AIAA SciTech
4-8 January 2016, San Diego, California, USA
54th AIAA Aerospace Sciences Meeting

Optimization-Based Smoothing for Extruded Meshes

Steve L. Karman*
Michael G. Remotigue†
Pointwise, Inc.

A mesh smoothing method is implemented in a viscous layer mesh extrusion process for mixed surface domains containing triangles and quadrilaterals. As each layer is created the smoothing method ensures high quality elements with proper normal spacing. The smoothing is an extension of an optimization-based node perturbation technique that uses a cost function to enforce desired element shapes and layer height. Details of the smoothing method are described. Several three-dimensional examples are included that demonstrate the effectiveness of the method to produce high quality viscous layer meshes.

Nomenclature

[A]	=	Jacobian matrix for condition number
C, C_e	=	Cost function
h	=	Local marching distance
WCN	=	Weighted condition number
$\hat{J}$	=	Normalized Jacobian
$\vec{p}_n$	=	Perturbation vector for node n
$\vec{s}_n$	=	Sensitivity vector for node n
[W]	=	Weight matrix for weighted condition number
X, Y, Z	=	Cartesian physical coordinates
U_k, V_k, W_k	=	Edge vectors at a corner

I. Introduction

Numerical simulations of viscous flows require computational meshes that properly resolve the boundary layer region near no-slip surfaces. Traditional structured meshes can be created for most problems with good near-wall resolution using advanced algebraic methods, such as Trans-Finite Interpolation (TFI) [1]. Additional control can be achieved by employing elliptic Partial Differential Equation (PDE) smoothing methods, such as Winslow smoothing, that include forcing functions to enforce adequate boundary layer resolution [2]. There are also hyperbolic PDE methods with similar spacing control for growing structured meshes off the viscous surfaces [3-7]. These methods can easily create meshes for convex shapes that extend from the surface to the outer boundary. Concave surfaces are more difficult to mesh due to colliding layers extruding from corners. In severe cases this prevents the hyperbolic scheme from reaching the desired distance away from the surface

Semi-structured methods have been developed that combine the strengths of a parabolic PDE smoothing method with the flexibility of multiple element types [8, 9]. The grids are structured

* AIAA Associate Fellow
† AIAA Senior Member

in the layer where the parabolic smoothing equations are enforced, but the topology can change between the layers. Adding or removing elements from layer to layer alleviates the collision issues associated with fully structured hyperbolic schemes.

Fully unstructured mesh generation methods do not have TFI methods or robust Winslow smoothing methods that can be used to create computational meshes with adequately resolved boundary layer regions. Linear-Elastic PDE smoothing has been used to insert viscous layers into an existing volume mesh [10]. The number of inserted layers and spacing control can be restrictive in severely concave regions. An alternate approach uses an advancing front scheme to create highly-resolved, near wall tetrahedral meshes that are combined with an isotropic tetrahedral far-field mesh [11, 12].

Marching methods evolve the viscous mesh layer by layer by extruding new points in the boundary normal direction. This near wall mesh is combined with some form of volume mesh, typically an isotropic tetrahedral mesh. The viscous layer element type is dependent on the surface mesh type. Triangle surface elements extrude to become triangular prisms. Quadrilateral surface elements extrude to become hexahedra. Polygonal surface meshes would extrude into more general polyhedral elements, but those cases are not considered in this paper. The normal distance for each layer is defined by a distribution function, such as a geometric progression where each new layer is a fraction taller than the previous layer. The boundary surface shape dictates the normal direction for marching. Simple convex surfaces are easily handled using a static normal direction for each surface node. Concave shapes can cause element quality issues as the marching layers extrude out of corners. This necessitates the use of smoothing methods to prevent or delay the creation of poor quality elements.

Smoothing of these extruded layers has some key requirements that make it different from traditional unstructured mesh smoothing. The element quality is obviously important, but the shape of the elements is key. The desired angles and edge lengths for triangular prisms are different than the angles and edges for hexahedra. The layer height is just as important, as the goal is to resolve a viscous boundary layer. The smoothing scheme must maintain good element shape and enforce desired normal distribution through the layers.

This paper describes a mesh-smoothing scheme applied to the unstructured mesh extrusion process for mixed surface domains containing triangles and quadrilaterals. The smoothing is an extension of an optimization-based mesh smoothing method that uses a cost function to enforce desired element shapes and layer height. The layer extrusion process will be briefly outlined, followed by a more details description of the optimization-based smoothing scheme extended to support mixed prismatic/hexahedral extruded meshes. Results are shown for several complex 3D configurations. Initial surface meshes were generated using Pointwise [13].

II. Viscous Layer Extrusion

The viscous extrusion process defined below assumes the number of elements per layer is constant. The alternative is to allow a stack of elements to terminate when collisions or poor quality arises while neighboring element stacks may continue to march. Since the focus of this paper is on the smoothing method the assumption of full layers is maintained.

The subset of triangular prisms, shown in Figure 1, will be used to illustrate the extrusion and smoothing process. The black triangles represent the surface tessellation of a viscous boundary. Performing a weighted average of the normal vectors for the surrounding triangles creates normal vectors for each surface node. The weighting is typically area or included angle. In some cases the viscous surface is adjacent to other boundaries, such as symmetry planes, so where

needed, normal vectors are adjusted to become tangent with the adjacent boundary or boundaries.

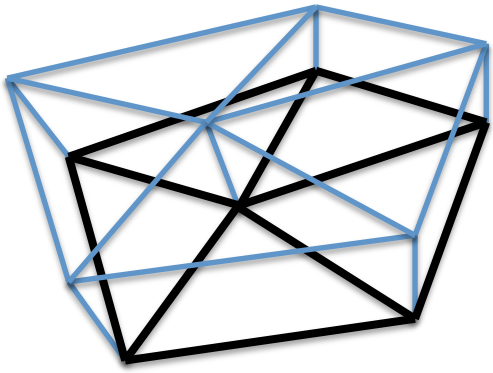

Figure 1. Collection of triangular prisms elements near surface.

The prisms, shown in blue, are created by advancing the nodes in the direction of the computed normal vector the prescribed distance for the layer. For planar or convex boundaries the computed normal direction is usually sufficient. Concave and highly curved convex boundaries may require mesh smoothing to maintain element quality as the viscous layers evolve. The smoothing scheme must respect the layer height to ensure adequate resolution of the viscous region.

