The Use of Conformal Voxels for Consistent Extractions from Multiple Level-Set Fields

Max O. Bloomfield[1], David F. Richards[2], and Timothy S. Cale[1]

[1] Rensselaer Polytechnic Institute, Dept. of Chemical Engineering, Troy, NY 12180-3590
[2] Synopsys Inc., Mountain View, CA 94043

Abstract. We present and analyze in 2D an algorithm for extracting self-consistent sets of boundary representations of interfaces from level set representations of many phase systems. The conformal voxel algorithm which is presented requires the use of a mesh generator and is found to be robust and effective in producing the extractions in the presence of higher order junctions.

1 Introduction

The level-set method [1] is used in a wide range of simulations in which evolving geometries are important. This method, in which the interface between two phases is represented implicitly as a contour (usually the zero contour) in a scalar field, has many advantages over explicit methods. The most notable advantage is that the geometric and topological complexities associated with updating explicit representations of interfaces are avoided, as level-set evolution is just a matter of solving an equation on the scalar field [1]. If and when an explicit representation is needed in order to perform part of the simulation, the interface can be extracted as a level of the scalar field (usually the zero level). When the geometry contains more than two phases that need to be tracked, such as for thin film grain structures simulations, the method extends quite naturally through the use of multiple scalar fields, identifying each phase with one field [2]. These fields can then be evolved independently using the same level-set equation and be periodically reconciled as shown by Merriman *et al.* [2]. However, the multiple level-set approach suffers from one setback that the standard level-set method does not: Recovering an explicit geometry by extracting the contours is not trivial because each interface is represented multiple times. At points near triple lines and higher order junctions, the explicit representations extracted from each field often disagree. These representations can differ even topologically.

At such points it is difficult or perhaps impossible to construct a consistent boundary mesh without gaps or holes in order to perform a desired simulation on the explicit structure. Consistent meshes are required for an entire class of simulation tasks in which values need to be related across the interfaces, such as mass, momentum, or energy fluxes. This class of simulation includes stress-strain and electromigration calculations on grain structures, which are desireable to link to structure evolution simulations. Bloomfield *et al.* [3] proposed a partial solution for this problem which employed rectilinear voxels and robustly arrived at extracted structures that were always consistent. However, these structures (as shown in Figure 1) were problematic in that they had surface triangles with only a small number of normal directions (*i.e.*, the structures were limited to “Manhattan geometries”). In this paper we demonstrate an algorithm that extends the voxel approach and removes the limitation of producing Manhattan geometries. By employing a mesh generator, we produce voxels that conform to the natural interfaces far from higher order junctions, and produce intuitive, topologically consistent, explicit boundary representations at higher-order junctions.

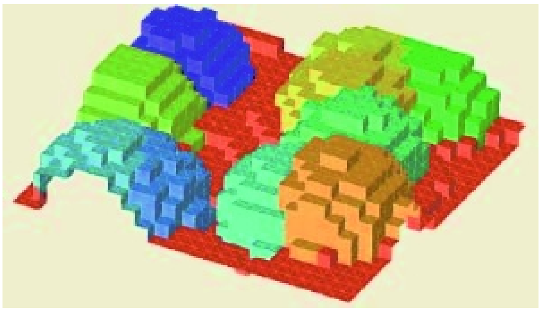

Fig. 1. The voxel extractor reported on by Bloomfield *et al.* [3] solves the consistency problem, but is limited to producing Manhattan geometries

2 Non-consistent Extraction

Although for the purposes of this paper, we represent level-set scalar fields (φ_i) on triangular meshes using linear finite element basis functions, the ideas regarding representation and extraction may be extended to non-simplex and mixed-type meshes, higher-order basis functions, and finite difference grids in both 2 and 3D. We will maintain the convention that $\varphi_i<0$ inside phase *i* and $\varphi_i>0$ outside phase *i* and use the signed distance to the interface as the value of φ_i. Note that each phase has a distinct level set function (φ-field). Figure 2 illustrates the problem with representing even simple structures. We begin by initializing three φ-fields to be the signed distances from the explicit starting interfaces (solid lines) and represented the fields on the underlying triangular mesh (dotted lines). Finally, we “recover” the explicit interfaces by extracting line segments (dashed lines) from the φ-fields by interpolating the zeros along the triangles’ edges. We see that what would have been a slight rounding of a corner in the standard level-set method [1.], becomes an unphysical situation in the multiple level-set method, leaving a *void* behind in which no single phase is present. It should be noted that the term void as used here does not refer to a physical vacuum, which in this level-set formulation would be treated as a phase and have its own φ-field. Here, void refers to a region of space not associated with any single phase.

