] or by inserting additional vertices into the cell for recovering sharp features of the original surfaces [24], using for example Hermite data (position of intersection points of the original surface with the grid edges, together with their normals). Single-entry algorithms obtain valid and locally consistent surfaces but they do not have control on the global topology of $\tilde{S}$ which is instead determined by the preferred local reconstruction stored in the LUT (i.e. they cannot decide the number of holes or shells of $\tilde{S}$).

Multiple-entry LUT approaches try to reproduce in each cell the topology of a bilinear or trilinear interpolation function [28–33]. From the vertex values, they compute the value of the function in its saddle points on the cell and its faces. The classification of the grid vertices allows to choose a basic MC configuration and the computed saddle points allow to choose among all the possible topologies. Ho et al. [25] used Hermite data stored at the grid vertices to decide the topology in ambiguous cases, and recover local sharp features by introducing mesh vertices in G_0 cells. All the algorithms in this group obtain valid, consistent and correct surfaces (according to the preferred function or shape) but do not have a global control on the topology of the final mesh. The Dual Contouring algorithm [34] creates a vertex of the final mesh for each G_0 cell. Based on the hypothesis that each black-white edge of the grid stabs once the final mesh, the algorithm generates a quad by joining the four points located in the cells sharing the edge. This reconstruction method guarantees that the output surface stabs all G_0 cells, so the algorithm is a 0-reconstruction. Additionally, to the black/white classification of the grid vertices, the input data includes Hermite data which is used to properly place the output vertices. This algorithm creates surfaces with non-manifold topology in ambiguous cells (see Fig. 9(d)).

The selection of a valid two-manifold topology according to a desired topological complexity (such as number of connected components, or the total genus) was explored in [26]. The paper identifies two independent ways to control the topology of the final mesh, by deciding how to slash the *X-faces* (see Fig. 5) and by deciding to have one or two sheets in an *X-cube* (see Fig. 6). For non-ambiguous cells the reconstruction follows the classical MC algorithm. For the ambiguous cells, in order to take locally and consistent decisions with respect to the user-desired global topology, the paper uses a graph encoding X-faces and a merge tree of equivalence classes of vertices. This method optimizes some properties of the final mesh, for example the number of triangles and the number of connected components.

3.4.2. 1-reconstruction algorithms

Varadhan [35] uses directed distances at grid vertices to perform an exact edge-intersection test. This test is used to detect complex edges, i.e. edges intersected by the surface but not exhibiting a sign change (and thus obtaining G_1 cells). The reconstruction algorithm is very similar to the Dual Contouring algorithm [34] but it considers that an edge can have up to two intersection points and that there can be multiple error-minimizing vertices per cell. Once the edge intersection points have been computed, it separates them into components achieving a two-manifold surface. It also reconstructs some sharp features, but does not have a global control on the topology of the final mesh.

3.4.3. 3-reconstruction algorithms

An extension of the method above [35] is presented in [36]. Instead of considering just complex edges (leading to G_1 cells) G_2 and G_3 cells are also identified and recursively subdivided. A cell is complex if it has a complex voxel, face, edge, or an ambiguous sign configuration. A cell (face) is defined to be complex if it intersects the surface and the grid vertices belonging to the cell (face) do not exhibit a sign change. Their algorithm subdivides recursively the space using an octree until all cells are not complex and satisfy a star-shaped criterion. The proposed reconstruction algorithm is able to preserve the original topology of the surface but at the expense of an arbitrary number of subdivisions.

Andujar et al. [37] propose another G_3 surface extraction algorithm. Given a volume V bounded by a surface S, the intersection between the cells of the grid and S is detected and for each of these G_3 cells the classification of their vertices with respect to the volume is computed and stored. The surface reconstruction algorithm first performs exactly one additional subdivision of the G_3 cells that have at least one white neighbour cell. The classification of the new grid vertices is computed using the type of their neighbour cells. If a vertex has a white neighbour cell, it is classified as white. Otherwise, it is classified as black. The central vertex of a subdivided cell is always classified as black. Resulting subcells do not have ambiguous faces and their topology is decided using the LUT of the basic MC algorithm [13].

3.5. Comparison of methods

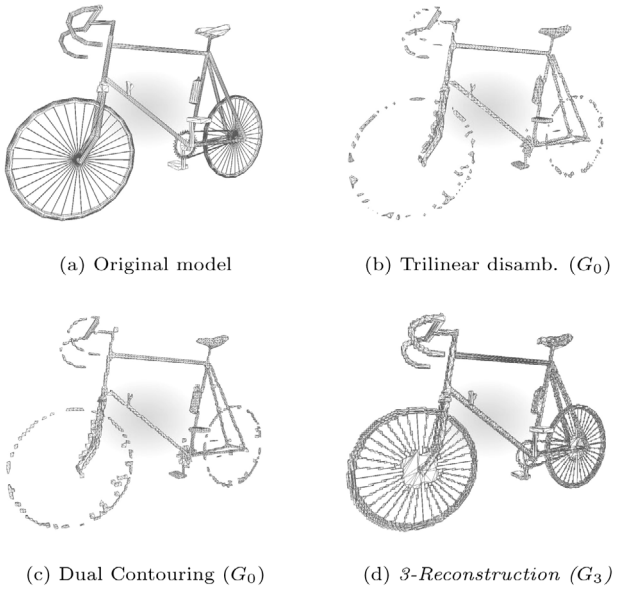
Table 1 provides a comparison of the surface reconstruction methods discussed above. The **cells** column indicates the type of G_k cells considered by the algorithm. Notice that methods designed for scalar fields (e.g. CT and MRI data) only see data at grid points and thus are G_0. The next column refers to the information the algorithms use for **disambiguation** of ambiguous cases, using for example the binary (black/white) classification of the grid vertices or scalar values at the grid vertices.

