Solid Freeform Fabrication 2023: Proceedings of the 34th Annual International Solid Freeform Fabrication Symposium – An Additive Manufacturing Conference
Reviewed Paper

Level Set Grids for Hybrid Manufacturing

Liam White[*†], Bryan Quaife[†], Michael Borish[*], Cameron Adkins[*], Alex Roschli[*]

[*]Manufacturing Science Division, Oak Ridge National Laboratory, Knoxville, TN 37932
[†]Department of Scientific Computing, Florida State University, Tallahassee, FL 32301

Abstract

We propose a novel hybrid model, the Level Set Grid, to facilitate parallel additive and subtractive processes in hybrid manufacturing. The Level Set Grid combines the strengths of explicit and implicit representations, offering precise modeling of evolving geometries and fast and efficient collision detection. This research focuses on integrating Level Set Grids into the additive slicing and subtractive pathing generation processes, laying the groundwork for future advancements in the parallelization of hybrid manufacturing.

Keywords: hybrid manufacturing, additive manufacturing, subtractive manufacturing, OpenVDB, sparse volumetric grid, voxel grid, level set function, signed distance function, marching squares

Introduction

Hybrid manufacturing, the integration of additive and subtractive manufacturing techniques within a single machine system, holds the potential to revolutionize the manufacturing industry [1]. By combining the strengths of both methods, it transcends the limitations of either method when used in isolation. However, current hybrid systems execute additive and subtractive processes sequentially, resulting in a significant inefficiency. To fully realize the potential of hybrid manufacturing, these processes must be executed in parallel.

The parallelization of additive and subtractive processes necessitates two crucial capabilities. First, the ability to represent the object as it is being additively constructed is essential, as only surfaces that have already been fabricated can be machined. As a result, subtractive pathing must be generated based on partial representations of the object. Second, fast and efficient collision detection is indispensable to ensure that the tools used in each process do not collide.

To fulfill these requirements, a model representation that is conducive to both additive and subtractive manufacturing processes is required. Model representations in manufacturing can be broadly categorized into two types based on how they represent 3D objects: explicit and implicit representations. Explicit representations, such as meshes, point clouds, and voxel grids, directly define the geometry of the object. They are straightforward, compatible with numerous software tools, and can represent a diverse range of shapes and topologies. Conversely, implicit representations including solid models, parametric models, and functional representations, define the geometry of the object indirectly through mathematical functions or rules. They offer precision,

support complex geometric operations, can be used to calculate physical properties such as the surface curvature and normal, and are robust to certain types of transformations.

We propose the use of a novel hybrid representation, the Level Set Grid, which combines benefits of both explicit and implicit representations into a single model. An object's geometry is represented using a level set function, typically the signed distance function, that implicitly defines its shape. This function assigns a scalar value to each point in space to indicate the distance from that point to the object's surface: negative values for points inside the object, positive values for points outside, and zero for points on the surface. This implicit representation enables precise and robust modeling of complex geometries and topologies. To further enhance the representation, the signed distance function is evaluated at specific spatial coordinates around the object's surface, a region referred to as the 'narrow band'. The resulting scalar values are stored in a sparse volumetric grid that resembles a voxel grid. This grid-based model serves as an explicit representation, where each voxel contains the scalar value derived from the signed distance function. By unifying the accuracy and robustness of implicit representations with the simplicity and compatibility of explicit representations, this approach offers a comprehensive solution for geometric modeling.

The use of Level Set Grids in hybrid manufacturing toolpath planning is particularly advantageous, not only for collision avoidance but also for storing a partial representation of an object as it is being built. By tracking which voxels have been fabricated and which have not, the system flags a voxel as built when the additive machine traverses it. Using this methodology, a filter is applied at any instance in time to hide voxels that have not been built, leaving a partial representation of the object for which subtractive pathing can be generated. Meanwhile, the scalar values stored in the grid provide a signed distance field around the object, enabling fast identification of potential collisions. If a potential toolpath intersects with a voxel containing a negative value (indicating it is inside the object), a collision is identified, and the system promptly generates an alternative toolpath, averting the collision altogether. This real-time responsiveness ensures uninterrupted parallel execution of additive and subtractive processes, enhancing the overall efficiency and safety of hybrid manufacturing.

To demonstrate the viability of Level Set Grids for hybrid manufacturing, this paper focuses on the integration of Level Set Grids into the traditional additive slicing process and the generation of subtractive pathing. This lays the foundation for future research into the parallelization of these processes, which is the key to unlocking the full potential of hybrid manufacturing.

Background and Related Work

The field of manufacturing has predominantly relied on tessellated meshes for representation, largely due to their simplicity and the standardization of the STL file format. However, the tessellated mesh is not without its limitations. A significant challenge is accurately representing curved surfaces, which are often approximated with triangular facets. This necessitates tessellating such objects into a multitude of small facets, leading to large file sizes and a reduction in geometric/topological accuracy [2].

Recognizing the need for an alternative, Guduri et al. proposed an implicit representation known as Constructive Solid Geometry (CSG) [3, 4]. This approach allows for the direct extraction of slice contours from the CSG, bypassing the need for an explicit tessellated mesh and thereby preserving the model's geometric and topological features. This concept of directly slicing an implicit representation was later formalized by Jamieson and Hacker, a process now widely recognized as Direct Slicing [5]. The application of implicit representations for slicing has since gained widespread acceptance, particularly for slicing complex geometries characterized by high curvature surfaces. A notable example is the slicing of triply periodic minimal surfaces, as demonstrated by Ding et al. and Rastegarzeadeh and Huang [6, 7].

In parallel with the development of implicit representations, there has been significant progress in the use of voxel models for layered manufacturing. Chandru et al. initially demonstrated that employing dense voxel grids that, while accurate, was computationally intensive and inefficient [8]. However, the field has advanced substantially, leading to more efficient and accurate methods for manufacturing complex objects. A notable contribution came from Ghadai et al., who introduced a direct method for additive manufacturing using multi-level voxelized models [9]. This pioneering technique addressed the challenges associated with the slicing operation for conventional CAD models. It facilitates the direct printing of thresholded voxel models, that are typically obtained from CT or MRI scans, thereby enabling the fabrication of physical 3D representations of medical data.

In the domain of subtractive manufacturing, significant strides have been made in the application of voxel-based methods. Lynn et al. introduced a voxel-based computer-aided manufacturing (CAM) system that facilitates direct digital subtractive manufacturing (DDSM) of mechanisms without the need for assembly [10]. This voxelized CAM system optimizes the process by automatically generating toolpaths, based on an analysis of material that can be removed to achieve a specific clearance in the assembled state. Complementing this, Li et al. proposed a voxel model-based process planning method for five-axis machining, demonstrating its effectiveness particularly for objects with intricate features or weak structures [11]. This approach leverages voxel representation to identify a sequence of intermediate machining layers, ensuring uninterrupted tool access and circumventing problems related to self-intersection and layer redundancy.

Further enhancing voxel representations, Joy et al. introduced a Frame-Sliced Voxel representation (FSV-rep) to depict the boundary of the workpiece volume [12]. The FSV-rep provides a boundary representation of the workpiece model with a level of accuracy that significantly surpasses that of a basic voxel with equivalent grid resolution and model size.