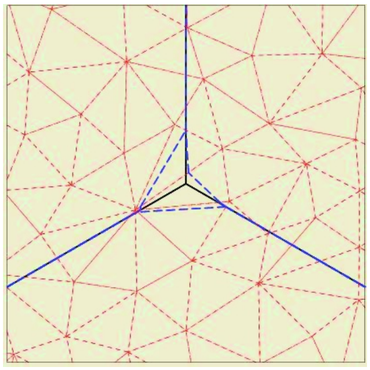

Fig. 2. An explicit set of interfaces (black solid lines) can be represented as contours in scalar fields, themselves represented on a finite element mesh (light dotted lines) or finite difference grid. However when an attempt is made to extract the interfaces, the extracted interfaces (dashed lines) do not come together at triple points and higher order junctions, leaving regions that do not correspond to any physical situation

Each φ-field can be extracted separately, giving smooth boundary representations of each phase in the most of the domain. In these regions, a match for each boundary element can be found among the elements extracted from other fields, allowing the boundary elements to be classified as being a particular type of interface. Constrained meshes may be constructed that include these elements as internal or boundary entities, allowing for further simulations. However, voids of the type shown in Figure 2 are not uncommon, occurring at almost all higher order junctions, and typically are one to two mesh-lengths across. The boundary elements that make up this void do not have matches in the set of elements extracted from other φ-fields; this prevents further execution of a program that requires tracking quantities across interfaces, from one phase to another.

3 Solution Methods

Carefully constructed meshes can potentially capture a single junction, allowing a consistent extraction, but this advantage is lost after the first time step in which the interfaces move. Alternatively, an approach may be taken to “fix” the non-physical situations after they occur, by locating their occurrence, localizing them, and then collapsing vertices and entities in such a way as to achieve a topologically consistent set of boundary representations. This approach, although fraught with special cases and tortuous decision trees, has been successfully implemented for extractions in two-dimensions [4]. We are unaware of a successful attempt to perform the three-dimensional analog. It is worth remarking that in three dimensions, the problem areas are not polygons, but are networks of long, thin “void wires” that meet at higher order junctions. Because of the daunting complexity of the problem in 3D, this approach does not appear promising.

3.1 Voxel Extraction

Voxel extraction has been employed as a partial solution to this dilemma by approaching the problem of extraction by constructing explicit *phases* before attempting to extract the interfaces between them. Bloomfield *et al.* [3] proposed filling the domain with regular rectilinear (*i.e.*, squares or cubes) and using the information embedded in the φ-fields to identify each of there volume elements (voxels) as a part of the explicit representation of the phases. They then looked for neighboring voxels that belonged to different faces and identified the elements between them as boundary elements of both faces. This generated a set of faces in three dimensions and a set of segments in two dimensions that were both consistent and located on or near the intuitively identified interface. Figure 1 shows and example of faces extracted using this voxel method. The results are completely consistent and each boundary element is identified from both sides of the interface.

The clearest drawback of this voxel method is the so-called Manhattan geometry of the results. Because the voxels have a limited set of directions for their faces, the extracted interfaces, which are made up of a subset of these faces, do not reflect the smoothly varying normals associated with most of the systems being represented. Two further improvements can be made to these output. The first approach is to apply a smoothing operator to the results, such as the volume-conserving algorithms reported on by Kuprat *et al.* [5.]. The downsides to this approach are that smoothing operations do not preserve shape global shape, tend to be motivated by aesthetic considerations rather than physical ones, and can be complicated to apply to very unsmooth surfaces. The second approach is to project onto each face a "pseudo-normal" calculated to be the unit direction along the gradient of the appropriate φ-field, similar to the technique used in Phong shading [6]. Such a projection method may be useful for tasks that require normals that more accurately represent the system, such as surface scattering calculations.

