TOWARDS AN AUTOMATED MESH SMOOTHER FOR HIGH-ORDER MESHES: REPRESENTING THE BOUNDARY BY A LEVEL SET FUNCTION [*]

Veselin Dobrev[1] Patrick Knupp[2] Tzanio Kolev[1] Ketan Mittal[1]
Vladimir Z. Tomov[1,†]

[1] *Lawrence Livermore National Laboratory, Livermore, CA, U.S.A.*
kolev1@llnl.gov, mittal3@llnl.gov, tomov2@llnl.gov
[2] *Dihedral LLC, Bozeman, MT, U.S.A. knupp.patrick@gmail.com*
[†] *Corresponding author*

ABSTRACT

This research note presents an algorithm that takes an input mesh and constructs a finite element function, on an automatically defined background mesh, whose zero level set agrees with the boundary of the input mesh. The input does not include any additional geometric information except the mesh itself. A critical requirement for this algorithm is to define gradients of the constructed level set function both inside and *outside* of the input domain. The main challenges for this construction is to produce a smooth field, on both sides of the boundary, and to obtain a zero level set that agrees with the input domain. This algorithm is a first step in developing an automated mesh smoother that takes an input mesh, without any extra geometric information, and optimizes its quality by node movement of all internal and boundary nodes.

Keywords: implicit geometry, level set function, distance solver

1. INTRODUCTION

The capability to improve the quality of an input mesh, without any additional geometric information except the mesh itself, is of practical importance because often times the geometric information of a mesh (STL files, analytic descriptions, etc) is not available, especially for researchers and practitioners who are not involved in mesh generation work. Our goal is to develop a mesh optimization tool that has the above capability and requires minimal user input. The first step in this direction is to represent the boundary of the domain in a form that can be used for mesh optimization calculations. Because we are targeting high-order finite element (FE) discretizations [1], we choose to represent the boundary by the zero level set of a FE function. This research note goes over the technical details of the construction of this FE function. The main challenges for this procedure is to produce a smooth field, on both sides of the boundary, and to obtain a zero level set that agrees with the boundary of the input domain.

2. CONSTRUCTION OF THE LEVEL SET FUNCTION

Let $\mathcal{M}_0$ be an input mesh of order k in 2D or 3D with any standard type of elements (triangles / quadrilaterals / tetrahedra / hexahedra). By x_0 we denote physical positions on $\mathcal{M}_0$. We also construct a background mesh $\mathcal{M}_B$, details are given later, and denote physical positions on $\mathcal{M}_B$ by x_B. The background mesh

[*]PERFORMED UNDER THE AUSPICES OF THE U.S. DEPARTMENT OF ENERGY UNDER CONTRACT DE-AC52-07NA27344 (LLNL-CONF-840061).

$\mathcal{M}_B$ completely covers $\mathcal{M}_0$, i.e., for every x_0 there's a corresponding x_B, but not the other way around. Our goal is to represent the boundary of $\mathcal{M}_0$ as the zero level set of a finite element function $\sigma_B(x_B)$, on the background mesh $\mathcal{M}_B$. Using $\mathcal{M}_B$ is necessary because it would allow to compute gradients of σ_B on both sides of the original boundary, which is essential for optimization methods. Furthermore, as the goal is to compute gradients, σ_B must be a smooth function. This motivates our choice to compute σ_B, on $\mathcal{M}_B$, as the FE distance function to the boundary of the input domain $\mathcal{M}_0$.

The computation of σ_B consists of several steps which are discussed in the following paragraphs:

1. Computation of a FE distance function σ_0, on $\mathcal{M}_0$, to the boundary of the input mesh $\mathcal{M}_0$.
2. Construction of adaptively refined background mesh $\mathcal{M}_B$ and transfer of $\sigma_0(x_0)$ to $\sigma_{0B}(x_B)$ on $\mathcal{M}_B$.
3. Computation of σ_B using the one-sided distance to the boundary $\sigma_{0B}(x_B)$.

The description of each step also includes a visual illustration on a sample 2D input turbine blade mesh shown in the left panel of Figure 1.

