An Accurate Implicit Field Representation for Meshes and Its Adapted Triangulation Algorithms

M. Fournier[1], J-M. Dischler[1] and D. Bechmann[1]

[1]Image Sciences, Computer Sciences and Remote Sensing Laboratory
[1]LSIIT – UMR 7005 – CNRS – Louis Pasteur University, Strasbourg, France

Abstract
The classic implicit scalar field distance transform representation of a mesh is very useful to perform many mesh processing operations and to obtain better results than with other methods. In this paper we propose a new and more accurate implicit vector field distance transform representation of a mesh. We adapt Marching Cube and Marching Triangle, the two most widely used triangulation algorithms, to our new vector field representation to correctly reconstruct the final mesh after data processing in the implicit domain. According to a reliable surface error metric, we show our new vector field is more accurate than the classic scalar field to implicitly represent a mesh. We adapt to our vector field a previously introduced mesh denoising algorithm performed on the scalar field. Results show mesh denoising with our vector field outperforms the one with classic scalar field in terms of an error metric comparison.

Categories and Subject Descriptors (according to ACM CCS):
I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling – Curve, surface, solid, and object representations; Geometric algorithms, languages, and systems; Hierarchy and geometric transformations.
I.4.3 [Image Processing and Computer Vision]: Enhancement – Filtering; Geometric correction; Smoothing.

1. Introduction

The classic implicit scalar field distance transform representation (SFDT) is a voxel-based volumetric implicit representation to describe a 3D object surface mesh. To compute SFDT we need to create a voxel grid inside the mesh bounding box. For each voxel the closest point on the mesh surface is found and the distance between the voxel and that closest point is saved in the voxel structure. When the voxels size tends toward zero, we have a continuous implicit surface representation of the mesh which correctly represents both the surface topology and geometry. The zero-set $f(x,y,z) = 0$ of the SFDT defines the mesh surface. The SFDT is used to perform mesh processing operations in the implicit domain. After data processing, we need to triangulate the SFDT in order to produce the resulting mesh. In practice the voxel size is finite and the domain conversion itself introduces a surface error which can be controlled by choosing an appropriate voxel grid resolution required for specific applications. Converting a mesh in its SFDT and without performing any operation on the SFDT, if we triangulate that SFDT to create a new mesh equivalent to the initial one, we will not obtain the exact same surface but only its approximation.

In this paper we propose a new implicit vector field distance transform representation (VFDT) which is more accurate than the SFDT. The VFDT minimizes the surface error introduced by the domain conversion. To triangulate our new VFDT we adapt the two most widely used algorithms, Marching Cube [LC87] and Marching Triangle [HSI*96], to correctly reconstruct the resulting mesh after implicit data processing. Since our new VFDT data structure is different than the SFDT, all mesh processing operations usually performed on SFDT will need to be adapted to VFDT. As a first application of our VFDT, we adapt a previously introduced mesh denoising algorithm [FDB06] performed on SFDT and show the results quality is better with our VFDT. The goal of this paper is to introduce a more accurate VFDT for meshes and to adapt SFDT mesh processing operations to VFDT in order to produce more accurate results compared to SFDT. The remaining parts of the paper are organized as follows: Section 2 overviews related works. Section 3 presents our new VFDT and its adapted triangulation algorithms. Section 4 presents the mesh denoising algorithm adapted to our VFDT. And in Section 5, before concluding, we discuss the results of triangulation and mesh denoising with our new VFDT compared with the SFDT representation.

2. Related works

The SFDT is an important volumetric alternative surface representation which is widely used to perform many mesh processing operations. For many applications, even if the domain conversion introduces a surface error, using the SFDT produces better results than other methods and the algorithms are often easier to implement or they provide more versatile tools which enable new processing features. In the context of real 3D objects reconstruction from scanned data, the SFDT has the advantage of being able to process every step of the reconstruction procedure in the same data structure and it produces good results at each step with relatively simple algorithms. In that context the SFDT was previously used in mesh fusion [CL96] to integrate all range images into a unique representation, followed by mesh repair and mesh simplification [NT03] to fill holes in the mesh and to produce a more compact model, and then mesh smoothing and denoising [FDB06] to remove acquisition noise introduced by the scanner.