In a significant development, Brunton and Rmaileh introduced displaced signed distance fields, a hybrid implicit voxel representation, designed to accurately, efficiently, and robustly 3D-print intricate and smoothly curved surfaces at the device's native resolution [13]. This implicit approach enables the enhancement of low-polygon meshes with compact meso-scale topographic data, such as displacement maps, and the creation of curved polygons, all while utilizing efficient, streaming-compatible, discrete voxel-wise algorithms. This methodology substantially advances the application of alternative representations, combining the benefits of implicit representations with the efficiency and flexibility of explicit voxel-based methods.

Working towards the parallelization of hybrid manufacturing, this paper builds upon the foundational work of Brunton and Rmaileh [13]. It showcases the versatility of such a hybrid representation, demonstrating its applicability across a range of manufacturing processes, including additive, subtractive, and hybrid methodologies.

Grid Construction

To construct and store the Level Set Grid, we use OpenVDB, an open-source C++ library developed by Ken Museth at DreamWorks Animation [14]. The library allows for the efficient storage, manipulation, and operation of volumetric data. The grid is generated using OpenVDB's 'meshToLevelSet' function which takes as inputs a tessellated mesh, the desired voxel size, and the width of the narrow band (in voxels).

The process begins by defining a 3D grid of voxels in the space occupied by the mesh. The generated grid is always larger than the mesh to ensure that all parts of the mesh are captured in the voxelization process. An example of this process can be seen in *Figure 1* which shows the triangulated mesh of an object and its voxelized representation.

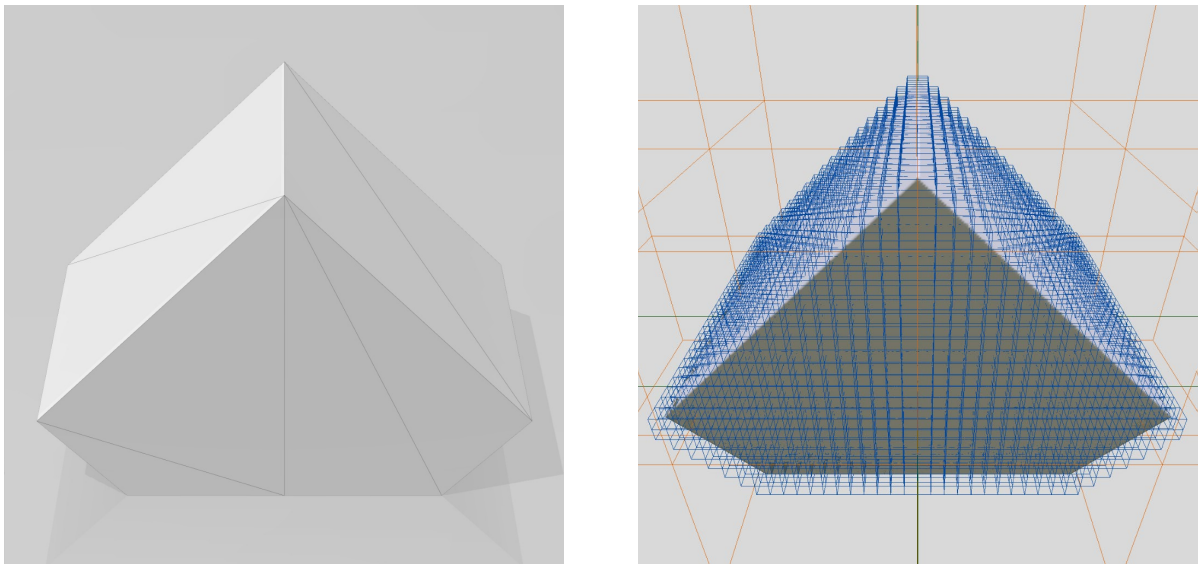

Figure 1. Visualization of the triangulated mesh of an object (left) alongside its corresponding voxelized representation (right) constructed using OpenVDB.

Once the grid is generated, each voxel contained within the narrow band (i.e., within a user-specified distance of the mesh surface) is labeled as inside, outside, or on the mesh surface. There are several methods to perform the labeling operation, but one common method is ray casting. This involves casting a ray from the center of the voxel in an arbitrary direction and determining how many times it intersects the mesh. If it intersects an odd number of times, the voxel is inside the mesh; if it intersects an even number of times, the voxel is outside or on the mesh surface. A 2D example of the ray cast labeling process is illustrated in *Figure 2*.

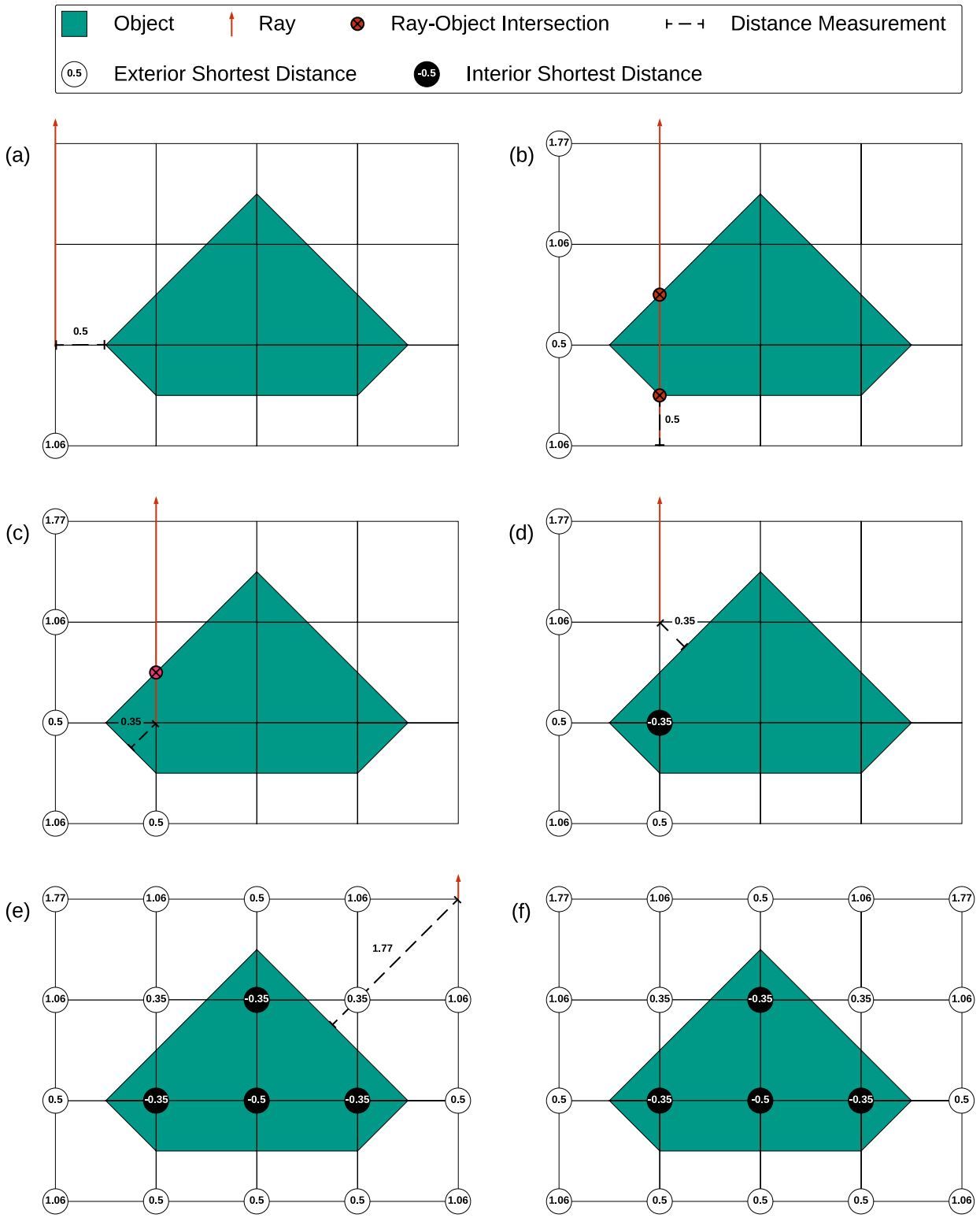

Figure 2. Detailed process of the signed distance field generation for a 2D object. (a-c) Rays cast from voxels intersect the object boundary zero, two, and one times, dictating the sign of the corresponding distance value. (d & e) Iterative steps showcasing the assignment of signed distances to all voxels. (f) The completed signed distance field of the object.