III. Optimization-Based Mesh Smoothing

Optimization-based mesh smoothing seeks to optimize a cost function related to element quality. The use of element condition number as the quality measure was published by Freitag and Knupp [14, 15]. The approach described in this paper is an extension of a method developed for adaptive smoothing [16]. A cost function is defined for each corner based on the status of the corner. If the corner normalized Jacobian, $\hat{J}$, is negative or zero the cost function will be equal to the Jacobian. If the corner is not inverted the cost is based on the weighted condition number, WCN, and will be greater than zero. The reason for two formulations is because the weighted condition number does not recognize when the corner is inverted. The unification of the two states allows the optimizer to handle inverted corners in a more natural way. The two formulae are shown in Equation (1). This formulation is C0 continuous at the value of zero.

$$\begin{aligned} C &= \hat{J} \quad \text{if } \hat{J} <= 0.0 \\ C &= \frac{1}{WCN} \quad \text{if } \hat{J} > 0.0 \end{aligned} \tag{1}$$

The normalized Jacobian is defined as the magnitude of the determinant of the Jacobian matrix where the edge vectors have been normalized and inserted into the columns of the matrix, defined in Equation (2). The edge vectors are shown in Figure 2.

$$\hat{J} = \begin{vmatrix} \hat{E}_x & \hat{F}_x & \hat{G}_x \\ \hat{E}_y & \hat{F}_y & \hat{G}_y \\ \hat{E}_z & \hat{F}_z & \hat{G}_z \end{vmatrix} \tag{2}$$

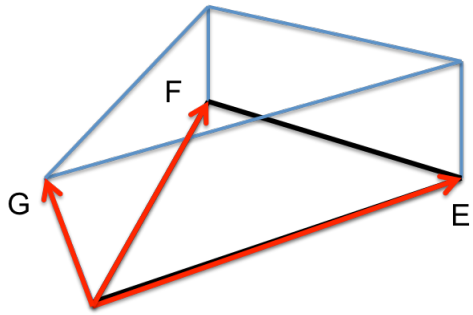

Figure 2. Edge vectors for prism corner.

The second form of the cost function is based on the weighted condition number, WCN, given in Equation (3).

$$WCN = \frac{\left\|AW^{-1}\right\|\left\|WA^{-1}\right\|}{3} \tag{3}$$

The bracketed quantities are the Frobenius norms of the matrix products. The [A] matrix is the Jacobian matrix defined in Equation (4). The columns of [A] are the three edge vectors emanating from a corner of the element, shown in Figure 2 for a prism. This element is taken from the upper right region of the collection of element in Figure 1.

$$A = \begin{bmatrix} E_x & F_x & G_x \\ E_y & F_y & G_y \\ E_z & F_z & G_z \end{bmatrix} \tag{4}$$

The [W] matrix is the weight matrix that transforms the reference corner (a right-angled corner with unit length edges) to the desired corner shape. This is depicted in Figure 3 where a regular corner is transformed to a desired shape. The angle θ and the edge lengths U, V and W are free parameters used to define the shape of the corner. For extruded meshes the height parameter, W, would be significantly smaller than U and V, especially for the initial layers. The parameters θ, U and V define the shape of the base surface.

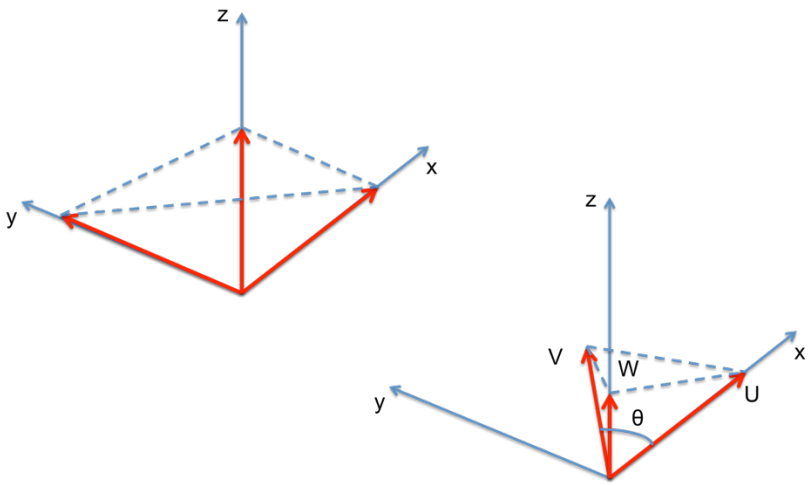

Figure 3. W matrix transforms "regular" corner shape (upper left) to desired corner shape (lower right).

The [W] matrix for a prism corner is defined in Equation (5). The magnitudes of the edges describing the weight matrix are relative lengths. The same is true for the [A] matrix. In computing the weighted condition number the scaling of the Frobenius norms of [A] and [W] are cancelled by the scaling of the norms of the inverse of the matrices. In other words, the condition number is scale invariant. In the context of extruded mesh generation it is simply easier to use known or desired physical heights and edge lengths in the construction of [A] and [W].

$$W = \begin{bmatrix} |\vec{U}| & |\vec{V}|\cos\theta & 0 \\ 0 & |\vec{V}|\sin\theta & 0 \\ 0 & 0 & |\vec{W}| \end{bmatrix} \tag{5}$$

The value of the weighted condition number will be unity when the actual shape of the corner matches the defined shape. The cost will vary from one for perfectly shaped corner and approach zero for a collapsed corner. At that point the normalized Jacobian formula will take over and the cost will decrease with a minimum value of negative one as the lower limit.

Simplex elements (triangles in 2D and tetrahedra in 3D) require a single weight matrix to define the shape. Prisms and hexahedra require a weight matrix for each corner. Figure 4 shows the shape defining parameters overlaid on the base of the example prism. The parameters θ, U and V define the actual shape of the base surface since those base nodes cannot be moved in the current marching layer. The W vector is defined to be perpendicular to the base with a length equal to the layer marching distance. It is apparent that the vertical edge does not exactly line up with the W vector from the base. This will result in a cost less than one for this corner.