We propose a new VFDT which will contribute to enhance the results quality of these mesh processing operations in the implicit field representation. We already have adapted a mesh denoising algorithm [FDB06] to our VFDT which shows good results and we are currently working to adapt mesh fusion algorithm [CL96] to the new VFDT. As far as we know, we will be able to adapt most of SFDT mesh processing operations to our VFDT and since the model is more accurate than SFDT, we are confident the results will also be more accurate. Furthermore many recent works such as [JLS*02, BPK05] focused on improving the generated surface quality and approximation over classical Marching Cube by proposing Marching Cube extensions and dual contouring from SFDT. These works aim at better fitting the Marching Cube surface to the input data using polynomial fitting or equivalent techniques and they are suitable for our VFDT. We also are currently working to adapt these major improvements to the new VFDT which will hopefully produce better results compared to their SFDT implementation.

3. VFDT definition and its adapted triangulations

As for SFDT, the VFDT of a mesh is defined over a voxel grid. For each voxel the closest point on the mesh surface is also found. But instead of saving only the distance as in SFDT, in VFDT the 3D vector starting from the voxel and pointing to that closest point found on the surface is saved in each voxel. This new VFDT is more precise and complete than SFDT representation because at each voxel we know the distance as well and in addition we know the exact orientation of the closest point on the surface. As for SFDT, zero-set $f(x,y,z) = 0$ when VFDT vectors length are zero defines the mesh surface. In practice with a discrete and finite grid resolution we need to fix a threshold to determine if a voxel is on the surface. With the SFDT representation it introduces an approximation error on the surface vertices. In the VFDT representation we use the small residual vector of the surface voxels to correct the vertices coordinates and retrieve the exact surface position.

3.1. First Marching Cube triangulation adaptation

The Marching Cube is a widely used triangulation algorithm to reconstruct a mesh from SFDT. We adapt this algorithm to correctly reconstruct the mesh from our VFDT representation. The overall algorithm is the same with the VFDT except for the interpolation part to find the new triangle vertices on the cubes edges. Figure 1 is a 2D example which shows the new VFDT interpolation method and its advantage compared to SFDT vertex interpolation.

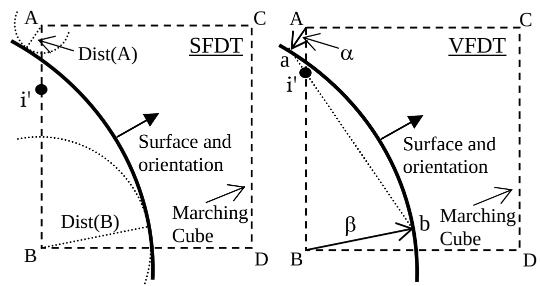


Figure 1: *SFDT and VFDT surface interpolation*

In Figure 1 example, A, B, C and D are neighbor voxels of the distance transforms and the doted square represents the "marching cube". In the SFDT representation, Dist(A) and Dist(B) are the scalar minimal distances to the surface from voxels A and B. In the VFDT representation, α and β are the vectors from voxels A and B to the closest points a and b on the original surface. At reconstruction step with the Marching Cube algorithm, we need to interpolate a new surface vertex i' on edge AB. With the SFDT representation, adding the two distances Dist(A) and Dist(B) leads to a first incoherence according to edge AB length. In general cases, either there is no solution or two solutions (infinity of solutions in 3D) for the intersection of the two circles of radius Dist(A) and Dist(B) which are centered on A and B. So the rough interpolation is made with the ratio of distances Dist(A) and Dist(B) transposed on edge AB length which leads to a greater error.