For each voxel in the narrow band, the shortest distance to the surface of the mesh is computed and stored within the voxel as a signed value. If the voxel is labeled as being inside the mesh, the distance is stored as a negative value; otherwise, the voxel is outside or on the mesh surface, and the stored value is non-negative. The distance field is computed using a fast-marching method, an efficient algorithm for computing distances in a grid. An overview of this process in 2D is demonstrated in *Figure 2*.

Once the signed distance field has been generated, it is then converted into a level set. This sometimes requires refining the grid to produce a smooth signed distance field that is continuous and has a gradient whose magnitude is one everywhere. This is done using a process called redistancing or reinitialization, which involves solving a partial differential equation known as the Eikonal equation on the grid.

It is important to note that voxelization is a discretization process and can introduce some approximation errors. The quality of the voxelization can be improved by using a smaller voxel size, but this comes at the cost of increased memory usage and computational cost.

Slicing

Slicing is a pivotal step in additive manufacturing, transforming a 3D model into a sequence of print-ready layers. This procedure has two core elements: creating polygonal geometries for each layer and generating paths to fill these geometries. When working with Level Set Grids, polygonal layer geometries are produced in two stages: Cross-sectioning and Contour Extraction. Cross-sectioning uses OpenVDB's ‘clip’ function to obtain a planar 2D grid of voxels from the 3D grid. Contour Extraction employs a modified marching squares algorithm to isolate contours from the resulting cross-sections. The extracted polygonal geometries are integrated into standard slicing software to generate additive paths. This section outlines the steps required to produce polygonal layer geometries from Level Set Grids.

Cross-sectioning

To obtain a planar cross-section of a Level Set Grid, we use OpenVDB’s ‘clip’ function. This function takes an axis-aligned bounding box and the target grid as inputs and outputs a new grid featuring only the voxels within the designated bounding box. The bounding box itself is defined by its minimum and maximum corner coordinates.

To ensure a comprehensive planar cross-section, the bounding box is initially defined to encompass the entire grid. The corner coordinates are then adjusted to align with the target clipping plane, and the grid is clipped accordingly.

The clipping process yields a 2D grid of voxels, each retaining its associated signed distance value. Polygonal contours are subsequently extracted from the resulting cross-sections.

Contour Extraction

To extract the contours contained within a cross-section, a modified version of the marching squares algorithm is employed. Unlike the standard version, this modified algorithm produces consistently oriented contours, enabling a straightforward conversion to standard polygons with holes. The implementation follows a similar approach by Inui et al. [15].

The process begins with a linear scan of 2D grid cells. A cell is defined as a 2x2 block of nodes. For each cell, a binary index is generated through a clockwise traversal of its nodes. Nodes with positive signed distances contribute a '0' to the index, while those with negative signed distances contribute a '1'. This binary index is used to reference a pre-compiled lookup table containing 16 entries that specify the edges needed to represent each cell; this table is detailed in Appendix A. Throughout the scan, a 2D bit vector records which cells have been visited to prevent redundancy. A visual summary of this process is provided in *Figure 3a & 3b*.

When an edge is identified, bilinear interpolation determines the position of the segment's endpoints along the cell walls, as illustrated in *Figure 3c*. An important constraint is imposed during the segment generation process: the interior of the object must consistently reside to the right of the segment when observed from the starting endpoint. This ensures that the winding of all contours is clockwise. Although this approach does not inherently produce polygons with counter-clockwise winding holes, converting them to meet the appropriate filling rules is an accessible task.

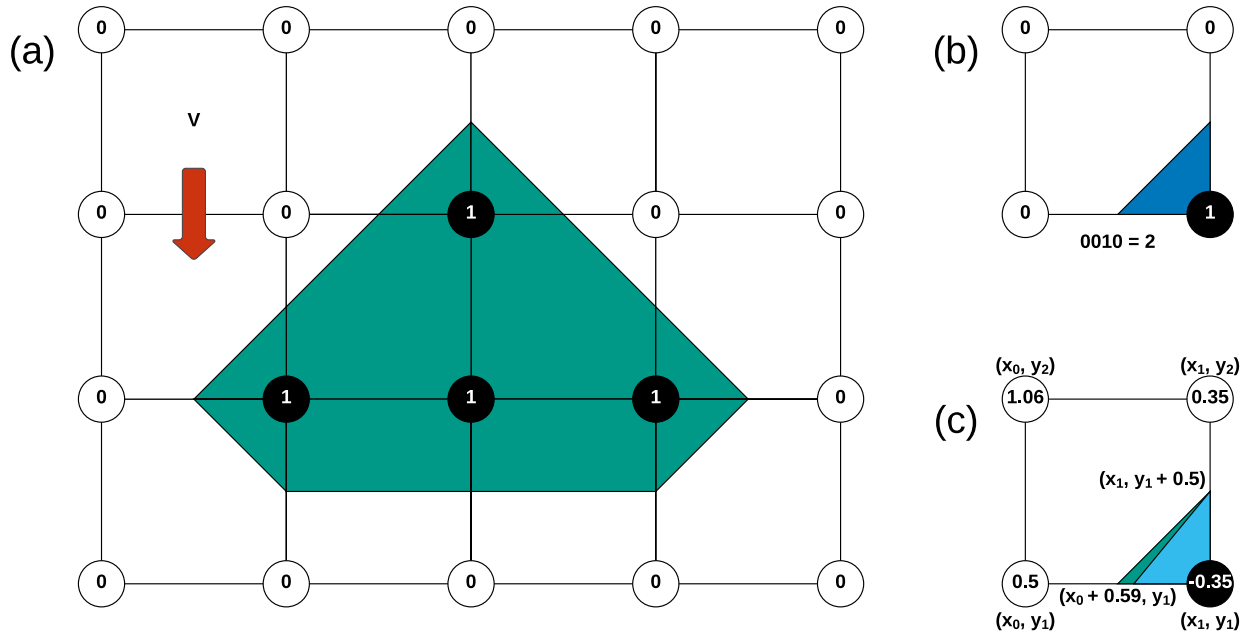

Figure 3. Visual representation of the contour extraction process from a Level Set Grid cross-section. (a) Linear scan (red arrow) of 2D grid cells with generated binary indices. (b) Reference to the pre-compiled lookup table for edge representation based on binary indices. (c) Determination of the segment's endpoints using bilinear interpolation (cyan triangle).