The computation of the FE distance function σ_0 utilizes the p-Laplacian distance computation, see Section 7 in [2]:

$$\begin{aligned} &\nabla \cdot \left(|\nabla \sigma_0|^{p-2} \nabla \sigma_0\right) = -1 \text{ in } \mathcal{M}_0, \\ &\sigma_0 = 0 \text{ on } \partial \mathcal{M}_0, \quad 2 \le p < \infty. \end{aligned} \tag{1}$$

This formulation computes distance with respect to the boundary of the domain, but the same approach can be customized for arbitrary level sets of discrete functions by choosing appropriate finite element basis functions, see Section 3 in [3]. We use (1) in both boundary and LS regimes. Other distance solvers could also be used, but in our experience (1) reliably produces smooth behavior around the zero level set, which is our most major requirement. Although higher values of p increase the distance accuracy, we use $p = 5$ as the distance property is not critical for our purposes. The computed distances are chosen to have positive sign. The obtained σ_0 with respect to the boundary of the sample mesh is shown in the right panel of Figure 1.

Once σ_0 is calculated, we proceed by constructing the adaptively refined background mesh $\mathcal{M}_B$. The purpose of $\mathcal{M}_B$ is to provide (i) a smooth function σ_B on both sides of the boundary of $\mathcal{M}_0$, with the zero level set being at the boundary, and (ii) high resolution around the zero level set, so that values and gradients of σ_B are computed accurately. The domain of $\mathcal{M}_B$ is a box of size 20% more than the bounding box of $\mathcal{M}_0$, so that the boundaries of $\mathcal{M}_0$ are inside. The construction starts with creating $\mathcal{M}_B^0$ that has 4 elements in each direction, quads in 2D or hexes in 3D, and then repeats the following iteration a fixed number of times:

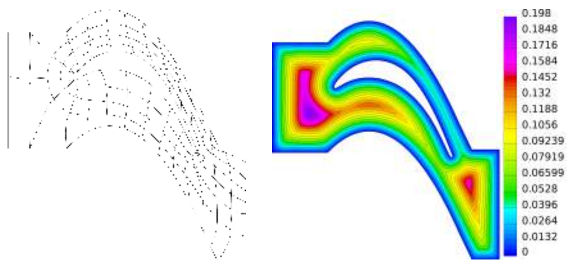

Figure 1: Input mesh $\mathcal{M}_0$ (left) and the resulting distance function σ_0 (right).

1. Interpolate σ_0 from $\mathcal{M}_0$ to $\mathcal{M}_B^i$, obtaining σ_B^i on $\mathcal{M}_B^i$. This operation is a mesh-to-mesh tranfer in physical space of a finite element function. It is performed through the *gslib* library, see [4, 5]. For points that are outside the domain of $\mathcal{M}_0$, we set $\sigma_B^i = -0.1$, an arbitrary negative value.
2. If $i > i_{\max}$, then stop and take $\mathcal{M}_B = \mathcal{M}_B^i$ and $\sigma_B = \sigma_B^i$. Otherwise refine those elements of $\mathcal{M}_B^i$ that contain both positive and negative values of σ_B^i. Refine their face-neighbors as well, to improve the resolution around the zero level set. The resulting mesh is $\mathcal{M}_B^{i+1}$, which is then used in step 1.

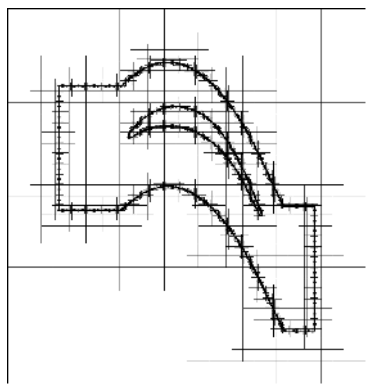
Figure 2: Background mesh $\mathcal{M}_B$ with 9 refinement levels around the boundary of $\mathcal{M}_0$.

Each iteration adds another level of refinement. The result of the above procedure on the sample mesh is shown in Figure 2, with $i_{\max} = 9$. The corresponding one-sided distance function σ_{0B} is shown in Figure 3. In the elements of $\mathcal{M}_B$ that cover the boundary of

$\mathcal{M}_0$, i.e., those elements that are cut by the zero level set of σ_0, σ_{0B} always exhibits a non-smooth transition to -0.1. These oscillations are evident in Figure 4 that zooms around the left tip of the blade. More specifically, the zero LS of σ_{0B} does not coincide with the zero LS of σ_0, because these non-smooth transitions introduce numerical errors in the cut elements; resolving this will be the focus of the final step below.