The column on **topological control** summarizes how the different algorithms deal with topology. Only [26] can perform a global optimization of the topology, whereas the other algorithms take local decisions at cell level. Some methods [22,23] resort to a tetrahedral subdivision, thus arbitrarily fixing the topology within each cell. Others use extended lookup tables [24,27], or attempt to reproduce the topology of a trilinear interpolant for the scalar values at the vertices [30,32,33]. The approach used in [25] exploits the Hermite data to decide the topology within each cell. In the case of [34], ambiguous cells give rise to unambiguous triangulations which are non-manifold. The rest of algorithms guarantee a **two-manifold** surface.

The last column indicates the type of **output** generated by the algorithms. Only a few methods use all G_3 cells, being able to reconstruct *thin* parts of the model (Fig. 4) and providing a 3-reconstruction algorithm [36,37]. The rest of the algorithms use G_0 cells and thus cannot reconstruct *thin* parts of the volume.

Fig. 8 illustrates the reconstruction differences between G_0 and G_3 approaches when the volume includes thin parts (e.g. the spokes). Algorithms based on G_0 cells are not able to reconstruct the complete surface of the model, unless the number of subdivision levels is high enough.

Fig. 9(a) compares different reconstruction methods on a discrete model with multiple ambiguous cases. The algorithms using an alternating tetrahedrization (Fig. 9(b)), sometimes join the ambiguous cubes and sometimes separate them, but the decision adopted has no relation with the topology of the original surface. Algorithms based on a trilinear disambiguation (sub Fig. 9(c)) are at an advantage with this test case, as the very smooth and regular geometry is approximated very well by a trilinear surface in each cell. Note that algorithms based on dual contouring produce non-manifolds on ambiguous configurations (see Fig. 9(d)). The last two reconstructions correspond to [26] using two different optimization strategies: minimizing the number of solid components (Fig. 9(e)) or maximizing them (Fig. 9(f)). This last strategy is the only one recovering the initial two shells.



(a) Original model (b) Trilinear disamb. (G_0)

(c) Dual Contouring (G_0) (d) *3-Reconstruction* (G_3)

Fig. 8. Different results of G_0 and G_3 algorithms.

3.6. Related methods

Here we review some recent surface reconstruction methods that, although requiring special input, training, or optimization steps, can be used to create a polygonal mesh from a discrete

Table 1
Comparison of surface reconstruction algorithms.

Methods	Cells	Disambiguation	Topol. control	Two-manifold	Output
Gueziec-95 [22] Pascucci-04 [23]	G_0	binary B/W	Local (alternancy based)	Yes	0-*rec*.
Montani-94 [27] Kobbelt-01 [24]	G_0	binary B/W	Local (LUT based)	Yes	0-*rec*.
Cignoni-00 [30] Lopes-03 [33] Nielson-03 [32]	G_0	vertex values	Local (trilinear based)	Yes	0-*rec*.
Ho-05 [25]	G_0	binary B/W + hermite data	Local (hermite based)	Yes	0-*rec*.
Tao-02 [34]	G_0	binary B/W	Local (non-manifold)	No	0-*rec*.
Varadhan-03 [35]	G_1	directed distances	Local	Yes	1-*rec*.
Andujar-02 [37]	G_3	binary B/W extra subdiv.	Local (W-surface)	Yes	3-*rec*.
Varadhan-04 [36]	G_3	directed distances	Local	Yes	3-*rec*.
Andujar-05 [26]	G_0	binary B/W	Global (optimization)	Yes	0-*rec*.

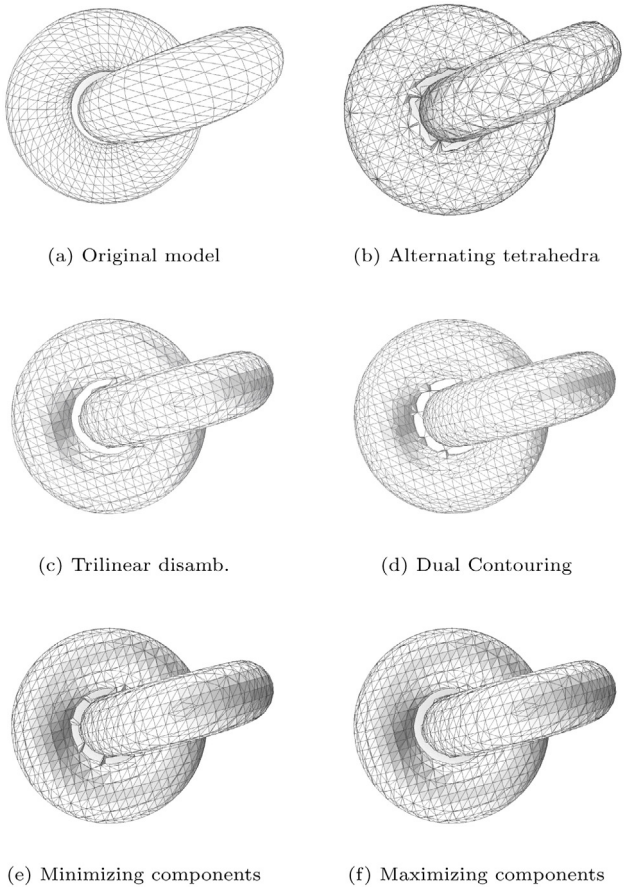


Fig. 9. Different results from different G_0 algorithms. The model has been generated with five octree subdivision levels.