Once the first segment of a new contour is generated, a walk of the grid is initiated, guided by the orientation of the previously generated segment. Each segment's endpoint within the walk is strategically positioned on the adjacent wall of the succeeding cell that is part of the contour.

The next cell in the walk is selected based on this constraint and the segment calculation is repeated. As each segment is generated, its endpoint is added to an accumulating list designated for the current contour, and the corresponding cell is flagged as 'visited'. This procedure continues until a cell marked as visited is revisited, signaling the completion of the closed contour. At this juncture, the resultant contour is stored, and the algorithm returns to the linear scan to identify potential additional contours for traversal. To preclude the generation of redundant contours within each composite grid component, any components previously visited are deliberately skipped. An overview of this process is illustrated in *Figure 4*.

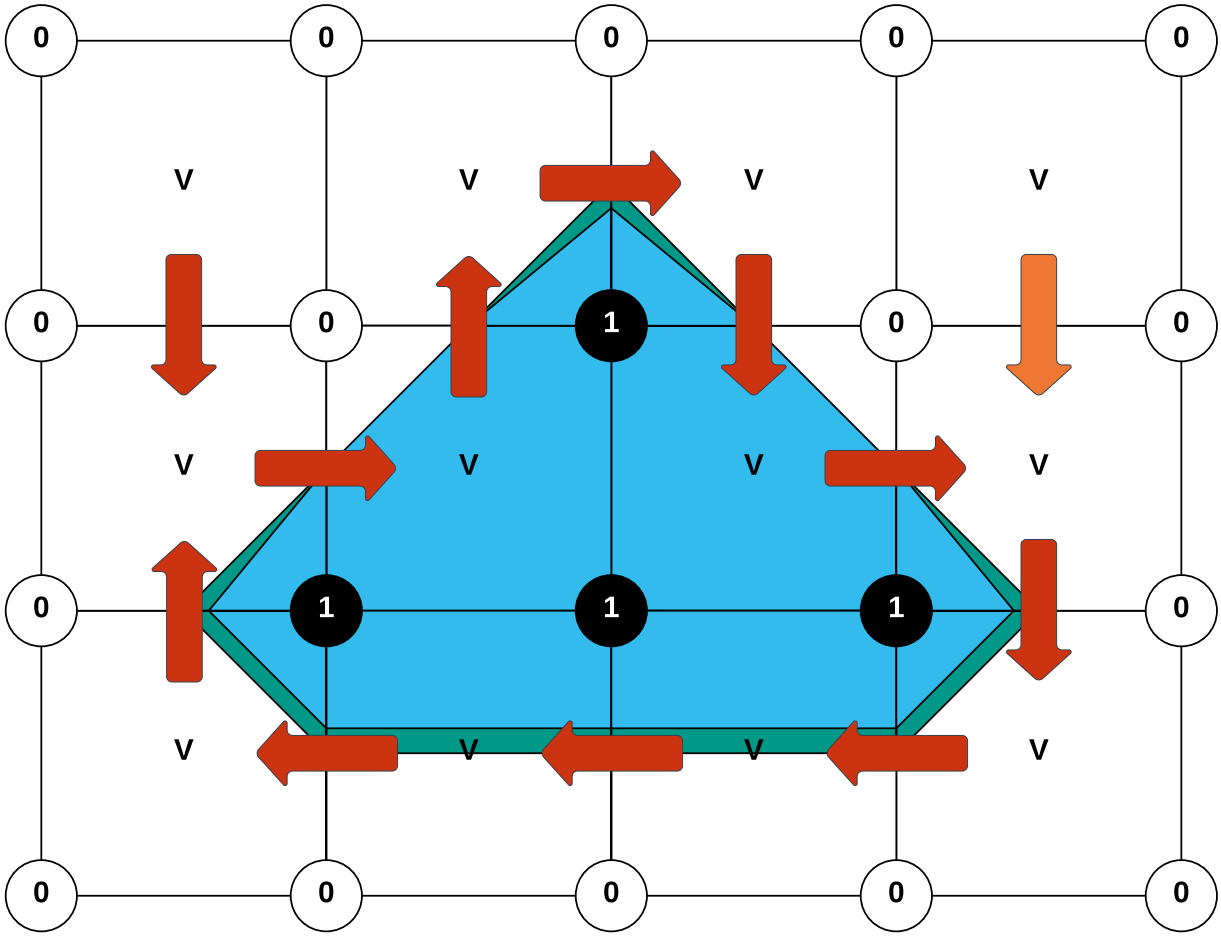

Figure 4. Schematic overview of the grid walk procedure for contour extraction. Beginning with the first segment, the walk (red arrows) progresses through adjacent grid cells, accumulating segment endpoints for the current contour. The process is terminated upon revisiting a 'visited' cell ('V'), indicating a completed contour. The algorithm then proceeds to identify potential additional contours (orange arrow), while avoiding previously visited grid components.

For special cases where multiple edges are detected within a single cell, a queuing system is utilized. Should such an event occur during a grid walk, the segment not associated with the active contour is enqueued into a list designated for future contour extraction. Upon the completion of a given contour, the queue is examined to ascertain if additional contours can be extracted. If so, the contour extraction process seamlessly resumes, ensuring comprehensive coverage.

Surface Pathing

The slicing algorithm outlined in the preceding section serves as a foundation for generating subtractive surface pathing. This process utilizes the contours derived from slicing to construct a path for the subtractive manufacturing tool, guiding it seamlessly across the object's surface.

Surface Normal Computation

The process begins by iterating through the points encompassed within each contour. At every point, the surface normal is computed by evaluating the gradient of the signed distance field at the corresponding location in the Level Set Grid. This gradient calculation employs the method of central differences to yield a vector pointing in the direction of maximum field increase, thereby identifying the surface normal at the given point. A 2D graphical representation of the surface normal computation is depicted in *Figure 5a*.