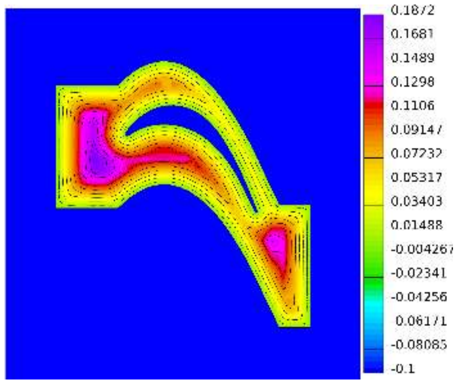

Figure 3: One-sided distance function σ_{0B} on $\mathcal{M}_B$
.

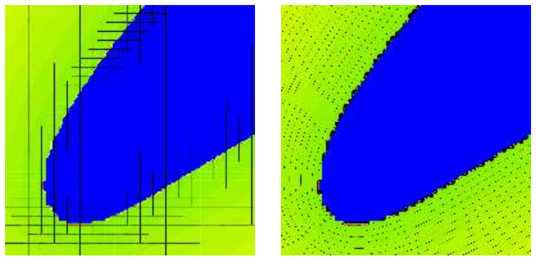

Figure 4: Zoom at the left tip of the blade showing the oscillating cut elements around the zero level set of σ_{0B}.

The last ingredient to represent the boundary is the computation of the two-sided distance function σ_B. Since the zero LS of σ_{0B} is not correct due to the numerical oscillations in the cut elements (Figure 4), direct distance computation (1) from the zero LS of σ_{0B} is not feasible. But one can observe that the distances are correct and smooth in the next layer of elements, as these elements are fully contained inside $\mathcal{M}_0$. Thus we can consider a *shifted* level set $\sigma_{0B}(x) = \Delta x$, which agrees with $\sigma_0(x) = \Delta x$ without numerical oscillations, where Δx is the minimum edge length of $\mathcal{M}_B$. We use (1) to compute the distance $\mathcal{D}_B$ to the zero level set of the function $\sigma_{0B}(x) - \Delta x$. Finally, we set

$$\sigma_B(x) = \mathcal{D}_B(x) + \Delta x.$$

This produces a smooth zero LS for the reasons explained above, and it is a good approximation since the zero level sets of σ_0 and $\mathcal{D}_B$ are parallel (up to the numerical error of solving (1)) and very close to each other, as Δx is the minimum size of the adaptively refined mesh. The computed σ_B for our sample mesh is shown in Figure 5. Its zero LS is shown next to the starting mesh in Figure 6. Zoomed views of the zero level set and isolines of σ_B at different parts of the domain are shown in Figures 7, 8, and 9.

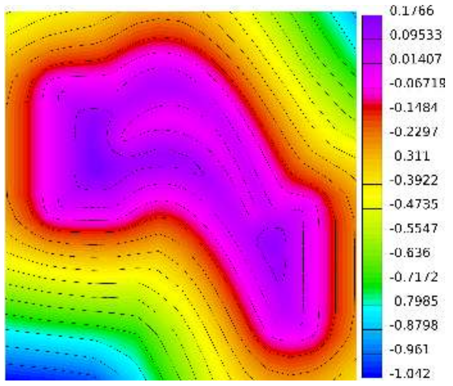

Figure 5: Two-sided distance function σ_B on $\mathcal{M}_B$
.

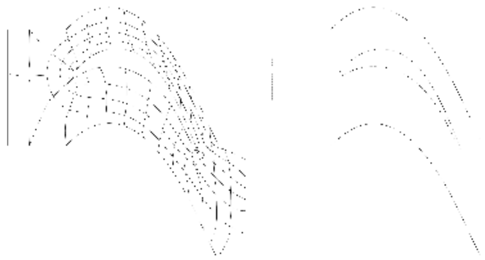

Figure 6: Input mesh and its finite element representation by the zero level set of σ_B.