Conformal-Voxel Extraction. We demonstrate an extension of the above voxel extraction method that avoids the limitation of producing Manhattan geometries. By extending the concept of a voxelation to include not just tessellations of regular, identical shapes, but *any set of polyhedra that space-fill a domain,* we remove the constraint of having only a small number of distinct normals. With this definition, any geometric mesh that fills a volume (or fills an area in 2D systems) is a voxelation, and we can create a custom voxelation, using a ready-made meshing code and information about the structure in the φ-fields, that will have voxels optimally placed to give extractions that are consistent and are true to the implicit representation away from junctions.

Pulling information from the φ-fields is very important. Without providing guidance to the mesher, the voxel faces are as likely to be transverse to the zero contours in the implicit representation as to conform to them. The simplest way to provide this information is to use the parts of the directly extracted boundaries that do have matches in the other extracted boundaries. In two dimensions this is a set of segments and in three dimensions it is a set of faces, internal to the domain to be meshed. We call these internal elements *model entities*, and provide them to the area or volume mesher as constraints to be included as segments or faces in the resulting voxelation. As is shown below, this will cause the conformal-voxel extracted boundary representation to coincide with the directly extracted version in regions more than a few mesh lengths from triple points and higher order junctions.

The conformal-voxel extraction algorithm can be performed as follows:

1. Begin with discretizations of the n (assuming n phases) φ-fields that implicitly represent the structure to be extracted in a D dimensional domain $\mathbf{\Omega}$.
2. Extract a list $\mathbf{B}_\alpha$ of entities from each field φ_α, $\alpha \in [1,n]$ by interpolating along the level set contour of each.
3. Mark each entity which appears in exactly two of the extracted sets, $\mathbf{B}_\alpha$, as a model entity and add to list $\mathbf{B'}$.
4. Invoke an area (D=2) or volume (D=3) mesher to fill the simulation domain, providing the model entities in $\mathbf{B'}$ as internal constraints. Call this mesh the voxelation, $\mathbf{V}$.
5. Assign each voxel, $V \in \mathbf{V}$, a phase $\alpha(V)$, by an appropriate calculation referencing the fields φ_α.
6. For each voxel V, visit the voxel $U = V.neighbor(k)$ ($k \in [1,D+1]$) on the other side of each segment k (D=2) or face k (D=3). If $\alpha(V) \neq \alpha(U)$, add entity k to the set of conformal voxel extracted boundary entities, $\mathbf{B''}_{\alpha(V)}$.

Each entity in $\mathbf{B''}_\alpha$ should now have a match in some other $\mathbf{B''}_{\alpha' \neq \alpha}$ or be on the domain boundary $\partial\mathbf{\Omega}$ and the extraction is complete.

The algorithm is straightforward, with only two tasks left up to the discretion of the implementer. The first is the invocation of the mesher in step 4. Although this point is discussed in detail in section 4 of this work, in this work we use the freeware 2D quality mesh generator `triangle` [7] from the netlib repository [8] and find that no quality constraints on the mesh have to be specified as long as the maximum area of the triangles in the mesh is less than the average area of triangles in the finite element mesh that we use to represent the φ-fields. The second choice that must be made is how the identification of the voxels in done in step 5. Any heuristic that robustly determines in which phase the voxel is located should work; in this work we use the phase associated with the φ-field that has the lowest value at the voxel centroid as the identity of that voxel. This choice is intuitively pleasing, is easy to implement, and empirically works well.

3.2 Examples

In Figure 3, we show the various steps of a conformal voxel extraction of the same φ-fields that produced the void in Figure 2. First, from the directly extracted sets (left), model entities are identified (solid lines). Next, a voxelation (middle) is created using the model entities as internal constraints and each voxel is assigned to a phase (shading). Finally, the conformal voxel extraction (solid blue line, right) is derived by comparing neighboring voxels, and compared to the structure used to initialize the implicit representation (dashed line, right). The resulting extraction is consistent and faithfully captures the normals and positions of the initializing structure.