volume representation. Lopez et al. [38] extend the idea of Marching Cubes [13] to extract isosurfaces from general polyhedral cells (besides cubic cells). The isosurface is extracted using a polygon tracing procedure, which is valid for convex or non-convex cells, and produces consistent results even for ambiguous configurations. Several extensions of Marching Cubes [13] and Dual Contouring [34] have been proposed to create smoother meshes. Nielson et al. [39] optimize the position of the final vertices (constrained to the cell edges) by optimizing a certain energy functional. Coeurjolly et al. [40] apply a similar idea to Dual Contouring meshes [34]. The method adjusts the 3D position of the output vertices by minimizing an energy function that includes a term to prevent vertices to be too far away from the surface, an alignment term to make the quads match the normal vectors, and a fairness term to create a smoother quad mesh.

Since implicit representations are suitable for representing 3D shapes in deep learning approaches, many recent methods focus on the surface reconstruction problem so that these architectures can be trained end-to-end. Deep Marching Tetrahedra [41] uses a deformable tetrahedral grid to encode a discrete SDF together with a marching tetrahedra layer that converts the SDF to a polygonal mesh. The method uses a deep 3D conditional generative model to synthesize high-resolution 3D shapes from a coarse voxelization. However, the method requires training on a dataset with the target shapes. Voxel2Mesh [42] is an end-to-end trainable architecture for surface reconstruction. It takes as input a discrete volume and a spherical mesh, which are jointly encoded and then decoded into volumes and meshes of increasing resolution. At each mesh decoding stage, the decoder uses a set of features sampled from the volume to refine it by adding vertices where they are needed. The output mesh though is constrained to have a spherical topology. Liao et al. [43] propose a differentiable version of Marching Cubes so that triangle meshes can be predicted directly from volumetric data, allowing for end-to-end training. They identify two major reasons why the original MC approach is not differentiable. First, because the location of each vertex along cell edges (computed through linear interpolation) is singular when the edge endpoints have the same value. Second, voxel values affect only grid cells in their immediate vicinity and thus gradients would not propagate to cells further away from the predicted surface. The authors propose a differentiable approach which separates the mesh topology from the geometry. Instead of predicting signed distance values, they predict the probability of occupancy for each voxel, as well as the vertex location within cell edges. The resulting Differentiable Marching Cubes can be integrated as a final layer in deep learning architectures that generate an explicit surface representation, although only non-ambiguous configurations are handled by the method.

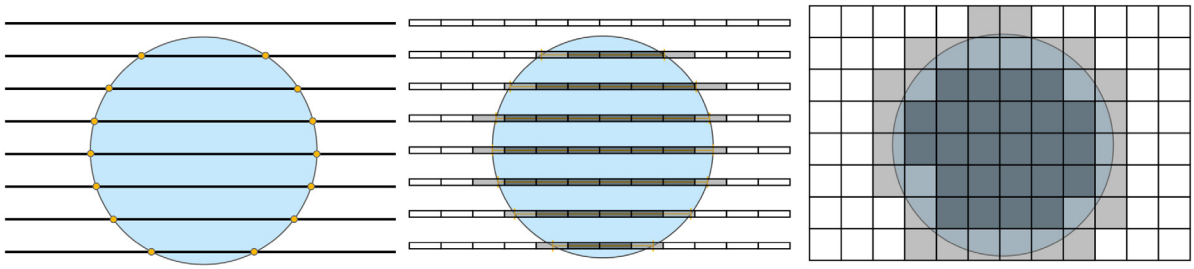


Fig. 10. Slicing a sphere. *Left:* By classifying planes, each slice results from the intersection of a plane and the model. *Middle:* By discretizing the planes we can classify each slice pixel into white (outside), grey (boundary), and black (inside). *Right:* By classifying cubical voxels with G_3-based algorithms, and assigning this classification to the pixels of the slices, the result is different. Observe that in this last case, slice pixels can inherit the corresponding volumetric cell classification of their associated voxel. The top slice, for instance, is completely White in the *Middle* case while it correctly shows two Grey pixels in the *Right* case. Moreover, in the *Middle* case there is an abrupt White-to-Black transition between the two top slices, which is unacceptable in the printing process (in closed objects, White and Black cells must always be separated by Grey cells containing part of the object boundary).

3.7. Smooth BReps from discrete models

Surface reconstruction algorithms, as discussed above, were essential for digital modelling and fabrication. In this process, a model is designed using CAD tools, and is then used to guide the automatic manufacturing process. The first set of techniques to be introduced were based on subtractive manufacturing using CNC machines [44]. Models used as input were commonly modelled using either B-splines or triangular meshes. In this phase, voxelizations and octrees [45] were reserved for their exclusive use as acceleration structures for the computations necessary to guide the machining process. Moreover, specific discrete hierarchical representations like extended octrees were proposed for the boundary evaluation of certain kinds of CSG trees [7], also helping in performing the explicit Boolean operations [46] required in the fabrication pipeline.

Nevertheless, as already discussed, discrete volumetric models began to gain prominence due to their medical applications [47] and their ability to represent the internal structure of the models, prosthesis manufacturing being a clear example of CAD-related tasks coming exclusively from voxelizations. Anyway, their use for fabrication continued to be limited by their difficulties to represent high quality smooth surfaces and small details. However, converting a volumetric model to a triangular mesh using the algorithms cited in the previous subsections was a possibility in manufacturing applications.