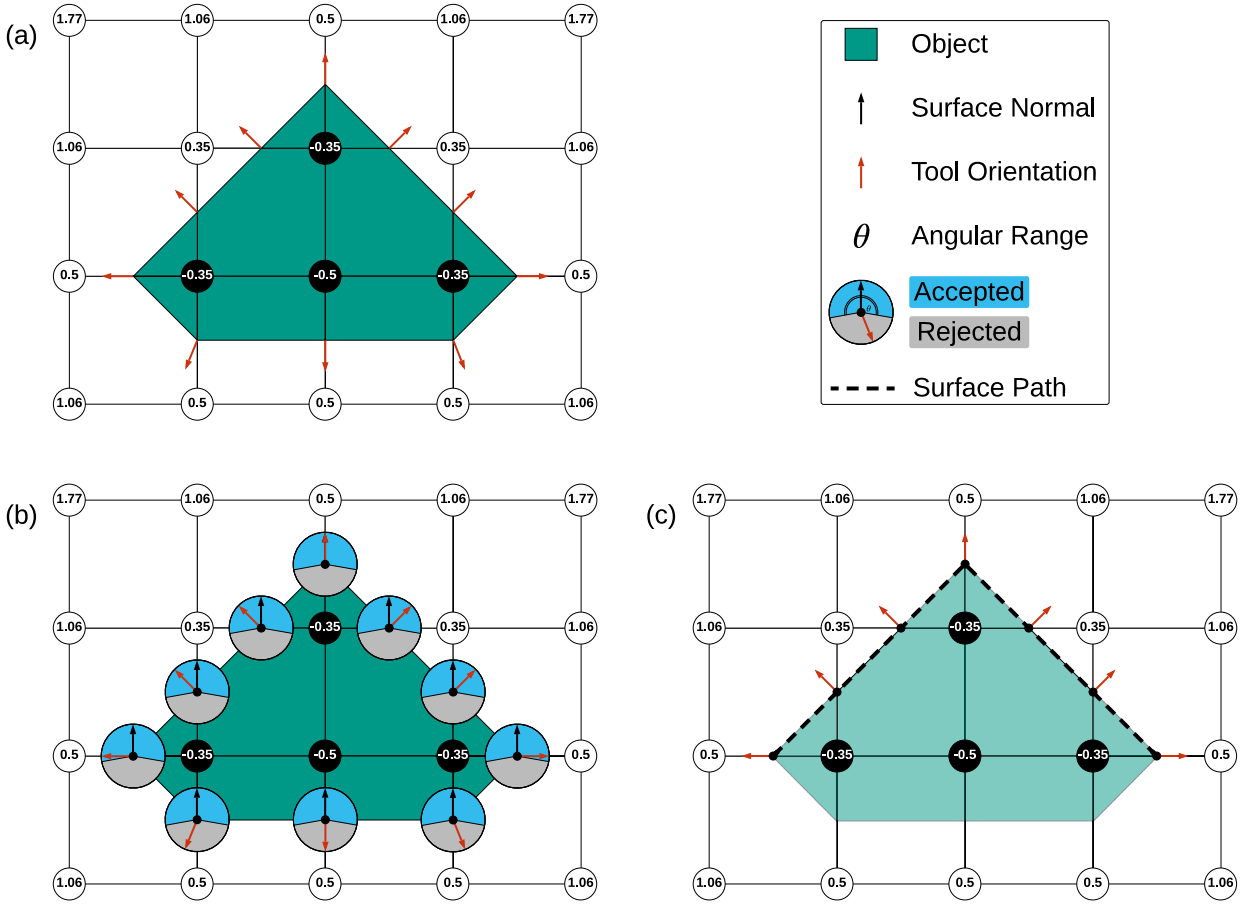

Figure 5. Workflow of surface pathing generation. (a) Graphical depiction of the surface normal computation using the gradient of the signed distance field. (b) Illustration of the orientation constraint evaluation, comparing the computed surface normal with the predefined tool orientation vector and acceptable angular range. (c) Resultant surface pathing based on the computed surface normal vectors and orientation constraints.

Orientation Constraint

Once the surface normal is calculated, it is evaluated in accordance with a predefined tool orientation vector $\left(O_t = (O_x, O_y, O_z)\right)$ and an acceptable angular range (θ). The vector serves as a standard reference direction, indicating the preferred orientation of the manufacturing tool during operation, while the angular range defines the maximum effective tilt. If the angle between the computed normal and the tool orientation vector falls within the defined range, the point, and its associated surface normal are incorporated into the surface path. This process is illustrated in 2D in *Figure 5b*.

Iterative Path Generation

The surface normal computation and orientation constraint are applied iteratively for each point in every contour, culminating in surface pathing that adheres to the object's contours while conforming to the constraints dictated by the manufacturing process. The resultant surface pathing serves as a route for the subtractive manufacturing tool, ensuring both accurate and efficient material removal (*Figure 5c*).

It is important to note that the fidelity of the generated surface pathing, and by extension the surface quality of the final manufactured object, relies heavily on the accuracy of the slicing process. This, in turn, is governed by the resolution of the Level Set Grid. Given this interdependence, the selection of an appropriate grid resolution is paramount for achieving the desired surface quality.

Results and Discussion

To authenticate the potential of Level Set Grids for hybrid manufacturing, a series of tests are administered, focusing on an object characterized by intricate high-curvature features. This object, measuring approximately 48mm x 48mm x 23mm, is converted from its original STL mesh representation into a Level Set Grid using the prescribed grid construction method. The Level Set Grid is constructed with a voxel size of 0.2mm and a narrow band width of 1 voxel. An illustration of the object's STL mesh representation and Level Set Grid representation can be seen in *Figure 6*.

Slicing Results

For the slicing experiment, planar cross-sections of the Level Set Grid are extracted at intervals of every 5 voxels (equivalent to 1cm) along the z-axis. Post extraction, the polygonal contours of each cross-section are isolated using the contour extraction procedure. As evident from *Figure 7*, the contours maintain a high fidelity to the object's geometry at each respective slice. This highlights the Level Set Grid's capability to meticulously capture the object's complex curvature.

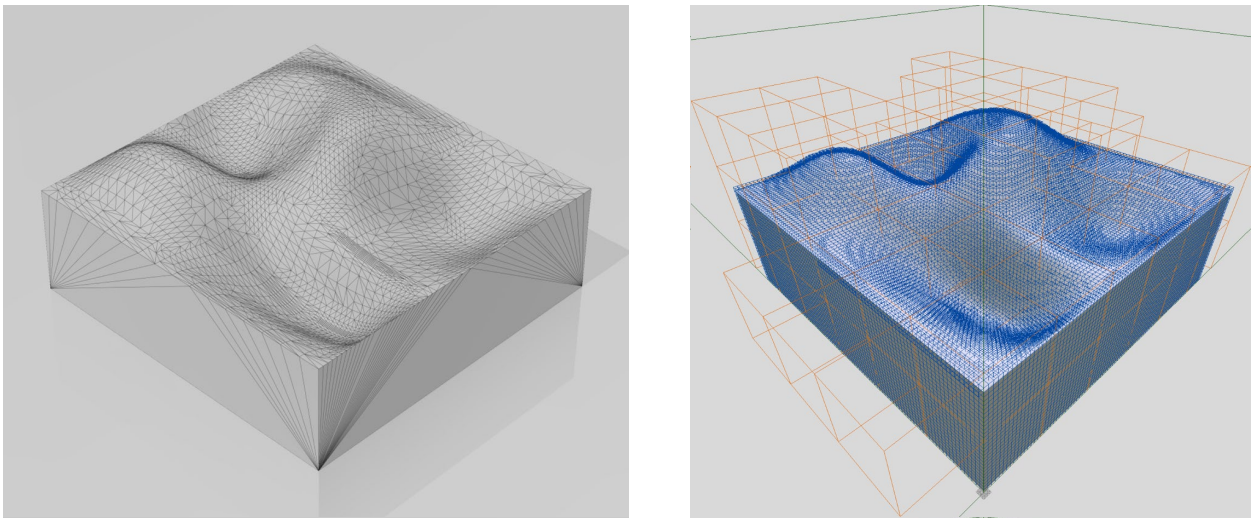

Figure 6. Visualization of the triangulated mesh representation of the object (left) alongside its corresponding Level Set Grid representation (right).