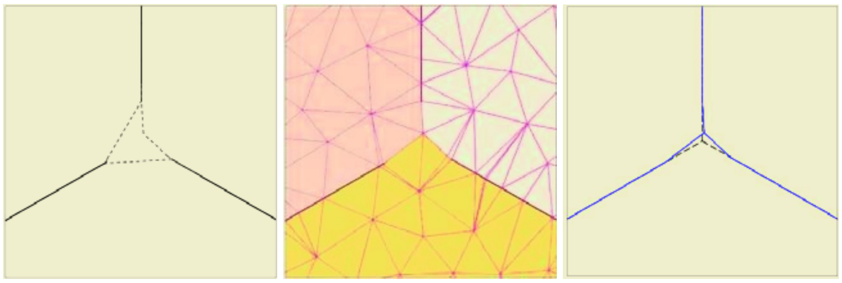

Fig. 3. The conformal voxel extraction procedure applied to the implicit representation shown in Figure 2. First, from the directly extracted sets (left), model entities are identified (solid lines). Next, a voxelization (middle) is created using the model entities as internal constraints and each voxel is assigned to a phase (shading). Finally, the conformal voxel extraction (solid blue line, right) is derived by comparing neighboring voxels, and compared to the structure used to initialize the implicit representation (dashed line, right)

In Figure 4, we show two more examples, the first (left) being a conformal voxel extraction of a higher order junction (a quadruple point). The extraction of the quadruple point indicates one aspect of voxel approaches, conformal or regular, which is that the topology of the extraction can be different that the topology of the structure the implicit representation is meant to represent. Here, the quadruple point present in the initializing structure (solid black lines) is present as a pair of triple points in the extraction. Further discussion of this phenomenon is given in the next section.

The second example in Figure 4 is a conformal voxel extraction (middle) of an 16-phase test structure (right), demonstrating the method's ability to robustly handle

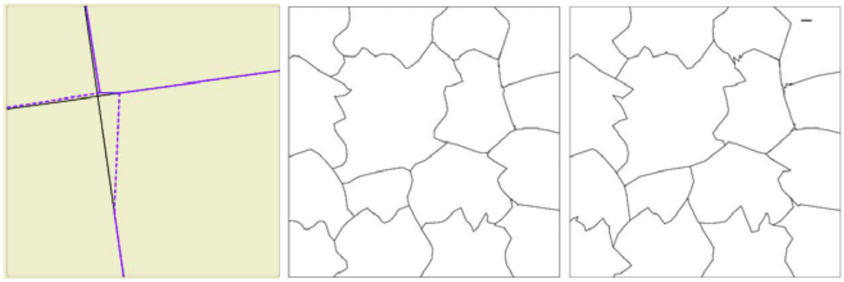

Fig. 4. (left) Drawing showing a conformal voxel extraction (dashed blue line) of a quadruple point (solid black line). Note that the extraction contains a pair of triple points instead of a single quadruple point. (middle) A conformal voxel extraction of (right) a complex 16-phase test structure containing significant fine detail, below the resolution of the underlying finite element mesh used

complex, many-phase structures. In this test structure, there are a variety of small and large geometric features. The extraction faithfully reproduces these geometric features underlying finite element mesh used for the implicit representation. (The input structure is 1x1; the bar in upper right corner of the right figure is 0.025 in length.)

Features smaller than this, such as the oscillations near some of the junctions in the test structure do not appear in the extraction.

4 Discussion and Conclusions

As mentioned in section 3.1, step 4 of the conformal voxel algorithm allows for discretion on the part of the implementer. It is essential that the voxelation include the internal constraints represented by the model entities, or at least alternative constraints based on them. That is, it may be permissible to adjust the coarseness of the model entities using a mesh coarsener to relieve some of the burden on the mesher to be used or to condition the elements in the set of extracted boundary entities. Any changes made to the information represented by the model entities will show up in the extracted boundary entities. However, we have observed in our 2D studies that the mesh quality of the voxelation is not of key importance in obtaining extractions that are true to the input structure. This is true providing that the mesher respects the constraints of the model entities and that the mesh size near the interfaces is less than or equal to the mesh size of the underlying discretizations of the field. In fact, we find that it is the mesh in the underlying finite element representation of the fields that has a determining effect on the amount of detail that is recovered using the conformal voxel extractor. In particular, the locations of triple and higher order junctions produced by the conformal voxel extractor deviate from the location of these junctions in the initializing structure by no more than the characteristic mesh size of the underlying finite element mesh.

