Quick Kernel Ball Region Approximation for Improved Laplace Smoothing

By Jean Cabello

Meshing & Abstraction Group
Digital Simulation Solutions
Siemens PLM Software
2000 Eastman Dr., Milford, Ohio 45244 USA

Abstract. Instead of using the polygon defined by adjacent vertices to a vertex (called the ball) or its kernel [1], we propose a modified polygon that is easy to compute, convex and an approximation of the kernel. We call this polygon the "quick kernel ball region." This novel algorithm is presented in details. It is easy to implement and effective in constraining a vertex to remain within its feasible region, preventing element folding.

1 Introduction

It is well known that the application of the Laplacian smoothing technique can result in inverted or negative elements. The presence of even one element with negative jacobian will render most field solvers inoperable.

Many Laplace variants exist that try to prevent the creation of inverted elements in non-convex domains. The most commonly used is the so called "smart" Laplace smoothing [2] that checks after a node has been moved if any inverted elements have been created.

The kernel of a polygon is the locus of the points internal to the polygon from which all its vertices are visible [1]. For a convex polygon, all vertices are visible to each other and its kernel is the polygon itself. For concave polygons, the kernel is the intersection of half planes determined by the polygon's edges.

The present work proposes a simple and natural modification to the ball of a vertex that prevents the generation of inverted elements near concave boundaries. When Laplace smoothing is restricted to this modified polygon inverted elements will not be created during the smoothing process

2 Quick Kernel Ball Algorithm

2.1 Laplacian smoothing in 2D

Laplace smoothing [2, 4] computes a new position P^* of vertex P that is in a linear combination of the vertices of the polygon formed by it surrounding vertices. The new location will lie in the convex hull of the ball $B(\mathrm{P})$.

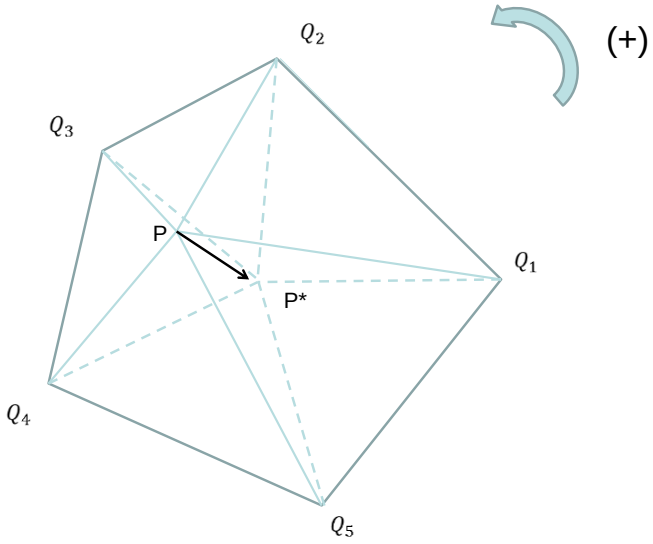

Fig. 1: $B(\mathrm{P})$ = union of surrounding elements. $P^*$is at the centroid.

The general formula for Laplace smoothing is:

$$P^* = \sum_{i=1}^{i=n} \omega_i \, Q_i \; ; \; \sum_{i=1}^{i=n} \omega_i \; = 1; \; \omega_i \geq 0 \qquad (1)$$

$$\begin{cases} P^* = P + \overrightarrow{\Delta P} \\ \overrightarrow{\Delta P} = \sum_{i=1}^{i=n} \omega_i \overrightarrow{PQ_i} \end{cases} \qquad (2)$$

2.2 Quick Kernel Ball Algorithm

Given a point P to move, we define B^0(P) = $\{Q_i$,i=1,n$\}$ the ball defined by the first neighbors to P. We assume that B^0(P) is a circular list oriented counter clockwise (P is to the left of the directed edge Q_iQ_{i+1}). The steps to compute the quick kernel ball qkB(P) are described below:

Do {

1. Find the most concave point in B^n(P) i.e. Q_{mcp}
2. If all the points in B^n(P) are convex (i.e. $Q_{mcp} = \{\emptyset\}$)
   a. qkB(P) = B^n (P).
   b. Exit.
3. else (i.e. $Q_{mcp} \neq \{\emptyset\}$)
   a. Remove from $B^n(P)$ both previous and next point to Q_{mcp}.
      B^{n+1} (P)= B^n(P)-{ Q_{mcp-1}, Q_{mcp+1}}
   b. Update the circular linked list of points.
   c. Recompute the new angles for Q_{mcp-1}, Q_{mcp}, Q_{mcp+1}

} until (B^{n+1}(P) is convex)

The advantage of our approach compared to the computational geometry computation of the kernel is that *no new vertices* need to be computed. qkB starts with a list of vertices that defines the ball of a point and retains a subset that *convexifies* the ball - The algorithm is constructive.