Still, the error measured as the distance between the original object and the surface reconstructed from the corresponding volumetric data was limited by the quality of this volumetric data, which in some cases could be binary. Due to this, various techniques arose, capable of producing smooth meshes from imperfect volumetric data. Ohtake and Belyaev [48] introduced a method to obtain a smooth mesh from an implicit function. Starting with the extraction of a mesh from a binary discretization, their algorithm smooths the mesh while keeping it very close to the isosurface of the input implicit function. Gibson [49] also proposed an algorithm based on applying Laplacian smoothing to the mesh extracted from the input volume, but restricting the vertices to the voxels that generated them. Whitaker [50], on the other hand, proposed to calculate a level set whose zero-isosurface had minimum area and respected the voxel classification of the input volume. Instead, Nielson et al. [39] developed an algorithm to optimize the meshes extracted using Marching Cubes by applying a Laplacian smoothing operator with the vertices restricted to the edges of the voxelization from which they were extracted. Finally, Chica et al. [51] extended the previous method by applying constrained Laplacian smoothing to various areas of the model independently. In this way, it is possible to detect flat and smooth areas separated by sharp edges.

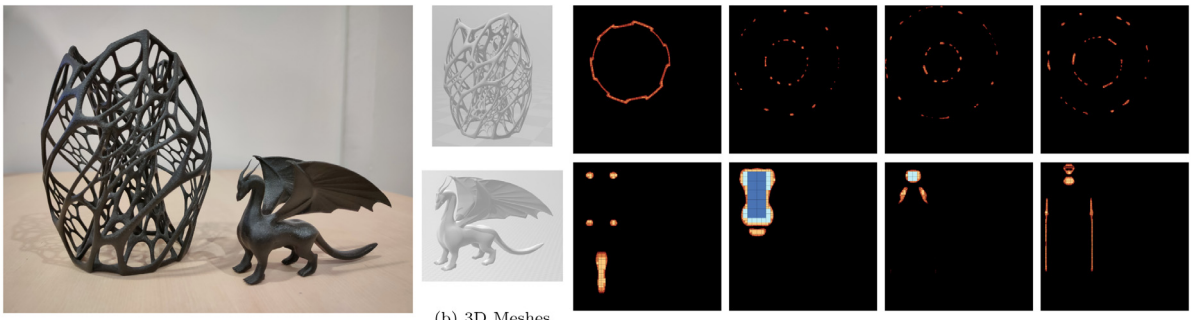
4. Voxels as a final representation

In the last decades, and in certain applications, discrete volume representations such as voxels and octrees are used as the final representation, thus avoiding the need for a boundary evaluation. This is a direct consequence of new hardware developments and the increase of storage capacity. To some extent, the boom of digital 2D images was followed by a boom on discrete volume representations. And, after pixel-based processing algorithms, we are now observing the blossoming of algorithms that work directly on voxels.

In additive manufacturing, 3D printing systems work by adding material in layers [52,53]. Each layer is represented by a 2d image and it is computed by intersecting the layer plane with the model mesh [54] in a process known as slicing. This operation is however not as trivial as it might look, and a naive implementation could produce non watertight results (see Fig. 10). But capturing all image pixels that intersect the surface is crucial in applications where precision is of paramount importance (e.g. medicine and engineering). Notice that only G_3-based reconstruction algorithms are guaranteed to fulfil this requirement (see Table 1 and Fig. 3).

For some technologies, like Fused Deposition Modelling (FDM), a toolpath needs to be derived from the slices, so that the model may be fabricated [55]. In this case, using representations as CSG combinations of unions of cells, each of them being a CSG solid [8] can be wise. For others, like Multi Jet Fusion (MJF), this step is not necessary.

Regarding recent 3d printing techniques, objects usually represented by triangle meshes may be converted into an octree that has the necessary depth according to the design tolerances and printing accuracy (i.e. voxels and terminal octree nodes have the same size as the microdots of deposed material) without any loss. Then, the octree representation is used to generate the sequential slices required by the printer. In this context, the authors [56] present the calculation and coding of a new linearization of an octree specially conceived so that sequential slicing is efficient. The result is a G_3-based encoding algorithm that is output-data bounded, such that we can guarantee a maximum extraction time for each slice. In fact, the resulting octree is also compact, which increases performance when reading it from disk, and successive slices can be extracted during printing efficiently. In this case the smoothing that was needed when dealing with discrete volumetric models is no longer required, partly because the volumetric elements already represent the smallest amount of material that can be added, and partly because the smoothing occurs for other reasons, for example due to thermal diffusion during the printing process, or by some mechanical post-processing of the part. Fig. 11 shows the result of printing two models using an octree as representation. Some 2D slices of the octree are also shown.



(a) Printed Models (b) 3D Meshes (c) Some slices

Fig. 11. *Left:* Two 3D models printed using an octree representation. *Right:* Slices generated from the octrees of both models. Nodes are drawn in a colour scale with blue for large nodes and red for small ones. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Also, by storing and using an octree representation of the objects, low-resolution models can be produced on the fly during interactive inspections, and are therefore also useful to quickly manufacture low-resolution parts for testing.

However, and in spite of recent progress, present research challenges are multiple. Additive manufacturing has become more than a prototyping solution [52]. Even so, research directed to improve its capabilities and the quality of the results is still required:

Toolpath optimization and orientation planning Once slices have been generated, and depending on the technology being used, a toolpath has to be generated. This toolpath may be optimized to reduce void formation [57] and thus undesired porosity. It may also be planned so that the orientation of the material can be aligned with specific directions, or is distributed uniformly [58], which may help control the resulting mechanical properties. Changing the orientation of the nozzle or even its shape should provide more flexibility. Trying to find orientation plans that optimize surface and mechanical characteristics of the final parts is an interesting avenue for future modelling research.