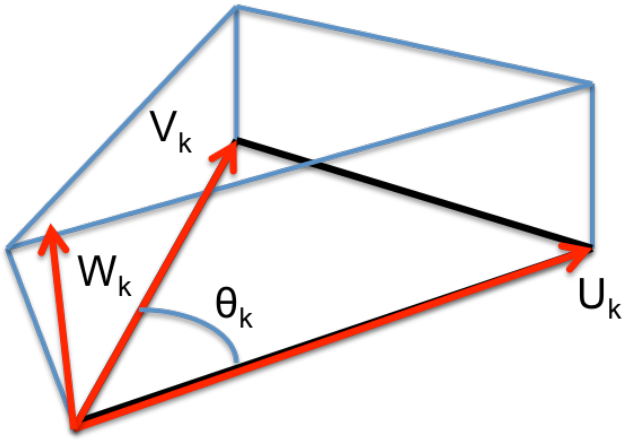

Figure 4. Shape defining edge vectors overlaid on prism base.

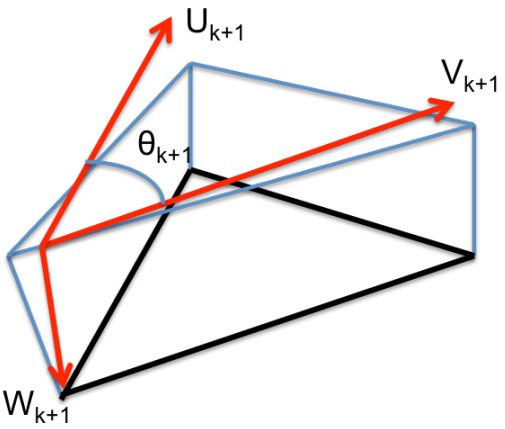

Figure 5. Shape defining edge vectors for top face of prism.

The corresponding shape defining vectors for the top corner are shown in Figure 5. In this view, it is apparent the actual edge is not orthogonal at the top corner compared to the shaped defining edges. Once again, the cost of this corner will be less than one. Notice the orientation of the shape defining vectors has been flipped for the top layer compared to the bottom layer. The formula for the [W] matrix is still Equation (5) with different values of U, V, W and θ. A different set of shape vectors can be defined for each corner of the element. The W parameter is the defined normal marching distance. The parameters θ, U and V are defined to influence the evolution of the mesh towards an "optimal" element shape. For this study, the lengths of the shape vectors in the top layer are equal to the average length of the base vectors. The angle used in the top layer is defined as $\pi/3$ for triangular prism elements. This set of parameters seeks to make the top triangular face an equilateral triangle, but respects the height and alignment of the vertical edge through the cost function.

The bottom nodes in the current layer are stationary. The top nodes of the layer are perturbed in the mesh-smoothing scheme. The direction of the perturbation is consistent with the sensitivity of the cost function with respect to each top node of the element. The method for computing the sensitivities uses C++ function overloading to compute the function value and the nine derivative quantities corresponding to the X, Y and Z sensitivities for the top three nodes.[17, 18] A cost function call is required for each corner of the element. The function call returns the cost for the corner and the sensitivities for each of the three top corner coordinates. Each triangular prism element will require 6 cost function calls. Calls for the bottom three corners will produce a sensitivity vector for the single top node for that corner. Calls for the top three corners will produce sensitivity vectors for each top node.

Hexahedral layer elements are processed in a slightly different manner. Figure 6 shows a collection of hexahedral elements for a marching layer. The black quadrilaterals represent the surface elements. The extruded hexahedral elements are shown in blue.

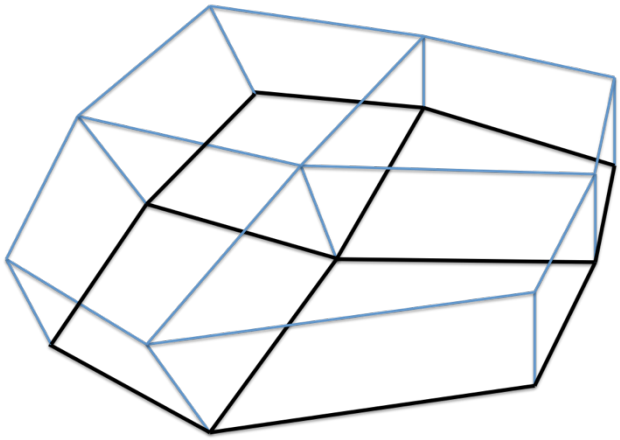

Figure 6. Collection of hexahedral elements near surface.

The defining shape vectors for a corner in the base of an example hexahedral element are shown in Figure 7. The parameters θ, U and V define the actual shape of the base surface since those base nodes cannot be moved in the current marching layer. The W vector is defined to be perpendicular to the base at that corner with a length equal to the layer marching distance. The weight matrix is produced using Equation (5). It is apparent that the vertical edge does not exactly line up with the W vector from the base. This will result in a cost less than one for this corner.

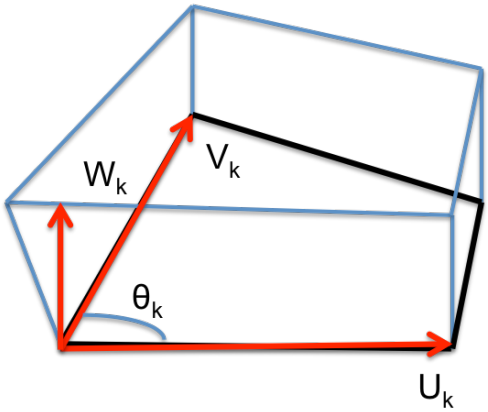

Figure 7. Shape defining vectors overlaid on hexahedral base.

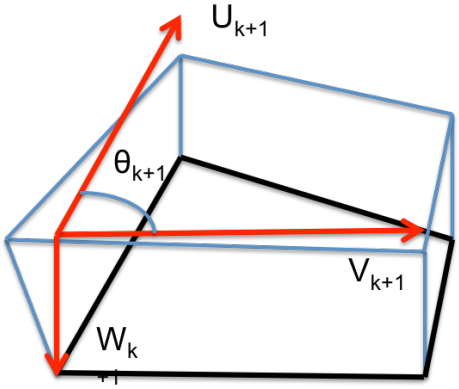

Figure 8. Shape defining vectors for top face of hexahedral element.