Figure 2 shows a comparison of the quick kernel Ball $\{Q_2, Q_5, Q_7, Q_8\}$ and the kernel $\{Q_2, I_{1256}, Q_5, I_{7845}, Q_8, I_{8123}\}$ for a simple polygon with eight vertices. QkB is almost included in the kernel except for the small triangular region $\Delta(I_{7845}, Q_5, Q_7)$ that is not visible from Q_4.

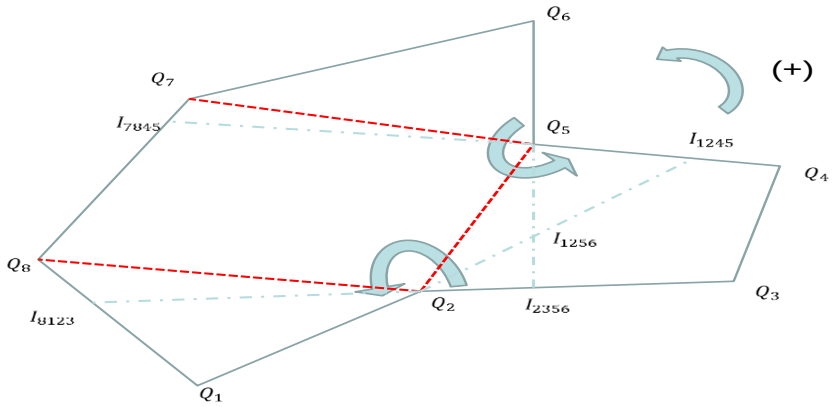

Fig. 2 : $qkB = \{Q_2, Q_5, Q_7, Q_8\}$ approximates the kernel

4 Results and Discussion

4.1 Parameters to Control the Iterative Laplace Smoother

The list of nodes that need to be updated is kept in a (C++) set. During, the first iteration the set is filled with all interior nodes. We define `maxSmoothingIteration` the maximum number of iterations. The stopping criterion at a node is chosen so that the node is not moved if the relative displacement $\frac{|\overrightarrow{\Delta P}|}{\bar{L}} < \varepsilon$ with $\bar{L} = \frac{1}{n}\left(\sum |\overrightarrow{PQ_i}|\right)$ representing a characteristic length defined as the average surrounding edges' length. When a node is not moved, it is removed from the set for the next iteration. When a node is moved, its coordinates are updated and its neighbor nodes are added to the set. The iteration stops when either the maximum number of iterations is reached or the set of nodes that need to be updated is empty.

4.2 Parameters to Control the Approximation of the Kernel

The reflex interior vertex angle threshold α_{thresh} is used to decide if a point is concave or not. The theoretical value for α_{thresh} is π. However, this parameter can be used to improve the stiffness of a mesh around highly concave regions. In our work, we found that a value of α_{thresh}=170 degrees led to good results.

4.3 Benchmark on a Non-Convex Domain

A framework was implemented to benchmark the reliability of the Laplace variants presented earlier as well as other smoothing strategies. In this framework, the convergence criteria remained the same for all the algorithms (`maxSmoothingIteration` =100, $\varepsilon = 0.01$). The only variable that changed was the method used to move a node.

For quadrilateral elements, initially we used the ball formed by the union of surrounding elements which, translated to triangular meshes, is equivalent to adding a fictitious diagonal edge centered at P. Our experience showed that the results improved if instead we used the enveloping polygon of the adjacent vertices to P as suggested in [7]. Figure 4 depicts the enveloping polygon $\{Q_1, Q_2, Q_3, Q_4\}$ of adjacent vertices to P in dashed

lines. Notice that, in general, the enveloping polygon of the adjacent vertices is contained in the union of surrounding elements (unless some quad elements have negative jacobians).

Figure 3a depicts the initial mesh that was used as a benchmark. An initial coarse structured mesh was constructed and the interior nodes perturbed so that no folded element exist and the mesh is valid. Figure 3b shows the converged mesh, after 12 iterations, using constant weight Laplace smoothing. The mesh is folded close to the concave boundary. Figure 3c shows the converged mesh, after 55 iterations using inverse distance Laplace. The mesh is valid but many elements exhibit high aspect ratio and are becoming almost degenerate. Figure 3d shows the mesh obtained modifying the surrounding polygon according to [6]. It converged after 40 iterations. Although the mesh is valid with no folded elements, they also exhibit high aspect ratio near the concave boundary. Figure 3e depicts the mesh using qkB, converged after 12 iterations. The mesh obtained preserved the mesh regularity and element did not get attracted and concentrated to the concave boundary. Finally, Figure 3f shows the results using Winslow smoothing [3] converged after 6 iterations. This is our reference mesh. Winslow smoothing does provide a better mesh than qkB on this structured example. However, our qkB algorithm is applicable to tri, quad and mixed meshes.