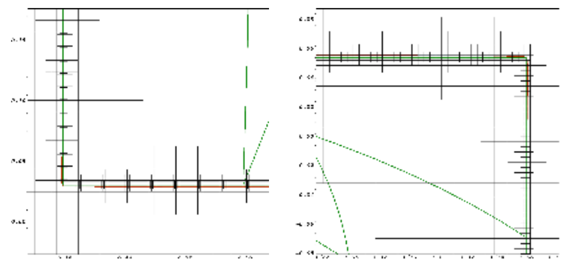

Figure 7: Zoom of the zero isolines (red) of σ_B at the bottom-left and top-right of the blade (green - $\mathcal{M}_0$ elements, black - $\mathcal{M}_B$ elements).

3. CONCLUSION

This research note presents an algorithm that constructs a finite element level set function, on a background mesh, that represents the boundary of a given input mesh. The method is demonstrated on a curved mesh with nontrivial boundary that has both smooth and sharp localized features. The results in Figures 7 and 8 show promise that the obtained zero level set

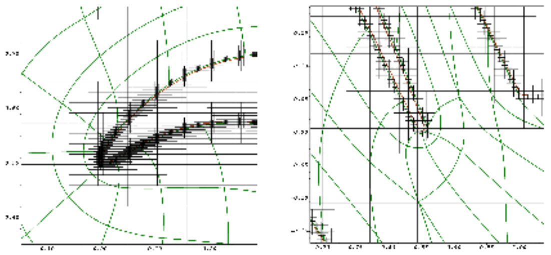
Figure 8: Zoom of the zero isolines (red) of σ_B at the left tip and the right tip of the blade (green - $\mathcal{M}_0$ elements, black - $\mathcal{M}_B$ elements).

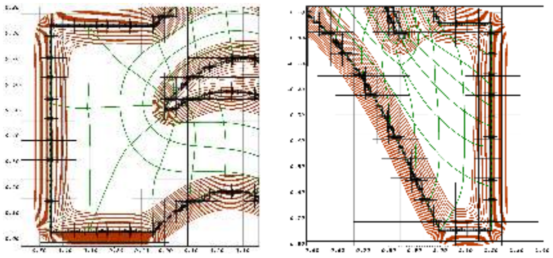
Figure 9: Zoom of the isolines (red) of σ_B at the left side and the right side of the blade (green - $\mathcal{M}_0$ elements, black - $\mathcal{M}_B$ elements).

can agree with the input boundary up to any accuracy (by performing more adaptive refinements on the background mesh), and the obtained function is smooth and well-behaved around the zero level set (Figure 9). Further testing will be performed on more 2D and 3D geometries. In the future we plan to combine this algorithm with volumetric mesh optimization [6] and tangential relaxation that moves boundary nodes along the zero level set of the constructed function [7].

References

[1] Anderson R., Andrej J., Barker A., Bramwell J., Camier J.S., Cerveny J., Dobrev V.A., Dudouit Y., Fisher A., Kolev T.V., Pazner W., Stowell M., Tomov V.Z., Akkerman I., Dahm J., Medina D., Zampini S. "MFEM: a modular finite elements methods library." *Comput. Math. Appl.*, vol. 81, 42–74, 2021

[2] Belyaev A.G., Fayolle P.A. "On Variational and PDE-Based Distance Function Approximations." *Comput. Graphics Forum*, vol. 34, no. 8, 104–118, 2015

[3] Rvachev V.L. *Theory of R-functions and Some Applications (In Russian).* Nauk Dumka, 1982

[4] Fischer P. "GSLIB: sparse communication library [Software]." `https://github.com/gslib/gslib`, 2017

[5] Mittal K., Dutta S., Fischer P. "Nonconforming Schwarz-spectral element methods for incompressible flow." *Computers & Fluids*, vol. 191, 104237, 2019

[6] Dobrev V.A., Knupp P., Kolev T.V., Mittal K., Tomov V.Z. "The Target-Matrix Optimization Paradigm for high-order meshes." *SIAM Journal on Scientific Computing*, vol. 41, no. 1, B50–B68, 2019

[7] Knupp P., Kolev T.V., Mittal K., Tomov V.Z. "Adaptive surface fitting and tangential relaxation for high-order mesh optimization." *29th International Meshing Roundtable*, 2021