The corresponding shape defining vectors for the top corner are shown in Figure 8. The W vector is the defined marching distance oriented in the reverse normal direction. The parameters θ, U and V are defined to influence the evolution of the mesh towards an "optimal" element shape. For this study, the lengths of the shape vectors in the top layer are equal to the average length of the base vectors. The angle used in the top layer is defined as π/2 for hexahedral elements. The [W] matrix for top nodes of hexahedra is shown in Equation (6). This set of parameters seeks to make the top face a square, but still respecting the height and alignment of the vertical edge. So in each case, prisms and hexahedra, the defining shape parameters seek to produce a unit aspect ratio top face with the nodes positioned at the desired height directly above the base corresponding base node.

$$W = \begin{bmatrix} |\vec{U}| & 0 & 0 \\ 0 & |\vec{V}| & 0 \\ 0 & 0 & |\vec{W}| \end{bmatrix} \tag{6}$$

A nodal perturbation vector for each top node in the marching front is constructed by combining the sensitivity vectors for that node computed during the cost function calculations. During the function calls the average cost is computed and the minimum (worst) cost is stored for each top node. In addition, the average sensitivity vector is computed and the minimum sensitivity vector corresponding to the minimum cost is stored. A blending of the average and minimum cost and sensitivity vectors is used to compute the final cost value and perturbation direction for a node. The node perturbation vector is computed using the formulae in Equation (7). The multiplying factor, MF, is plotted in Figure 9 as the vertical axis. The minimum cost value, C_{min}, is plotted along the bottom right axis and ranges from zero to one; negative values are clipped at zero. The blending exponent P is plotted along the bottom left axis and ranges from zero to two, the defined limit in this work.

$$\begin{aligned} \vec{P}_{avg} &= \frac{1}{nj}\sum_{j=1}^{nj}\left(\vec{s}_n\right)_j \\ MF &= Max(0,1-C_{\min})^P \\ AF &= 1-MF \\ C_{node} &= MF * C_{\min} + AF * C_{avg} \\ \vec{P}_{node} &= MF * \vec{P}_{\min} + AF * \vec{P}_{avg} \end{aligned} \tag{7}$$

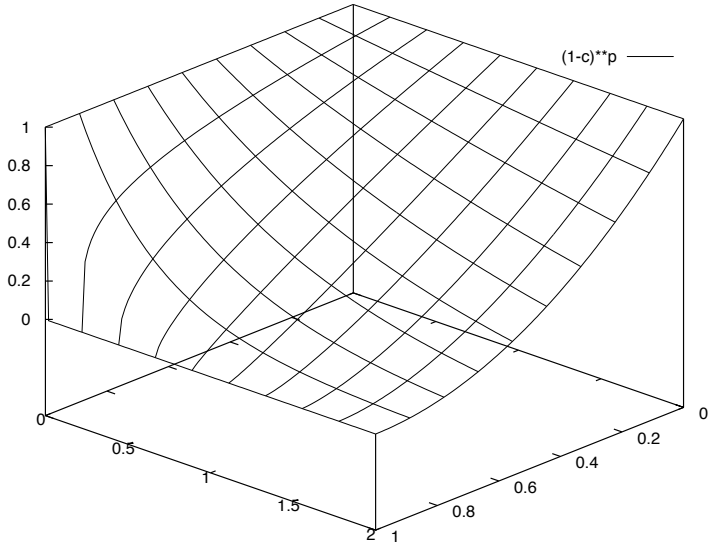

Figure 9. Carpet plot of minimum cost blending factor, MF.

When C_{min} is less than or equal to zero or P is equal to zero then the multiplying factor, MF, equals one and the average factor, AF, equals zero. This corresponds to the back left and back right limits in the plot. In this case, the perturbation vector is the sensitivity vector from the worst

cost value. If P equals one then there is a linear relationship between C_{min} and MF. Values of P less than one give more influence on the minimum quantities and values of P greater than one give more influence on the average quantities. C_{min} always has an influence for any value of P, except at the limit point when the minimum cost and average cost are equal to the ideal value of one.

Each mesh smoothing iteration of a given layer moves the top nodes according to Equation (8). The perturbation amount is restricted to a fraction of the local marching distance, h. This distance is also the magnitude of the W vector at the node. The relaxation amount, ω, is typically 50%. The magnitude of the perturbation vector is also limited to one or less by the denominator to prevent overshoots at nodes with large magnitude perturbation vectors. As the nodal cost approaches one the perturbation vector magnitude tends to diminish. The number of iterations is user specified, typically between 5-50, depending on the complexity of the marching surface.

$$\vec{X}_{node}^{i+1} = \vec{X}_{node}^{i} + \vec{p}_{node} \frac{h\omega}{MAX(1.0, |\vec{p}_{node}|)} \tag{8}$$

IV. Results

The following cases begin with a defined surface tessellation, generated by Pointwise. Some cases are self-contained, closed shell geometries. Other cases have boundaries on the surface tessellations that are planar and require additional control of the extrusion process to ensure the resulting viscous layer meshes reside on those adjacent boundaries.

A. Block Cross Extruded to Sphere

The first case is a closed surface created by assembling multiple unit-sized, cube-shaped blocks connected to form a three-dimensional cross. This case is used to illustrate the robustness of the method for a variety of surface types. The first mesh was created from surface meshes that are structured domains, shown in Figure 10. This case could be extruded using a hyperbolic PDE method, which would create a multiple-block structured three-dimensional mesh. The optimization-based smoothing method was applied to this case, which creates a single block of unstructured hexahedra. Each mesh created for this configuration used the same initial normal spacing equal to 0.01 with a geometric progression growth rate of 1.05 and a corner factor of 10 (all parameters used by Pointwise). The number of smoothing iterations was set at 200. The number of layers extruded was 100.