Optimal object partitioning and supports Support structures increase material costs, printing time, require additional labour to remove them, and may leave behind visible defects. An alternative approach is to partition the model to balance the amount of supports and cuts [59], or even look for a partition that allows for support-free printing [60]. In the case of hollowed objects, there is also the possibility of having internal supports, so that they remain invisible [61]. Introducing metrics that take perceptual, structural, or mechanical properties into account could provide additional improvements. Also, multi-material printing could produce supports that can be efficiently dissolved in post-processing. In this case, the investigation of optimal partitions of the geometric representations could be a fascinating geometric topic.

Physical surface bounded fairing When using digital light stereolithography it is possible to use greyscale pixels in the images provided to the printer. This allows for sub-voxel growth, which may be exploited to control the final surface appearance [62]. With other techniques like Multi Jet Fusion, thermal processes smooth out the staircase effect usually associated to 3D printing. Thus, adapting to the printing technology used offers the potential to help produce sub-voxel precise results and physical-based surface fairing. The correlation between subtle modifications in the Grey layer of the voxel models and the final smoothness of the manufactured object surface opens also the door to future research.

Design of internal microstructures Lattices have also demonstrated their usefulness in weight and used material reduction, as well as in energy absorption applications. In particular, cellular materials [63] may be used to modify the elastic response of printed parts. Other more complex microstructures [64] allow for orientation control of the deformation of the resulting material. Using a voxelized representation provides full control of the volumetric properties of the model, but the complexity of optimization algorithms increases with the number of voxels. One solution is to work in two levels so that cells in the macroscale level correspond to blocks of voxels on the microscale one [65]. The range of possible lattice and microstructure configurations is vast, and its combination with multimaterial manufacturing makes it even larger. Exploring this space could shed light on the possible spectrum of achievable behaviours.

Surface modelling Materials commonly used in additive manufacturing are translucent. Compensating for the resulting scattering effects produces much better colour reproduction [66]. The microstructure of the layer-by-layer approach also tends to produce dull surfaces. Varnishes are one solution, and its controlled application [67] produces spatially-varying gloss properties. It is even possible to optimize the printed model, bridging the gap between it and the intended appearance [68]. On the other hand displacement maps may be used to add detail at the meso-scale or print curved triangles [69]. They may also be used to add codes that identify identical parts [70], or produce tactile textures [71]. All these surface properties are coupled. Thus, future work on these issues and how they relate will be critical in order to achieve high-quality reproductions.

Controlled multi-material printing As 3D printer manufacturers fit their systems with sensors, in-process monitoring of the fabrication will be feasible. Moreover, multi-material 3D printing makes the manufacturing of composite materials possible with controlled thermal and mechanical responses. It is also beginning to allow the integration of sensors, circuits and wiring. Designing and planning models with these characteristics while taking the limitations of the chosen 3D printing technology into account will be a challenging task.

On the other hand, in medical visualization, direct volume rendering can be even more useful than showing organ boundaries obtained with Marching Cubes algorithms, for the simple reason that medical organs are essentially volumetric, so that their internal structure is relevant for the experts and clinical doctors. Moreover, these voxel models are "closer" to the raw input data, avoiding reconstruction errors and biases. Therefore, medical applications are also moving to voxel-based representations. But, despite this move from continuous surfaces and meshes to the direct use of discrete voxels, a number of research issues to enlarge and improve the capabilities of these novel medical systems are appearing, with clear challenges in the geometric and modelling domain,such as:

Modelling volume uncertainty Medical image acquisition systems are based on different physical principles which allow to

capture information related with internal anatomical structures. However, the acquisition process is affected by the uncertainty of several factors such as the reconstruction process (which includes the sample location, the transformation of the measured signal to data, and the machine resolution) and the patient movement. Usually the captured data is represented as a voxel model. Computational models are used to estimate the uncertainty of the acquisition systems. In recent years, different approaches to quantify and visualize per voxel uncertainty have been proposed by combining an uncertainty model with captured data [72,73]. However, it is still a challenge to obtain efficient and optimal algorithms to estimate and model the uncertainty not only in the captured data but in all the transformations of the volume model, by encoding neighbourhood information in different input scalar fields. Naturally, accuracy has vital importance in medical applications.

Volume transformation Combining data from different acquisition systems provides complementary information which allows to improve medical diagnosis. For this purpose, volume models with different resolutions and heterogeneous data need to be registered. Geometric transformations, feature extraction and data interpolation may include uncertainty and so they require special attention [74]. Another volume processing transformation is segmentation, which allows to identify particular structures [75]. The obtained voxel models may be directly used in 3D printing for prosthesis design or for computer aided surgery.Current and future trends for both registration and segmentation tasks show an increasing and promising avenue for machine learning approaches.

5. Conclusions

In this paper, we have presented some aspects related to the evolution of the use of Solid Modelling for manufacturing along the last 40 years, moving from the symbolic representations and continuous boundary evaluation algorithms that inspired Herb Voelcker's work, to approximate paradigms based on triangle meshes, intermediate voxel and octree representations, and to new applications like additive manufacturing techniques and medical imaging, where boundary evaluation is no longer required. This transition has been motivated by improvements in computing power and memory storage, as well as the development of technologies like additive manufacturing and 3D digitization.