With the VFDT representation, a and b original surface points are used to interpolate vertex i' on edge AB at intersection with interpolated surface segment ab. In general 3D cases, edge AB and interpolation segment ab will not intersect so the closest point to segment ab on edge AB is found and it is the new surface interpolated vertex i'. This interpolation method gives a better approximation of the original surface at same grid resolution compared to the SFDT representation. We define vector u = AB and vector v = ab. Then the interpolated vertex $i' = \lambda u$ and Equation 1 shows how to find λ factor which resizes vector u to obtain the interpolation vertex i' on edge AB. The dot operator ($\circ$) defines a dot product between vectors.

$$\lambda = \frac{(u \circ v)(v \circ \alpha) - (v \circ v)(u \circ \alpha)}{(u \circ u)(v \circ v) - (u \circ v)^2} \qquad (1)$$

Figure 2 shows an example of Marching Cube triangulation over a SFDT and a VFDT of the same object.

In Figure 2 example, the SFDT and VFDT of a mesh which defines a 3D step function have been computed. Then the Marching Cube algorithm has been applied to triangulate both of them, using the standard algorithm over the SFDT and the modified one over the VFDT representation. The same coarse grid resolution at same spatial position has been defined for both distance transforms to highlight the differences and advantages of the VFDT representation. With the SFDT sharp edge A has been respected, there are unwanted inflection points at B and E and sharp edges at corners C, D, and F have been truncated. With the VFDT only sharp edge at corner C has been truncated because of the very low grid resolution used in this example. At same grid resolution, the VFDT has a more accurate representation of the underlying mesh.

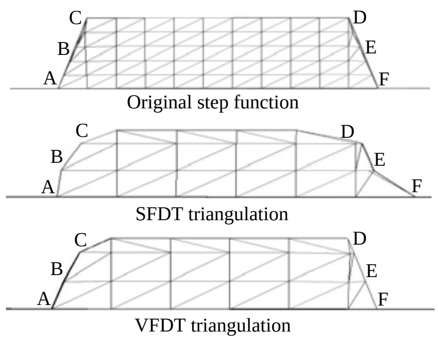
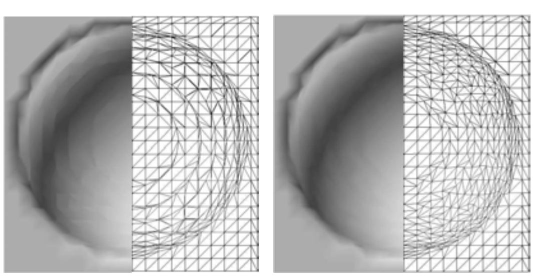


Figure 2: *SFDT and VFDT step function triangulation*

Figure 3 shows another advantage of the VFDT representation in terms of the triangles size and distribution produced by the Marching Cube algorithm on a half sphere model. In Figure 3 example both raw triangulations are shown without any mesh simplification algorithm. Both results have the exact same amount of triangles and vertices. On the SFDT triangulation, we see the usual Marching Cube elevation lines produced with unwanted small degenerated triangles. The VFDT representation produces a more uniform triangulation without any small degenerated triangle. The VFDT solid shaded model is therefore visually more uniform than the SFDT model. This VFDT representation advantage can save a simplification step for specific applications.