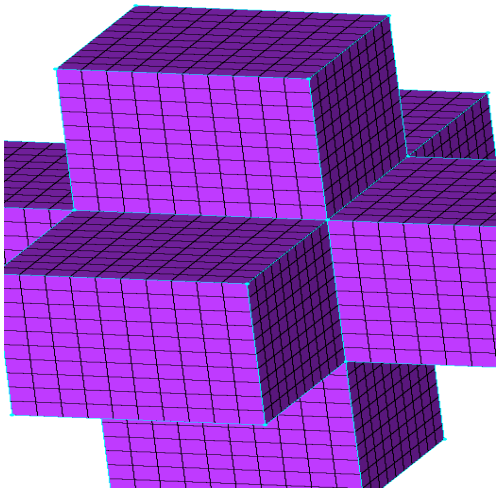

Figure 10. Quadrilateral surface meshes for 3D cross configuration.

After completing the 100 layers of extrusion the outer surface approaches a spherical shape, shown in Figure 11. The exponent P was specified as two, which combines the worst cost and sensitivity vector with the average cost and sensitivity vector and favors the average values more. A crinkle cut surface through X=0 is shown in Figure 12. The element shapes are approaching equilateral hexahedra.

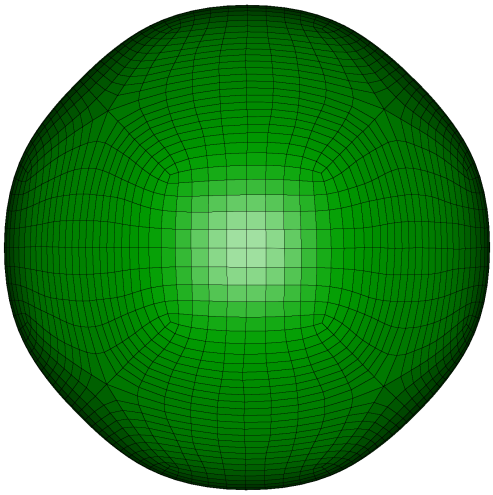

Figure 11. Outer layer for P=2 all quadrilateral surface mesh case.

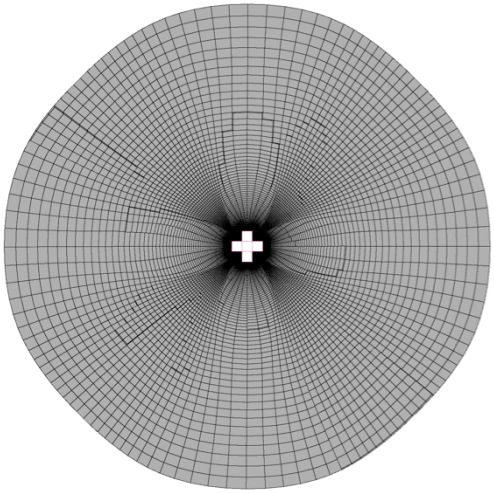

Figure 12. X-cut through extruded mesh for P=2 all quadrilateral surface mesh case.

Zoomed in views of the X=0 cut for different values of P are shown in Figure 13. The upper left image shows the result from using the hyperbolic mesh generation tool in Pointwise. The other images show the optimization-based smoothing extrusion results for P=0, 1, and 2. The hyperbolic result is smoother, as smoothness is part of the defining governing PDE. The P=0 optimized result shows the highest level of spreading of the elements out of the concave corners. The P=2 result is smooth, but has the least amount of spreading from the corners. Setting the exponent P to one seems to be a good compromise. It is interesting to note that if the shape of the elements from the hyperbolic result were used to compute the [W] matrices then the optimization

method would reproduce the hyperbolic result. Obviously for this quadrilateral surface mesh the hyperbolic scheme is preferred.

