Supplemental Material to the CSC2520 Project Report

Shashwat Sharma

April 09, 2018

Specific sections of the attached report are elaborated upon in this document for clarity and completeness.

1 Details on the Target Application

In some electromagnetic solvers, such as the ones targeted in this work, it is computationally advantageous to perform operations on several smaller segments of a mesh, rather than a few operations on large parts of the mesh. Additionally, certain simplifying approximations can be made on parts of the mesh that have some canonical constant cross section (such as a rectangle), and this can greatly speed up the overall computation. For these reasons, there is much to be gained by automatically segmenting the meshes in question into smaller parts that have some canonical structure. The meshes typically consist of several long distinct components. Some of the components have rectangular sections with bends along their length. This is a clear candidate for segmentation; splitting the long segments into shorter ones will speed up computation, and segmenting the objects with bends in them will lead to the creation of purely rectangular segments, which will also speed up computation. Since it is known that most mesh objects will be rectangles with bends, this information is used to simplify the segmentation process in this work.

2 Skeletonization Details

As mentioned in the main report, when computing signed distances, we do not need to consider the z-direction, since we know (or we are assuming) that the mesh has a constant height. This implies that the z-coordinate of the skeleton of the mesh must always be $(z_{\min} + z_{\max})/2$. If the z-coordinate is considered in signed distances, the optimization process may actually be hurt. This could be happen if the mesh is shorter than it is wide, as pictured in Fig. 1a. In this case, once the sample point (shown in red) reaches the centre line of the mesh (Fig. 1b), the condition of minimizing signed distance is satisfied, even though the point is clearly not at the actual centre of the

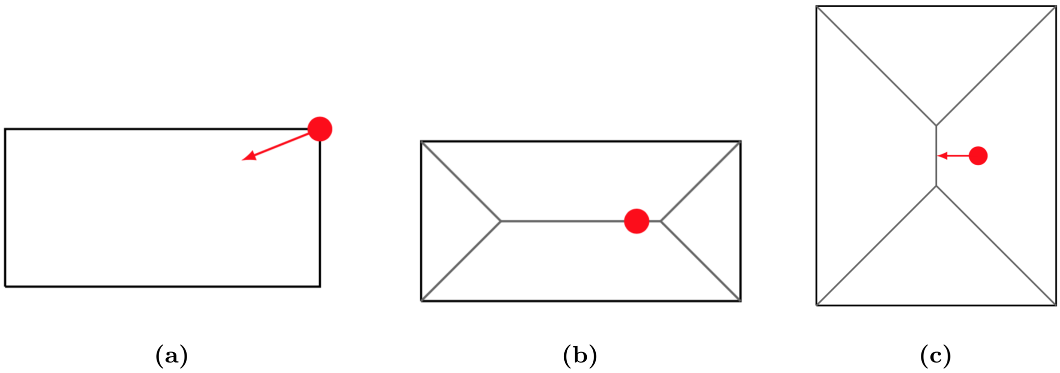


Figure 1 Disregarding the height of the cross-section in signed distance computation.

mesh. In order to remedy this, the upper and lower boundaries of the cross section can temporarily be displaced by a large, unrealistic amount so that the height is much larger than the width, as shown in Fig. 1c. Now, the sample point needs to move towards the actual centre of the mesh in order to minimize the signed distance. When done within a fine enough tolerance range, this ensures that the sample points end up along the desired skeleton of the mesh.

Once the z-direction is of no concern, the direction in which the sample points must move is given simply by the inward normal at the starting position of the sample point (which is known, since the sample point is just a mesh vertex to begin with). The z-component of this vector is set to 0 because we know that the point will only move along the xy-plane.

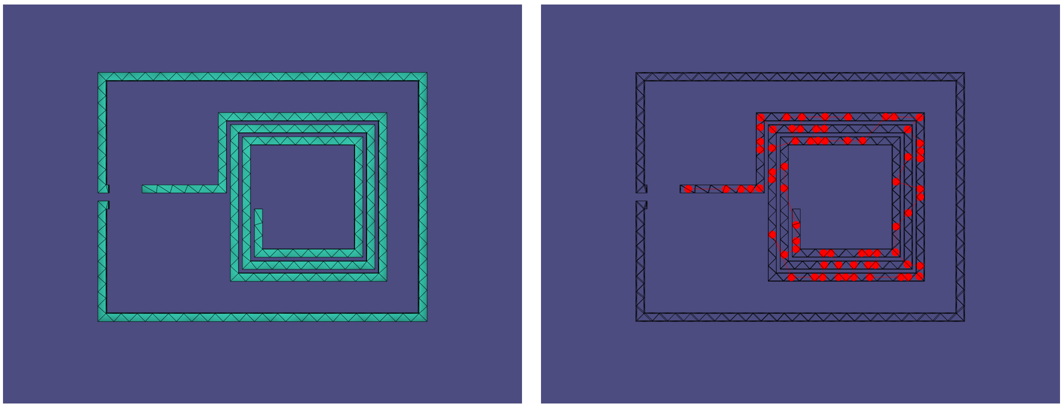
3 Mesh Segmentation Details

Once the sample points are ordered, they can be segmented based on the two conditions outlined in the main document. The collinearity condition is described here: suppose we are currently at sample point $\mathbf{p}_i$, surrounded by adjoining points $\mathbf{p}_{i-1}$ and $\mathbf{p}_{i+1}$. Then the condition for collinearity, given some tolerance ε, is

$$\left\|\mathbf{p}_i - \mathbf{p}_{i-1}\right\| + \left\|\mathbf{p}_i - \mathbf{p}_{i+1}\right\| - \left\|\mathbf{p}_{i+1} - \mathbf{p}_{i-1}\right\| \leq \varepsilon. \tag{1}$$

Once the skeleton is segmented, the mesh can easily be segmented as well. First, we define a plane using the following two vectors: $\mathbf{v}_1 = \begin{bmatrix} -n_x & -n_y & 0 \end{bmatrix}$ which is the lateral inward normal mentioned above, and $\mathbf{v}_2 = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$. The plane shared by these vectors is the plane perpendicular to the the skeleton at that sample point, and can be used to segment the plane. The normal vector to this plane can be written as $\mathbf{n}_p = \mathbf{v}_1 \times \mathbf{v}_2$. Additionally, we know that the newly optimized sample point $\mathbf{p}_i^*$ lies on this plane, and thus we can write the equation of the plane easily as follows:

$$(n_{px} - p_{ix}^*)x + (n_{py} - p_{iy}^*)y + (n_{pz} - p_{iz}^*)z = 0 \tag{2}$$



(a) (b)

Figure 2 A mesh example that cannot be handled by this algorithm.

where n_{px}, n_{py} and n_{pz} are the components of $\mathbf{n}_p$, and p^*_{ix}, p^*_{iy} and p^*_{iz} are coordinates of $\mathbf{p}^*_i$.

Now, any set of mesh vertices, when plugged into the above equation, can be classified as sitting on one or the other side of the plane, depending on whether the results of the equation are negative or positive. This clearly could have pitfalls for meandering meshes, but it works well for simpler structures such as the one considered in the main document.

4 An Example that Fails

As mentioned above and in the main document, the approach used in this work does not work for meshes that have multiple windings are bends beyond 90°. An example of this is shown in Fig. 2a.

The main reason that this example is not handled well is that the ordering the points along the skeleton is done based on the physical distances between points. However, this approach clearly cannot work in this case unless the sampling is extremely fine, which would be prohibitively expensive. Thus, a much better approach would be use edge connectivity information to find the ordering of sample points, prior to the optimization procedure. This would yield significantly more robust skeletonization.