This is a powerful result given that there is an implicit upper limit on the amount of detail retained about spatial variations in a field represented on a mesh or a grid with linear interpolants. By making a parallel to Nyquist's sampling theorem [9], we can say that the upper limit on the spatial resolution, *i.e.*, the smallest spatial variation represented is about twice the mesh length. For unstructured grids, this length is someone nebulous, but can be gauged within about a factor of 2, perhaps as a function of position. This agrees with what can be seen in the extraction of the complex test structure show Figure 4, indicating that the conformal voxel method is able to extract a large fraction of the information about the structure implicit in the finite element representation of the φ-fields.

The phenomenon of junction splitting seen in Figure 4 is also a matter of resolution. Although the implicit representation was initialized from an explicit structure with a quadruple point, it cannot be distinguished from the conformal voxel extracted version with two triple points within the resolution of the implicit representation. This situation highlights a characteristic of the comformal voxel method: a particular extraction from a given implicit representation is not unique. By using a voxelation produced by a different mesh algorithm or with different parameters, different extractions can be produced from the same implicit representation. However, in our experience, they will all be the same within the spatial resolution of implicit representation.

The computational complexity of this algorithm is difficult to analyze precisely, because of the use of a separate mesher to produce the voxelation, which will have its own complexity as a function of desired mesh size and number and type of internal constraints. However, if there are N mesh elements in the discretizations of the M input φ-fields, $O(N^{(D-1)/D})$ entities in the union of $\mathbf{B}_{\alpha}$. Matching up these entities is done pairwise, taking O(D) integer comparisons each, giving an overall complexity of $O(D \cdot N^{2(D-1)/D})$, or O(N) for 2D and $O(N^{4/3})$ for 3D for step 2. The complexity of step 5 is simply $O(N{\cdot}M)$, leaving the overall complexity for the algorithm system dependent based on the number of phases and the external mesher used.

We have shown several examples of the use of the algorithm in 2D. A demonstration of its use in 3D [10] is beyond the scope of this paper for reasons of brevity. The number of meshers available that can robustly handle the complex internal constraints in an automated fashion is much lower for 3D than for 2D. It should be pointed out that although we use triangular mesh elements for our voxelations, quadrilateral element should work just as well. In 3D, tetrahedral, wedges, and brick elements should be able to be used with equal facility.

This method is highly robust, given a reliable mesher. The only caveat is that if the voxel size is noticeably larger than the resolution of the implicit representation, then the resulting extraction may depend heavily on the voxelation. However, the method will not fail to complete even for very large voxel sizes, happily producing spurious results.

Finally, we note that there is a relationship between the voxelation mesh and the conformal voxel extracted result, namely that the former is a body-fitted mesh for the latter, and potentially can be used as the mesh for further simulation on the extracted system. Should this be desired, more attention should be paid to the quality of the voxelation as it is being produced.

References

1. Osher, S., Sethian, J.A.: Fronts Propogating with Curvature Dependent Speed: Algorithms Based on Hamilton-Jacobi Formulations. J. Comp. Phys. 79 (1988) 12-49.
2. Merriman, B, Bence, J,K, Osher, S.J.: Motion of Multiple Junctions: A Level Set Approach. J. Comp. Phys. 112 (1994) 334-363.
3. Bloomfield, M.O., Richards, D.F., Cale, T.S.: A Computational Framework for Modeling Grain Structure Evolution in Three Dimensions. Phil. Mag. 83(31-34) (2003) 3549-3568.
4. Bloomfield, M.O., Richards, D.F., Cale, T.S.: A Tool for the Representation, Evolution, and Coalescence of 3-Dimensional Grain Structures, presented during the Sixth United States National Congress on Computational Mechanics, August 1-3, 2001, Dearborn, MI.
5. Kuprat, A., Khamayseh, A., George, D., Larkey, L.: Volume Conserving Smoothing for Piecewise Linear Curves, Surfaces, and Triple Lines. J. Comp. Phys. 172 (2001) 99-118.
6. Phong, B.: Illumination for Computer Generated Pictures. Comm. of the ACM, 18(8) (1975) 311-317.
7. A Two-Dimensional Quality Mesh Generator and Delaunay Triangulator: http://www-2.cs.cmu.edu/~quake/triangle.html
8. Netlib Repository at UTK and ORNL: http://www.netlib.org/
9. Papoulis, A.: Signal Processing, McGraw Hill, 1997.
10. Bloomfield, M.O., Richards, D.F., Cale, T.S.: A Conformal Voxel Extraction Method for Extraction of Multiple Level Set Fields in Three Dimensions, *in preparation.*