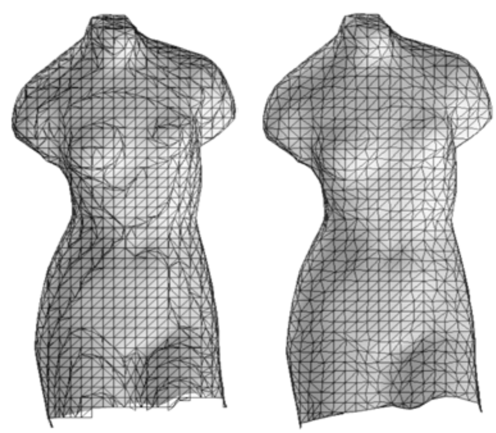


SFDT triangulation VFDT triangulation

Figure 3: *SFDT and VFDT half sphere triangulation*

3.2. Second Marching Cube triangulation adaptation

We propose a second Marching Cube adaptation and as before, the overall algorithm is equivalent to the SFDT one except for the interpolation part. Actually there is no more interpolation in this adaptation. Referring to Figure 1 VFDT example, if a new vertex position needs to be interpolated on edge AB, instead of actually interpolating that vertex, we simply use one of the two initial surface points a or b pointed by vectors α or β. This method is fast and easy to compute, we only compare both vectors α and β length and we keep the shortest one. In Figure 1 example, vector α is shorter than vector β and the new vertex position would simply be initial surface point a. This second Marching Cube adaptation has two major differences with SFDT triangulation. First the new vertices are no longer on the cubes edges, they are exactly on the initial surface and this gives a better surface approximation. Second the resulting mesh has fewer triangles, some of the triangles which would have been created in SFDT no longer exist. Depending on the cubes configuration, a triangle can collapse into an edge or into a single vertex if two or three adjacent voxels which need to be interpolated result in the same new vertex according to their vector length. These two differences do not affect surface continuity while adjacent cubes will produce same new vertices and if triangles collapse, their neighbors will adapt their size to fill empty spaces. Figure 4 shows triangulation results on the Venus model for the SFDT and the VFDT with this new Marching Cube adaptation.



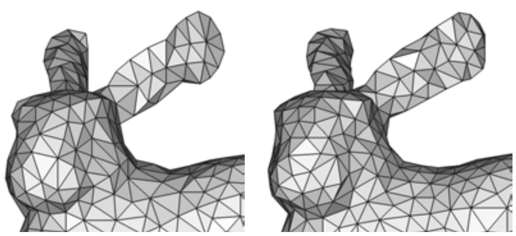
SFDT: Error = 1.47×10^{-2} VFDT: Error = 1.29×10^{-2}

Figure 4: *SFDT and VFDT Venus model triangulation*

Results in Figure 4 have been compared to initial Venus model mesh with vertex to surface error metric introduced in [FDB06]. VFDT triangulation has only 3228 triangles and it is 12.2% better than SFDT which has 5463 triangles. VFDT triangulation is even more uniform with this method while SFDT still has these small degenerated triangles elevation lines. Moreover at the model bottom end we see SFDT result do not match exactly the initial model underneath the mesh. VFDT do not have this problem at same grid resolution which is another VFDT advantage.

3.3. Marching Triangle triangulation adaptation

We adapt Marching Triangle to our VFDT simply by changing the algorithm step 2 defined in [HSI*96]. With SFDT, at step 2 a search is made to find the nearest surface point from a previously projected point. The result depends on a threshold and it introduces an approximation error on the new vertex position evaluation according to the initial surface. With VFDT, step 2 search is no longer needed. From the projected point, we simply add the vector of the current voxel to obtain the nearest exact initial surface point without any error. Figure 5 shows Marching Triangle results on the bunny model for SFDT and VFDT.



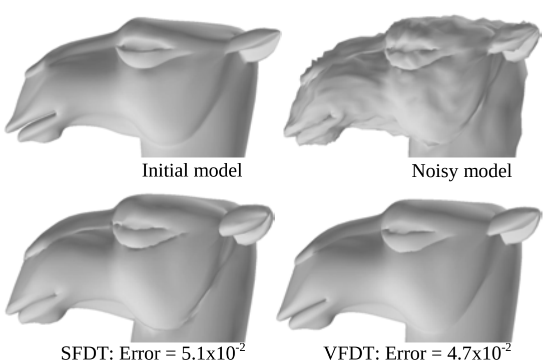
SFDT: Error = 5.81×10^{-2} VFDT: Error = 5.39×10^{-2}

Figure 5: *SFDT and VFDT bunny model triangulation*

In Figure 5 example, same triangulation parameters such as projection distance have been used for both resulting meshes which are visually similar. Low resolution has been selected to highlight differences between models which have almost same amount of triangles, 3472 for SFDT and 3621 for VFDT. The bunny left ear has some deformation in SFDT triangulation. VFDT resulting mesh has slightly more uniform triangles and it is 7.2% better than SFDT mostly because of error free new vertex position.