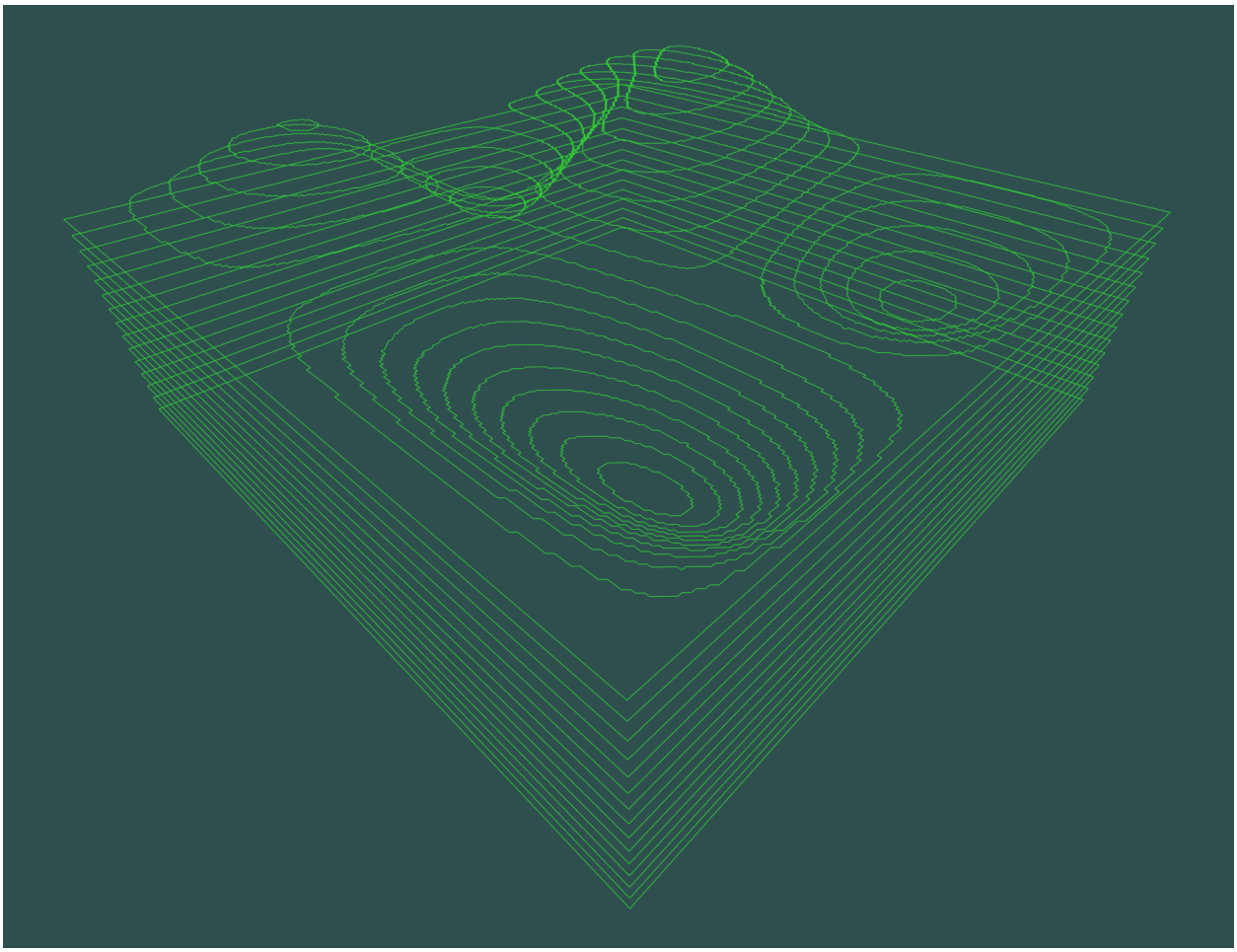

Figure 7. Extracted polygonal layer contours from the object's Level Set Grid, obtained by slicing the grid every 5 voxels (equivalent to 1cm) along the z-axis.

Surface Pathing Results

To showcase the process of generating surface pathing, the object's upward-facing surface is selected for subtracting toolpath generation. A predetermined tool orientation vector of $O_t = (0, 0, 1)$ is defined, with an angular range fixed at $\theta = 89°$. Sequentially, polygonal contours are extracted at every 5 voxels (equivalent to 1cm) along the y-axis (as shown in *Figure 8*). The gradient of the signed distance field at each point's location in the Level Set Grid is used to compute the corresponding surface normal (*Figure 9*).

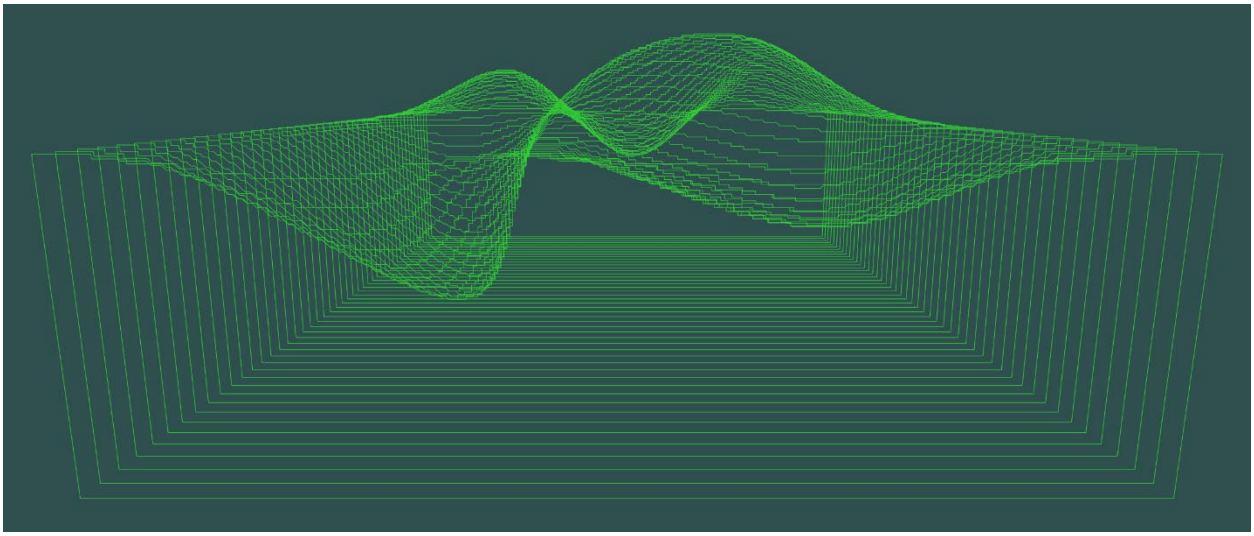

Figure 8. Extracted polygonal layer contours from the object's Level Set Grid, obtained by slicing the grid every 5 voxels (equivalent to 1cm) along the y-axis.