After characterizing voxel models, the paper also presents a number of surface reconstruction algorithms that work from discrete representations, and represent the counterpart of boundary evaluation for discrete models. We observe that discrete representations have been incorporating Voelcker's tolerances in an implicit way.

Based on a number of properties of the Grey cells and of the reconstruction algorithms, we have proposed a characterization of several surface extraction algorithms. The classification in Table 1 presents the inherent limitations of the different algorithms concerning global topology control and reconstruction of local features like thin portions of the volume and almost non-manifold regions. These limitations have been discussed with some practical examples.

However, recent manufacturing techniques and medical applications are again replacing in some sense these reconstruction algorithms, directly using discrete representations. In fact, we note that many novel medical 3D inspection applications and novel additive manufacturing techniques based on 3D printing do not require an explicit boundary evaluation. Medical diagnosis and planning systems are offering high-quality interaction experiences by directly using the discrete voxel representations, and recent 3D printing systems physically generate the boundaries of the Solid Models that Herb Voelcker wished to evaluate, also profiting from thermal smoothing processes that use diffusion and other techniques to obtain final manufactured objects with surfaces that meet requirements and tolerances. We have also presented a number of research challenges and several avenues for future research in these areas. We conclude that during the last 40 years, the area has evolved from Voelcker's localization and boundary evaluation algorithms to the direct use of discrete volume representations in cases like advanced 3D interaction and direct and physical additive manufacturing of objects with smooth surfaces, evolving, in the case of 3d printing, from boundary evaluation to voxel-based physical boundary printing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

Project TIN2017-88515-C2-1-R funded by MCIN/AEI/10.13039/501100011033/FEDER"A way to make Europe"

References

[1] Voelcker HB, Requicha AA. Geometric modelling of mechanical parts and processes. 1977, URL http://hdl.handle.net/1802/26357.
[2] Requicha AA, Voelcker HB. Boolean operations in solid modelling: Boundary evaluation and merging algorithms. 1984, URL http://hdl.handle.net/1802/1179.
[3] Tilove RB, Requicha A, Hopkins MR. Efficient editing of solid models by exploiting structural and spatial locality. Comput Aided Geom Design 1984;1(3):227–39.
[4] Meagher D. Geometric modelling using octree encoding. Comput Graph Image Process 1982;19(1):129–47.
[5] Brunet P, Navazo I. Solid representation and operation using extended octrees. ACM Trans Graph 1990;9(2):170–97. http://dx.doi.org/10.1145/78956.78959.
[6] Samet H, Webber RE. Hierarchical data structures and algorithms for computer graphics: I. fundamentals. IEEE Comput Graph Appl 1988;8(3):48–68.
[7] Navazo I, Fontdecaba J, Brunet P. Extended octtrees, between CSG trees and boundary representations. In: EG 1987-technical papers. Eurographics Association; 1987, p. 239–47. http://dx.doi.org/10.2312/egtp.19871018.
[8] Kurzeja K, Rossignac J. CSG combinations of tran-similar two-patterns of CSG cells. Comput Aided Des 2022;146(May). 10.1016/j.cad.2022.103212.
[9] Hu Y, Zhou Q, Gao X, Jacobson A, Zorin D, Panozzo D. Tetrahedral meshing in the wild. ACM Trans Graph 2018;37(4, Article 60):1–14.
[10] Diazzi L, Attene M. Convex polyhedral meshing for robust solid modeling. ACM Trans Graph 2021;40(6, Article 259):1–16.
[11] Zhou Q, Grinspun E, Zorin D, Jacobson A. Mesh arrangements for solid geometry. ACM Trans Graph 2016;35(4):1–15, 39.
[12] Newman TS, Yi H. A survey of the marching cubes algorithm. Comput Graph 2006;30(5):854–79.
[13] Lorensen W, Cline H. Marching cubes: A high resolution 3D surface construction algorithm. Comput Graph 1987;21(4):163–9.
[14] Brunton A, Rmaileh LA. Displaced signed distance fields for additive manufacturing. ACM Trans Graph 2021;40(4). http://dx.doi.org/10.1145/3450626.3459827.
[15] Chen Z, Zhang H. Learning implicit fields for generative shape modeling. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019, p. 5939–48.
[16] Park JJ, Florence P, Straub J, Newcombe R, Lovegrove S. Deepsdf: Learning continuous signed distance functions for shape representation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019, p. 165–74.
[17] Andujar C, Brunet P, Fairen M, Navazo I, Vinacua A. A topological comparison of surface extraction algorithms. Tech. rep., Universitat Politècnica de Catalunya; 2006, URL.
[18] Bloomenthal J. An implicit surface polygonizer. In: Heckbert PS, editor. Graphics gems IV. Academic Press; 1994, p. 324–49.
[19] Lachaud J-O. Topologically defined iso-surfaces. In: Proc. 6th discrete geometry for computer imagery. Berlin: Springer-Verlag; 1996, p. 245–56.