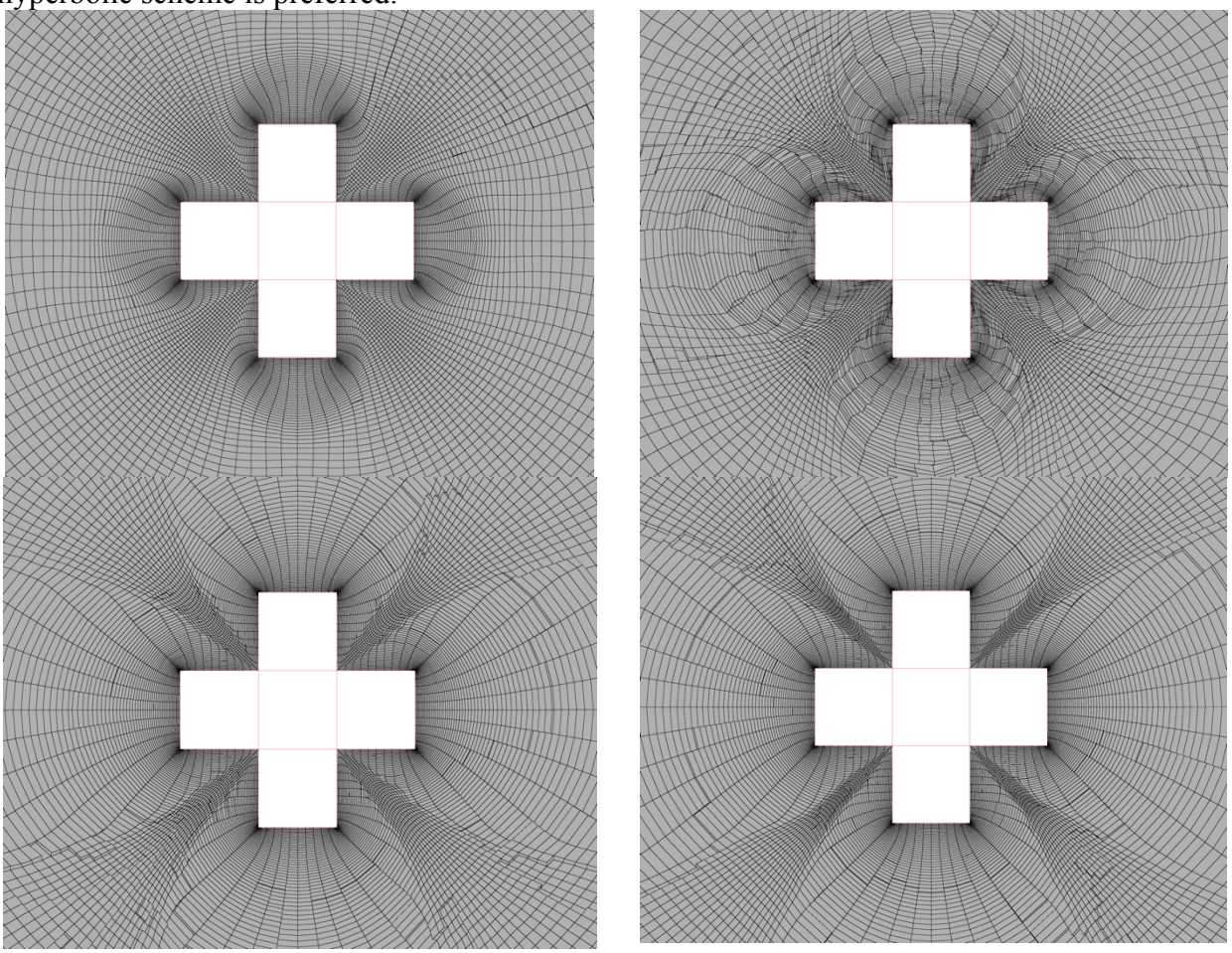

Figure 13. X=0 crinkle cut surfaces for hyperbolic (upper left), P=0 (upper right), P=1 (lower left) and P=2 (lower right).

The surface mesh was recreated using all triangles, shown in Figure 14. This case was computed with P=1. The resulting outer layer surface is shown in Figure 15. A crinkle cut through X=0, Figure 16, reveals the prisms approaching a uniform shape at the outer boundary. A magnified view of the X=0 crinkle cut is shown in Figure 17.

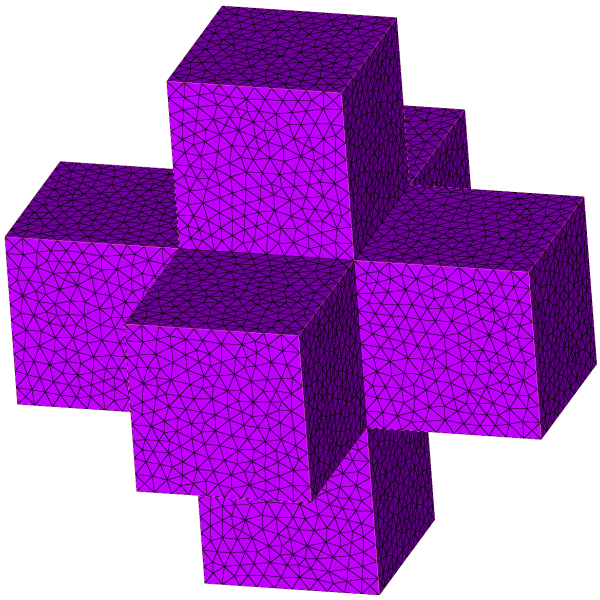

Figure 14. Triangular surface mesh for 3D cross configuration.

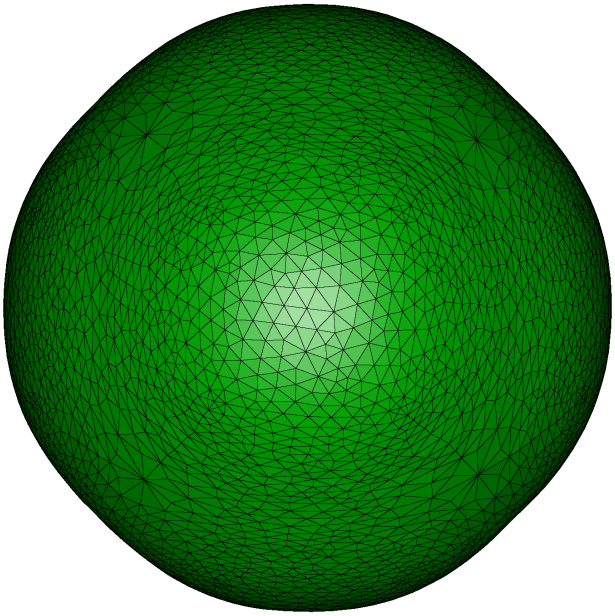

Figure 15. Outer layer for P=1 all triangular surface mesh case.

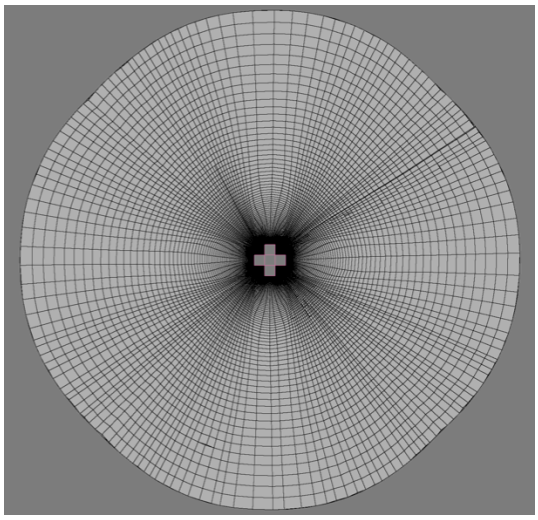

Figure 16. X=0 crinkle cut through P=1 triangular surface mesh case.

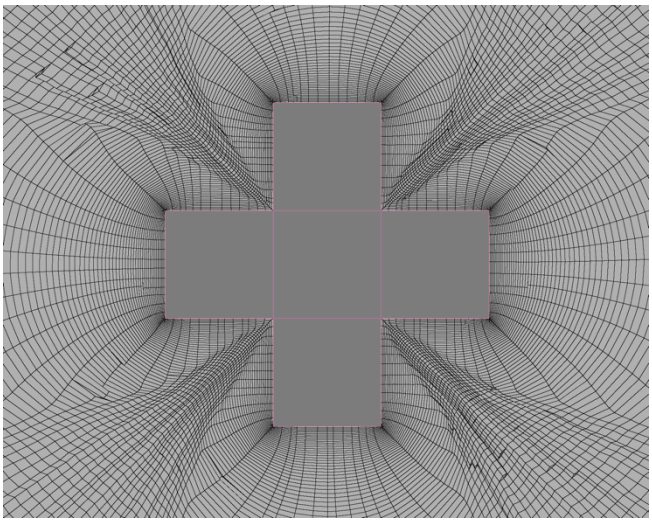

Figure 17. Magnified view X=0 crinkle cut through P=1 triangular surface mesh case.