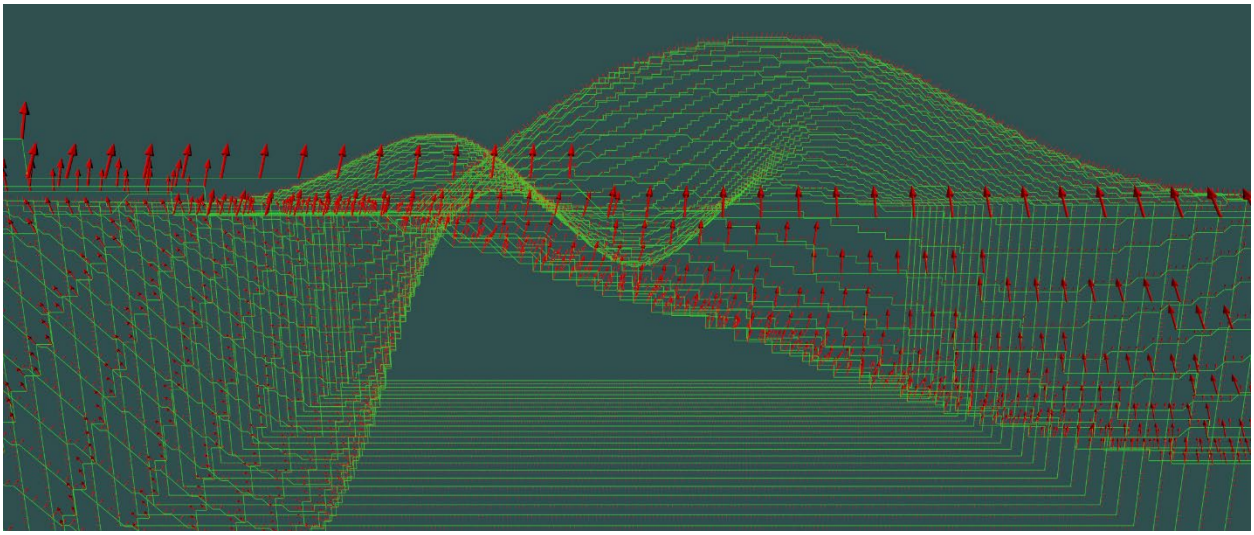

Figure 9. Computed surface normal vectors (red unit vectors) for each point in the contour, based on the gradient of the signed distance field in the object's Level Set Grid.

Each contour's points along with their corresponding surface normal vectors are subjected to a comparative assessment against the predefined tool orientation vector (O_t) and angular range (θ) to determine their inclusion in the surface pathing. As demonstrated in *Figure 10* the extracted

surface pathing aligns seamlessly with the object's upward-facing surface, accurately reflecting its curvature.

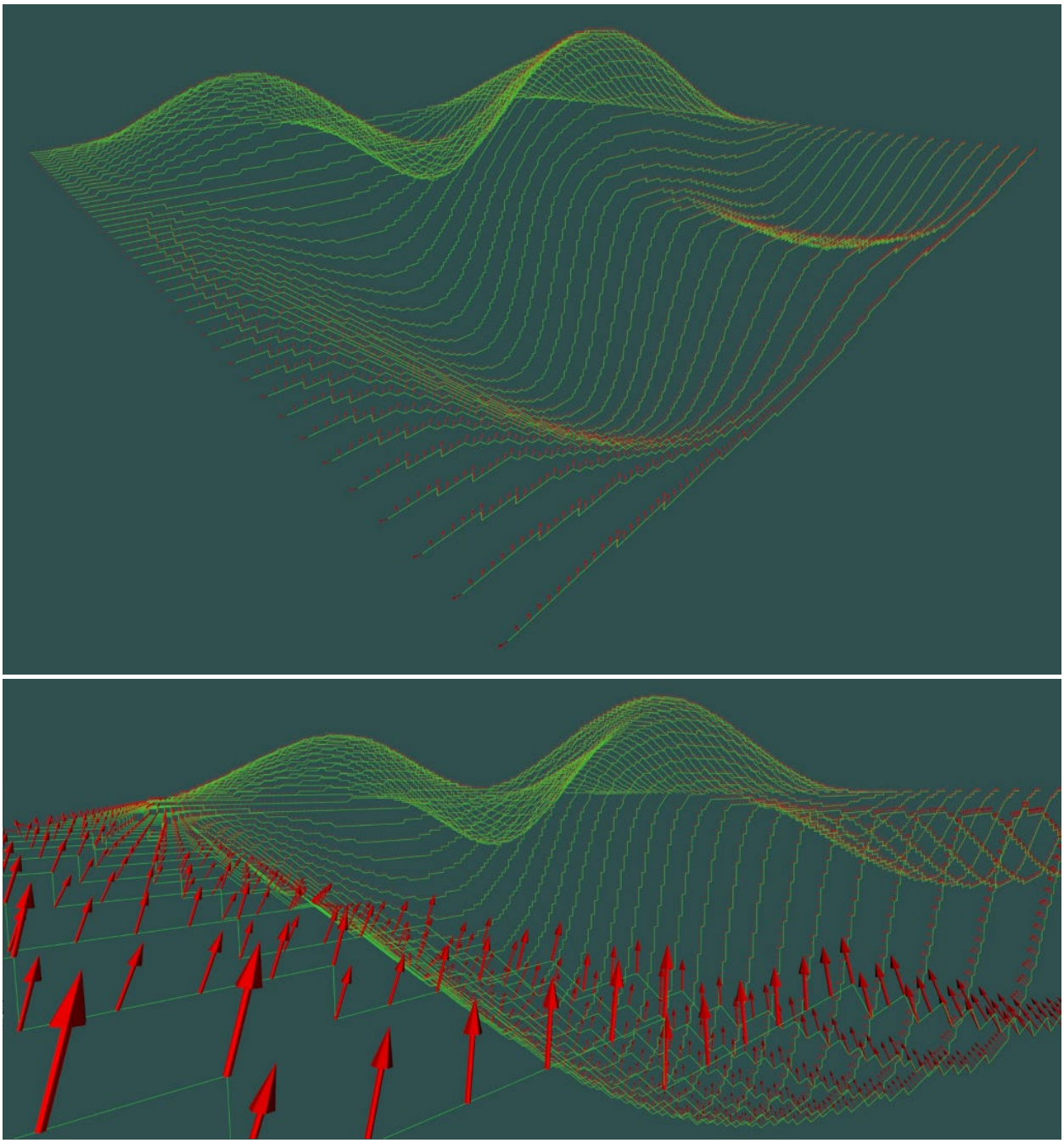

Figure 10. Results showcasing the generated surface pathing composed of points with surface normal vectors within the acceptable angular range when compared to the tool orientation vector.

A key insight from *Figure 10* highlights that the quality of surface pathing is closely tied to the accuracy of the slicing process, which is determined by the resolution of the Level Set Grid. Thus, an optimal grid resolution is pivotal for achieving the best surface quality in the final manufactured object. To ensure the best realized pathing resolution, the voxel size of the Level Set Grid should be smaller than the positional accuracy of the manufacturing system. If achieving this

grid resolution is challenging due to computational constraints, polyline simplification can also be used to enhance surface pathing accuracy.

Conclusion

This research introduced the Level Set Grid, an innovative hybrid model representation, showcasing its potential in advancing the parallelization of additive and subtractive processes in hybrid manufacturing. The synergistic combination of explicit and implicit representations within the Level Set Grid promises precise geometric modeling of evolving structures, coupled with rapid and efficient collision detection—a vital aspect for the parallel enactment of manufacturing operations.

Our findings underline the successful integration of Level Set Grids into the conventional additive slicing process and the generation of subtractive pathing. The derived results signify that the Level Set Grid can faithfully represent an object's geometry during the slicing process and can adeptly construct surface pathing for subtractive manufacturing.

Although the investigations presented in this manuscript mark significant progress towards achieving hybrid manufacturing parallelization, this is merely the commencement. Anticipated future endeavors will emphasize further integration of the Level Set Grid into the manufacturing workflow, performance optimization, and exploring its applicability in diverse scenarios.

Acknowledgements

This material is based upon work supported by the U.S. Department of Energy, Office of Energy Efficiency and Renewable Energy, Office of Advanced Manufacturing, under contract number DE-AC05-00OR22725.

References

[1] K. B. Fillingim and T. Feldhausen, "Operator 4.0 for Hybrid Manufacturing," *Proceedings of the Design Society,* vol. 3, pp. 2835-2844, 2023.