[20] Montani C, Scateni R, Scopigno R. A modified look-up table for implicit disambiguation of marching cubes. Vis Comput 1994;10(6):353–5.
[21] Zahlten C. Piecewise linear approximation of isovalued surfaces. In: Post FH, Hin AJS, editors. Advances in scientific visualization. Springer-Verlag; 1992, p. 105–18.
[22] Gueziec A, Hummel R. Exploiting triangulated surface extraction using tetrahedral decomposition. IEEE Trans Vis Comput Graphics 1995;1(4):328–42. http://dx.doi.org/10.1109/2945.485620.
[23] Pascucci V. Isosurface computation made simple: Hardware acceleration, adaptive refinement and tetrahedral stripping. 2004, URL http://www.pascucci.org/pdf-papers/vissym-2004.pdf.
[24] Kobbelt LP, Botsch M, Schwanecke U, Seidel H-P. Feature sensitive surface extraction from volume data. In: Proceedings of the 28th annual conference on computer graphics and interactive techniques. New York, NY, USA: Association for Computing Machinery; 2001, p. 57–66. http://dx.doi.org/10.1145/383259.383265, 10.1145/383259.383265.
[25] Ho C-C, Wu F-C, Chen B-Y, Chuang Y-Y, Ouhyoung M. Cubical marching squares: Adaptive feature preserving surface extraction from volume data. Comput Graph Forum 2005;537–45. http://dx.doi.org/10.1111/j.1467-8659.2005.00879.x.
[26] Andújar C, Brunet P, Chica A, Navazo I, Rossignac J, Vinacua A. Optimizing the topological and combinatorial complexity of isosurfaces. Comput Aided Des 2005;37(8):847–57. http://dx.doi.org/10.1016/j.cad.2004.09.013, 10.1016/j.cad.2004.09.013.
[27] Montani C, Scateni R, Scopigno R. Discretized marching cubes. In: Proceedings visualization '94, vol. 1. Los Alamitos, CA, USA: IEEE Computer Society; 1994, p. 281–7, CP32. http://dx.doi.org/10.1109/VISUAL.1994.346308.
[28] Wilhelms J, Gelder AV. Topological considerations in isosurface generation. Comput Graph 1990;24(5):79–86.
[29] Nielson G, Hamann B. The asymptotic decider : Resolving the ambiguity in marching cubes. In: Proc. of IEEE visualization 91. 1991, p. 83–91.
[30] Cignoni P, Ganovelli F, Montani C, Scopigno R. Reconstruction of topologically correct and adaptive trilinear isosurfaces. Comput Graph 2000;24(3):399–418.
[31] Lewiner T, Lopes H, Vieira AW, Tavares G. Efficient implementation of marching cubes' cases with topological guarantees. J Graph Tools 2003;8(2).
[32] Nielson G. On marching cubes. IEEE Trans Vis Comput Graphics 2003;9(3):283–97.
[33] Lopes A, Brodlie K. Improving the robustness and accuracy of the marching cubes algorithm for isosurfacing. IEEE Trans Vis Comput Graphics 2003;9(1):16–29.
[34] Ju T, Losasso F, Schaefer S, Warren J. Dual contouring of Hermite data. ACM Trans Graph 2002;21(3):339–46.
[35] Krishnan S, Manocha D, Varadhan G, Kim YJ. Feature-sensitive subdivision and isosurface reconstruction. In: Visualization conference, IEEE. Los Alamitos, CA, USA: IEEE Computer Society; 2003, p. 99–106. http://dx.doi.org/10.1109/VISUAL.2003.1250360.
[36] Varadhan G, Krishnan S, Sriram T, Manocha D. Topology preserving surface extraction using adaptive subdivision. In: Scopigno R, Zorin D, editors. Symposium on geometry processing. The Eurographics Association; 2004, p. 241–50. http://dx.doi.org/10.2312/SGP/SGP04/241-250.
[37] Andújar C, Brunet P, Ayala D. Topology-reducing surface simplification using a discrete solid representation. ACM Trans Graph 2002;21(2):88–105. http://dx.doi.org/10.1145/508357.508359, 10.1145/508357.508359.
[38] López J, Esteban A, Hernández J, Gómez P, Zamora R, Zanzi C, Faura F. A new isosurface extraction method on arbitrary grids. J Comput Phys 2021;444:110579.
[39] Nielson GM, Graf G, Holmes R, Huang A, Phielipp M. Shrouds: optimal separating surfaces for enumerated volumes. In: VisSym, vol. 3. 2003, p. 75–84.
[40] Coeurjolly D, Lachaud J-O, Gueth P. Digital surface regularization with guarantees. IEEE Trans Vis Comput Graphics 2021;27(6):2896–907.
[41] Shen T, Gao J, Yin K, Liu M-Y, Fidler S. Deep marching tetrahedra: a hybrid representation for high-resolution 3D shape synthesis. Adv Neural Inf Process Syst 2021;34.
[42] Wickramasinghe U, Remelli E, Knott G, Fua P. Voxel2Mesh: 3D mesh model generation from volumetric data. In: Martel AL, Abolmaesumi P, Stoyanov D, Mateus D, Zuluaga MA, Zhou SK, Racoceanu D, Joskowicz L, editors. Medical image computing and computer assisted intervention – MICCAI 2020. Cham: Springer International Publishing; 2020, p. 299–308.
[43] Liao Y, Donne S, Geiger A. Deep marching cubes: Learning explicit surface representations. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2018, p. 2916–25.
[44] Chryssolouris G, Mavrikios D, Papakostas N, Mourtzis D, Michalos G, Georgoulias K. Digital manufacturing: history, perspectives, and outlook. Proc Inst Mech Eng B 2009;223(5):451–62.
[45] Chen HH, Huang TS. A survey of construction and manipulation of octrees. Comput Vis Graph Image Process 1988;43(3):409–31.