The hybrid surface mesh, shown in Figure 18, consists of structured domains, unstructured triangular domains and quadrilateral-dominant hybrid domains. Lobes of the cross in the X directions are structured quadrilateral domains. The lobe in the negative Y direction consists of Delaunay unstructured domains. The lobe in the positive Y direction consists of Advancing Front unstructured domains. And the lobes in the Z directions are quad-dominant domains. This case was run with P=1. X, Y and Z crinkle cuts through the center are shown in Figure 19 through Figure 21, respectively.

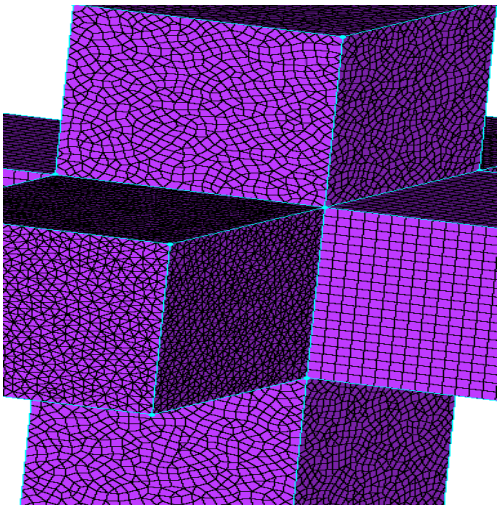

Figure 18. Hybrid surface meshes for 3D cross configuration.

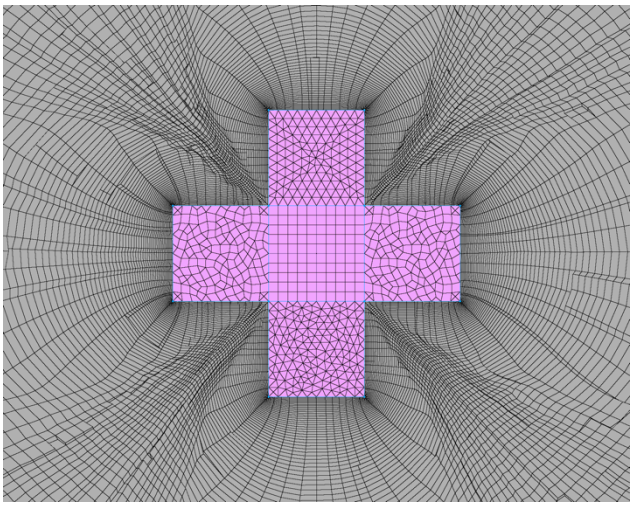

Figure 19. X=0 crinkle cut through P=1 hybrid surface mesh case.

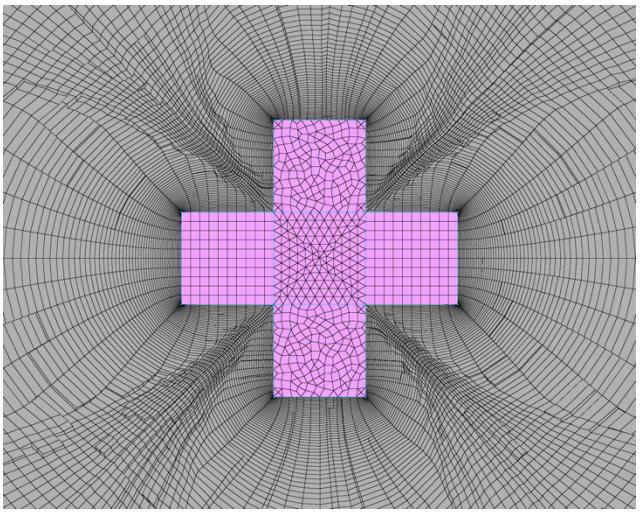

Figure 20. Y=0 crinkle cut through P=1 hybrid surface mesh case.

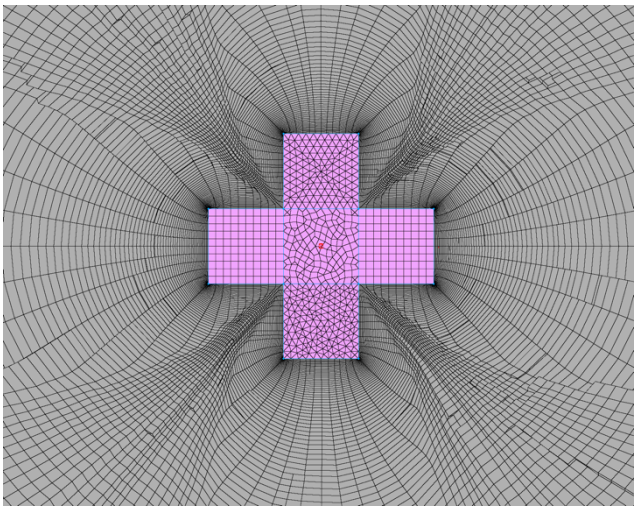

Figure 21. Z=0 crinkle cut through P=1 hybrid surface mesh case.

Elements grown off of the structured domains tend to be the smoothest. Elements grown from the triangular tend to be slightly smoother than elements grown from the quad-dominant domains.

B. Onera M6 Wing

The Onera M6 wing is a well-known CFD validation case [19, 20]. The geometry is a simple low aspect ratio wing with mild leading and trailing edge sweep, shown in Figure 22. The wing tip is modeled with a smooth rounded cap. The trailing edge is sharp. A symmetry plane, not shown, is modeled as a planar surface at the wing root. The surface mesh, shown in Figure 23, consists of triangles and quadrilaterals. The quadrilaterals were generated as structured patches at the leading and trailing edges to control clustering. The combined hybrid surface mesh is extruded to create a viscous layer block near the surface.

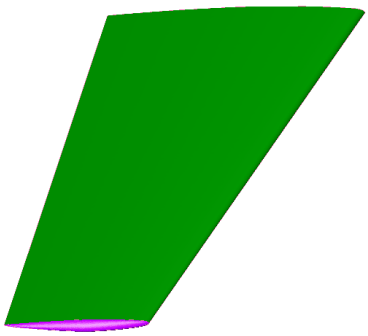

Figure 22. Onera M6 wing geometry.