5 Conclusion

A new technique to compute an approximation of the kernel of a polygon was presented. The technique is intuitive and easy to implement. It works by deactivating vertices before and after the most concave point of a simple polygon, updating the new polygon and repeating until a convex sub-polygon is obtained. The algorithm removes *most of* the regions in the original polygon that are not visible to all other vertices and therefore provides an *approximation* to the kernel of the polygon. The technique was coined “Quick Kernel Ball” or qkB in short. Preliminary results are encouraging and demonstrate that starting with a valid mesh, the present algorithm will maintain a valid mesh.

The key ideas of our technique have the potential to be extended to three dimensions. The dihedral angle of an edge is the extension to the interior angle at a vertex. The three dimensional ball at a point P is convex if, and only if, all the dihedral angles to the opposite edges to P have an angle that is lower than π. Once some edges have been classified as concave, it makes sense to start with the edge with the largest dihedral angle and drop

the two vertices on both sides of the edge as they are not visible from each other. The challenge is then to reconnect the triangles in the outer boundary to form a closed ball. Edge collapse or re-triangulation could be used to this end.

6 References

1. D. Lee and F. Preparata, "An Optimal Algorithm for Finding the Kernel of a Polygon", *J. ACM*, Vol. 26, No 3, pp 415-421, July (1979).
2. S. Canann, J. Tristano and M. Staten, "An Approach to Combined Laplacian and Optimization Based Smoothing for Triangular, Quadrilateral and Quad-Dominant Meshes", *7th IMR*, (1998).
3. P. Knupp, "Winslow Smoothing on two-dimensional unstructured meshes", *proc. 7th IMR*, Detroit, MI, pp. 449-457, (1998).
4. D. Field. "Laplacian Smoothing and Delaunay Triangulation", *Communication in Applied Numerical methods*, vol. 4, pp. 709-712, (1988).
5. D. Ives, "Unstructured Boundary Layer Grid Generation", Intl. Conf. Num. Grid Generation in Comp. Fields, pp 13-19, (2000).
6. P. Liakopoulos and K. Giannakoglou, "Unstructured Remeshing Using an Efficient Smoothing Scheme", *Eccomas CFD*, (2006).
7. N. Calvo and S. Idelsohn, "All Hexahedral Mesh Smoothing With a Node Based Measure of Quality", *IJNME 50(8)*, pp1957-1967, (2001).

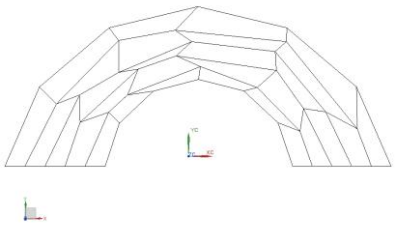

Fig. 3a: Initial mesh perturbed

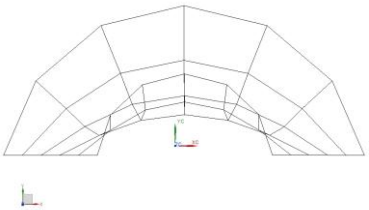

Fig. 3b: Standard Laplace

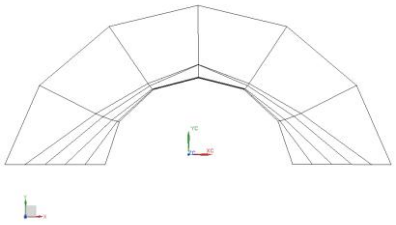

Fig. 3c: Laplace inverse distance

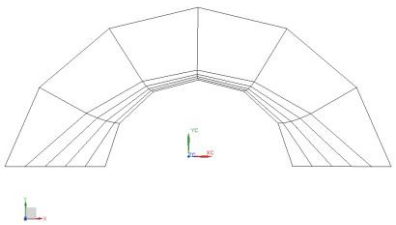

Fig. 3d: Inside feasible region

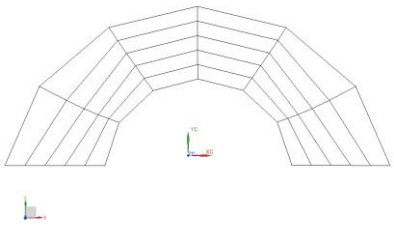

Fig. 3e: qkB + Laplace

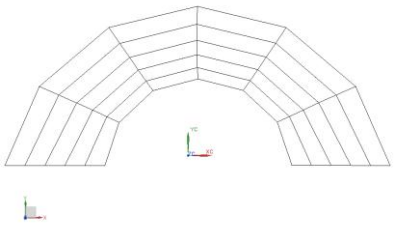

Fig. 3f : Winslow Smoothing

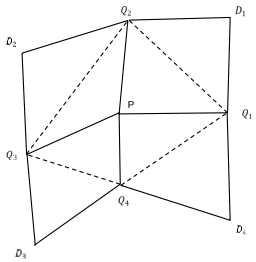

Fig. 4: Quad elements. Enveloping polygon of adjacent vertices to P.