[46] Brunet P, Navazo I. Solid representation and operation using extended octrees. ACM Trans Graph 1990;9(2):170–97.
[47] Coatrieux J-L, Barillot C. A survey of 3D display techniques to render medical data. In: 3D imaging in medicine. Springer; 1990, p. 175–95.
[48] Ohtake Y, Belyaev AG. Mesh optimization for polygonized isosurfaces. Comput Graph Forum 2001;20(3):368–76.
[49] Gibson SF. Constrained elastic surface nets: Generating smooth surfaces from binary segmented data. In: International conference on medical image computing and computer-assisted intervention. Springer; 1998, p. 888–98.
[50] Whitaker RT. Reducing aliasing artifacts in iso-surfaces of binary volumes. In: 2000 IEEE symposium on volume visualization (VV 2000). IEEE; 2000, p. 23–32.
[51] Chica A, Williams J, Andújar C, Brunet P, Navazo I, Rossignac J, Vinacua A. Pressing: Smooth isosurfaces with flats from binary grids. Comput Graph Forum 2008;27(1):36–46.
[52] Gao W, Zhang Y, Ramanujan D, Ramani K, Chen Y, Williams CB, Wang CC, Shin YC, Zhang S, Zavattieri PD. The status, challenges, and future of additive manufacturing in engineering. Comput Aided Des 2015;69:65–89.
[53] Wohlers T, Gornet T. History of additive manufacturing. Wohlers Rep 2014;24(2014):118.
[54] McMains S, Séquin C. A coherent sweep plane slicer for layered manufacturing. In: Proceedings of the fifth ACM symposium on solid modeling and applications. 1999, p. 285–95.
[55] Lefebvre S. IceSL: A GPU accelerated CSG modeler and slicer. In: 18th European forum on additive manufacturing. 2013.
[56] Comino M, Vinacua A, Carruesco A, Chica A, Brunet P. Sweep encoding: Serializing space subdivision schemes for optimal slicing. Comput Aided Des 2022;146:103–89. http://dx.doi.org/10.1016/j.cad.2021.103189.
[57] Hornus S, Kuipers T, Devillers O, Teillaud M, Martínez J, Glisse M, Lazard S, Lefebvre S. Variable-width contouring for additive manufacturing. ACM Trans Graph 2020;39(4). 131–1.
[58] Bedel A, Coudert-Osmont Y, Martínez J, Islam RI, Whitesides S, Lefebvre S. Closed space-filling curves with controlled orientation for 3D printing. 2021, URL https://hal.inria.fr/hal-03185200.
[59] Filoscia I, Alderighi T, Giorgi D, Malomo L, Callieri M, Cignoni P. Optimizing object decomposition to reduce visual artifacts in 3d printing. Comput Graph Forum 2020;39:423–34.
[60] Karasik E, Fattal R, Werman M. Object partitioning for support-free 3D-printing. Comput Graph Forum 2019;38:305–16.
[61] Tricard T, Claux F, Lefebvre S. Ribbed support vaults for 3D printing of hollowed objects. Comput Graph Forum 2020;39(1):147–59.
[62] Luongo A, Falster V, Doest MB, Ribo MM, Eiríksson ER, Pedersen DB, Frisvad JR. Microstructure control in 3D printing with digital light processing. Comput Graph Forum 2020;39(1):347–59.
[63] Efremov S, Martínez J, Lefebvre S. 3D periodic cellular materials with tailored symmetry and implicit grading. Comput Aided Des 2021;140:103086.
[64] Tricard T, Tavernier V, Zanni C, Martínez J, Hugron P-A, Neyret F, Lefebvre S. Freely orientable microstructures for designing deformable 3D prints. ACM Trans Graph 2020;39(6). 211–1.
[65] Coelho PG, Fernandes PR, Guedes JM, Rodrigues HC. A hierarchical model for concurrent material and topology optimisation of three-dimensional structures. Struct Multidiscip Optim 2008;35(2):107–15.
[66] Rittig T, Sumin D, Babaei V, Didyk P, Voloboy A, Wilkie A, Bickel B, Myszkowski K, Weyrich T, Křivánek J. Neural acceleration of scattering-aware color 3D printing. Comput Graph Forum 2021;40(2):205–19.
[67] Piovarci M, Foshey M, Babaei V, Rusinkiewicz S, Matusik W, Didyk P. Towards spatially varying gloss reproduction for 3D printing. ACM Trans Graph 2020;39(6).
[68] Nindel TK, Iser T, Rittig T, Wilkie A, Křivánek J. A gradient-based framework for 3D print appearance optimization. ACM Trans Graph 2021;40(4):1–15.
[69] Brunton A, Rmaileh LA. Displaced signed distance fields for additive manufacturing. ACM Trans Graph 2021;40(4):1–13.
[70] Peng H, Liu P, Lu L, Sharf A, Liu L, Lischinski D, Chen B. Fabricable unobtrusive 3D-QR-codes with directional light. Comput Graph Forum 2020;39(5):15–27.
[71] Tymms C, Wang S, Zorin D. Appearance-preserving tactile optimization. ACM Trans Graph 2020;39(6):1–16.
[72] Gillmann C, Saur D, Wischgoll T, Scheuermann G. Uncertainty-aware visualization in medical imaging - A survey. Comput Graph Forum 2021;40(3):665–89.
[73] Gillmann C, Smit NN, Gröller ME, Preim B, Vilanova A, Wischgoll T, Rhyne T. Ten open challenges in medical visualization. IEEE Comput Graph Appl 2021;41(5):7–15.
[74] Mambo S, Djouani K, Hamam Y, van Wyk B, Siarry P. A review on medical image registration techniques. Int J Comput Inf Eng 2018;12(1):48–55.
[75] Preim B, Botha CP. Visual computing for medicine, second edition: Theory, algorithms, and applications. 2nd ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.; 2013.