[2] V. Kumar and D. Dutta, "An Assessment of Data Formats for Layered Manufacturing," *Advances in Engineering Science,* vol. 28, no. 3, pp. 151-164, 1997.

[3] S. Guduri, R. H. Crawford and J. J. Beaman, "A Method to Generate Exact Contour Files for Solid Freeform Fabrication," in *Solid Freeform Fabrication Symposium*, Austin, 1992.

[4] S. Guduri, R. H. Crawford and J. J. Beaman, "Direct Generation of Contour Files from Constructive Solid Geometry Representations," in *Solid Freeform Fabrication Symposium*, Austin, 1993.

[5] R. Jamieson and H. Hacker, "Direct Slicing of CAD Models for Rapid Prototyping," *Rapid Prototyping Journal,* vol. 1, no. 2, pp. 4-12, 1995.

[6] J. Ding, Q. Zou, S. Qu, P. Bartolo, X. Song and C. C. L. Wang, "STL-Free Design and Manufacturing Paradigm for High-Precision Powder Bed Fusion," *CIRP Annals,* vol. 70, no. 1, pp. 167-170, 2021.

[7] S. Rastegarzadeh and J. Huang, "Novel STL-Free Design Paradigm for High-Resolution Multi-Scale Architected Cellular Materials in Additive Manufacturing," in *Manufacturing Science and Engineering Conference*, New Brunswick, 2023.

[8] V. Chandru, S. Manohar and C. E. Prakash, "Voxel-Based Modeling for Layered Manufacturing," *IEEE Computer Graphics and Applications,* vol. 15, no. 6, pp. 42-47, 1995.

[9] S. Ghadai, A. Jignasu and A. Krishnamurthy, "Direct 3D Printing of Multi-Level Voxel Models," *Additive Manufacturing,* vol. 40, p. 101929, 2021.

[10] R. Lynn, M. Dinar, N. Huang, J. Collins, J. Yu, C. Greer, T. Tucker and T. Kurfess, "Direct Digital Subtractive Manufacturing of a functional Assembly Using Voxel-Based Models," *Journal of Manufacturing Science and Engineering,* vol. 140, no. 021006, 2017.

[11] Y. Li, K. Tang and L. Zeng, "A Voxel Model-Based Process-Planning Method for Five-Axis Machining of Complicated Parts," *Journal of Computing and Information Science in Engineering,* vol. 20, no. 041012, 2020.

[12] J. Joy and H.-Y. Feng, "Frame-Sliced Voxel Representation: An Accurate and Memory-Efficient Modeling Method for Workpiece Geometry in Machining Simulation," *Computer-Aided Design,* vol. 88, pp. 1-13, 2017.

[13] A. Brunton and L. A. Rmaileh, "Displaced Signed Distance Fields for Additive Manufacturing," *ACM Transactions on Graphics,* vol. 40, no. 4, pp. 1-13, 2021.

[14] K. Museth, "VDB: High-Resolution Sparse Volumes with Dynamic Topology," *ACM Transactions on Graphics,* vol. 32, no. 3, pp. 1-22, 2013.

[15] M. Inui, M. Kawano, I. Watanabe and N. Umezu, "Improved Algorithm to Trace Boundary Curves on Two-Dimensional Square Meshes," *International Journal of Automation Technology,* vol. 14, no. 5, pp. 816-823, 2020.

Appendix A. Marching Squares Cases

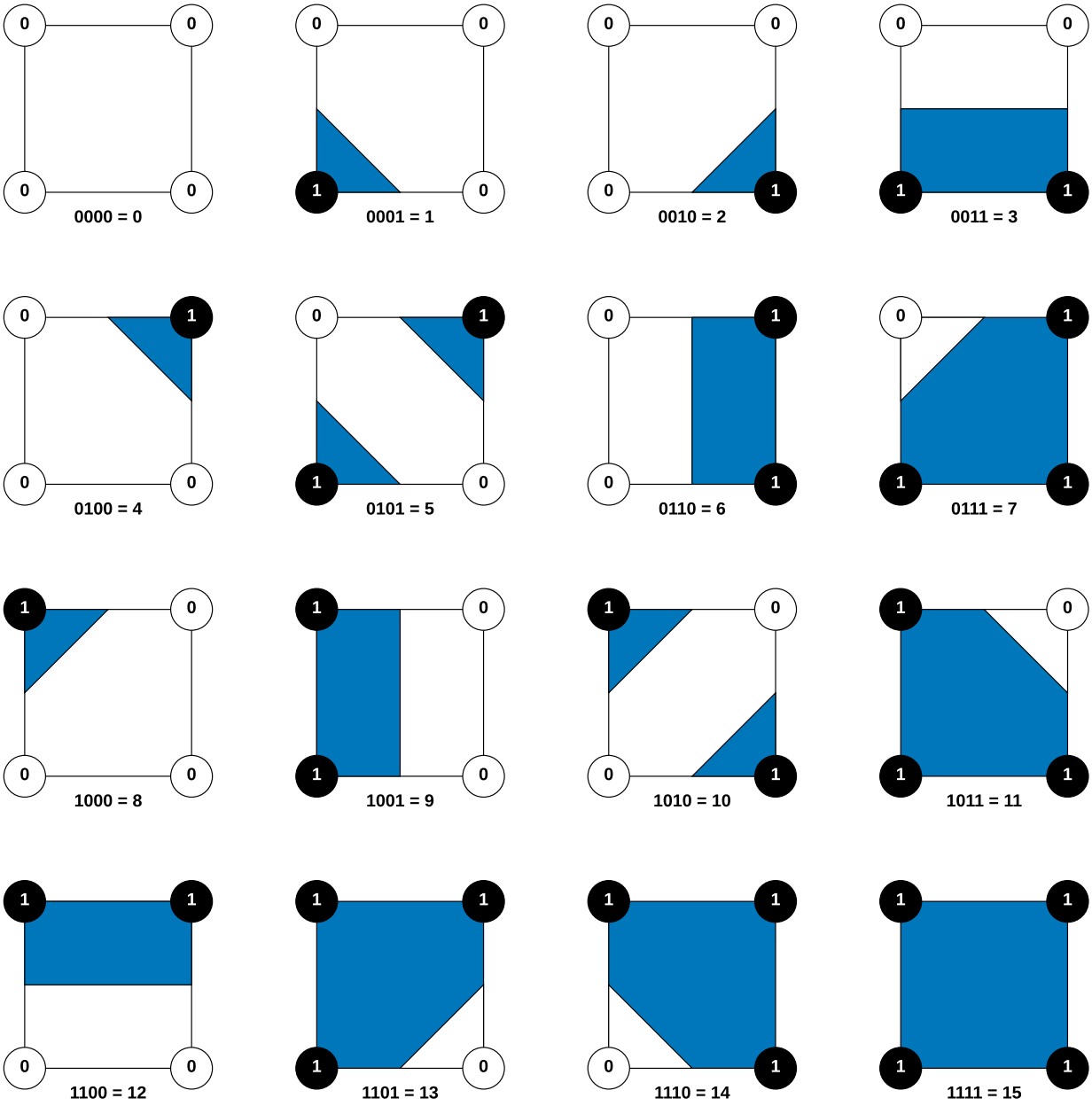

Figure 11. Marching squares lookup table. The binary index of each cell is determined by collecting in a clockwise direction starting from the upper left node. Blue denotes the interior of the object while white denotes the exterior.