4. VFDT mesh denoising algorithm

As a first application to demonstrate the efficiency of our VFDT, we adapt the mesh denoising method introduced in [FDB06] which is performed on SFDT. It is an adaptive and feature preserving filtering algorithm which is based on a noise variance threshold. The filter equation works on scalar data and we simply use it independently for each vector coordinate in order to filter the VFDT. Figure 6 shows filtering results on the camel model.



SFDT: Error = 5.1×10^{-2} VFDT: Error = 4.7×10^{-2}

Figure 6: *SFDT and VFDT camel model filter results*

The camel model has been corrupted with artificial noise, filtered with both implicit representations and compared to the initial model with the error metric. The VFDT result has smaller error and it is visually better especially in high curvature regions such as under the eye and along the jaw.

5. Results

Figure 2 and Figure 3 show the first Marching Cube triangulation with the interpolation adaptation to VFDT produce better results than the standard SFDT triangulation. The results are more uniform without small degenerated triangles and they are more accurate compared to the initial model. Figure 4 shows the second Marching Cube adaptation produce a more accurate mesh boundary result with fewer triangles. Figure 5 shows Marching Triangle algorithm is also suitable with our VFDT and the result is also better than with SFDT. In the second Marching Cube and Marching Triangle adaptation cases, both algorithms are simplified compared to the ones with SFDT. Figure 6 shows the new VFDT is well adapted to mesh denoising and the result quality is enhanced compared to the equivalent algorithm performed on SFDT.

Conclusion and future works

A new VFDT has been introduced in this paper. It is a more complete and accurate implicit mesh representation than the classic SFDT. The two most widely used triangulation algorithms have been adapted to the new VFDT to correctly reconstruct its resulting mesh. A mesh denoising algorithm previously performed on SFDT has been adapted to VFDT and results show it outperform the previous one with SFDT. We are currently working to adapt mesh fusion to VFDT and preliminary results are encouraging. In future works we will adapt to our VFDT other useful mesh operations usually performed on SFDT.

References

[BPK05] Bischoff S., Pavic D., Kobbelt L.: Automatic Restoration of Polygon Models. ACM Transactions on Graphics, Vol. 24, No. 4, 2005, 1332-1352.

[CL96] Curless B., Levoy M.: A Volumetric Method for Building Complex Models From Range Images. *ACM SIGGRAPH 1996*, New Orleans, USA, 1996, 221-227.

[FDB06] Fournier M., Dischler J-M., Bechmann D.: 3D Distance Transform Adaptive Filtering for Denoising Triangle Meshes. *Graphite 2006*, K.L., Malaysia, 2006.

[HSI*96] Hilton A., Stoddart A.J., Illingworth J., Windeatt T.: Marching Triangles: Range Image Fusion for Complex Object Modelling. *International Conference on Image Processing*, Lausanne, Switzerland, Vol. 1, 1996.

[JLS*02] Ju T., Losasso F., Schaefer S., Warren J.: Dual Contouring of Hermite Data. *ACM SIGGRAPH 2002*, San Antonio, USA, 2002, 339-346.

[LC87] Lorensen W.E., Cline H.E.: Marching Cubes: A High Resolution 3D Surface Reconstruction Algorithm. *ACM SIGGRAPH 1987*, Anaheim, USA, 1987, 163-169.

[NT03] Nooruddin F.S., Turk G.: Simplification and Repair of Polygonal Models Using Volumetric Techniques. *IEEE Visualization*, Vol. 9, No. 2, 2003.