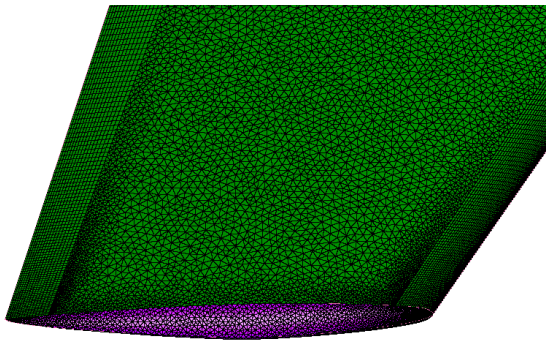

Figure 23. Hybrid surface mesh near wing tip.

The initial spacing was 0.0001 with a geometric progression factor equal to 1.1. A total of 40 layers were extruded with 50 smoothing sweeps per layer. The top layer mesh for a blending exponent equal to 0 is shown in Figure 24. The top layer mesh for a blending exponent equal to 2 is shown in Figure 25. A blending exponent equal to zero will perturb each node to reduce the worst cost element connected to that node. A higher blending exponent will attribute more influence to the average cost and sensitivity vector. The difference in this case is apparent near the trailing edge of the mesh. The final surface mesh elements are more isotropic for the blending exponent equal to zero.

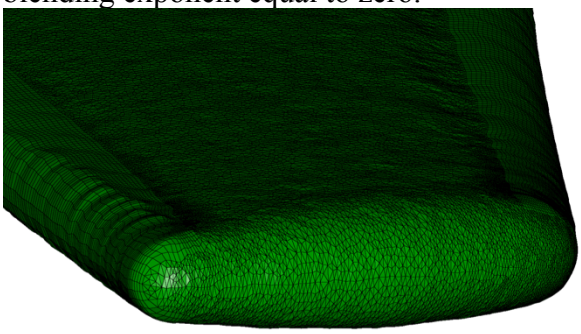

Figure 24. Top layer mesh for blending exponent P =0.

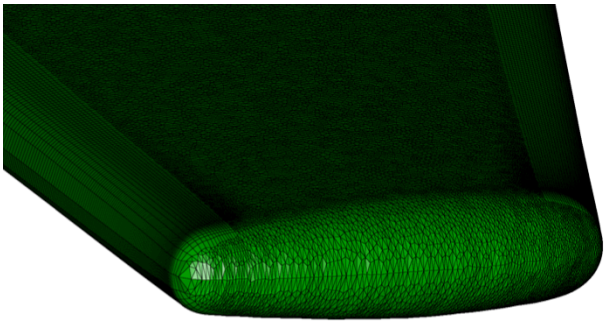

Figure 25. Top layer mesh for blending exponent P = 2.

The symmetry plane mesh is shown in Figure 26 for the blending exponent P=0. This geometry has a mild sweep angle with respect to the symmetry plane. The extrusion process wants to create layers that extend away from the symmetry plane. The smoothing boundary conditions prevent this by forcing the sensitivity vectors to lie in the plane. This influences all marching nodes on the symmetry plane and those directly connected to it. For P=0 this is manifested in some distortion of elements near the leading edge shown in the image. Two other results for P=1 and P=2 are shown in Figure 27 and Figure 28, respectively. Both results are much smoother than the P=0 case.

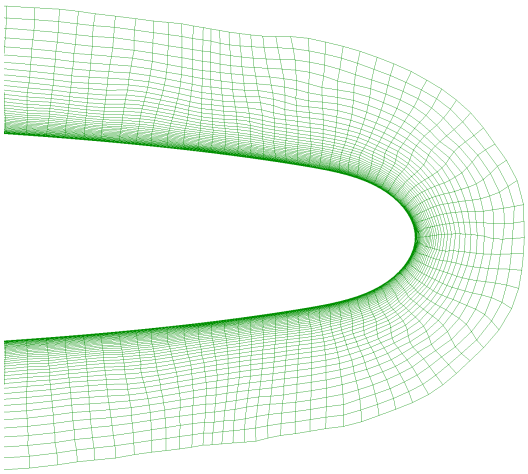

Figure 26. Symmetry plane mesh at leading edge for blending exponent P=0.

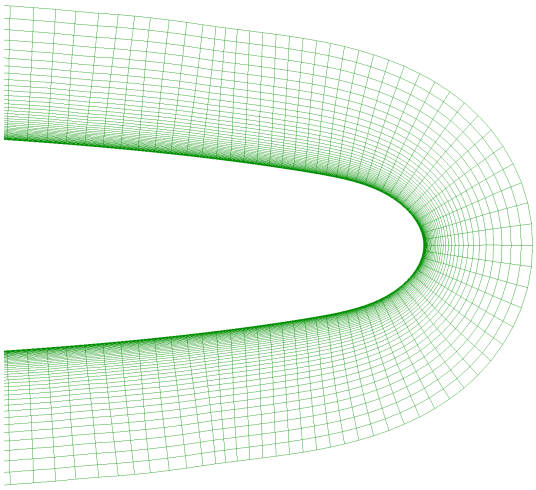

Figure 27. Symmetry plane mesh at leading edge for blending exponent P=1.

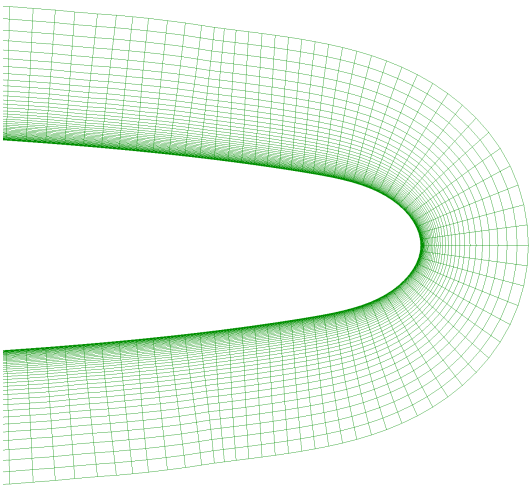

Figure 28. Symmetry plane mesh at leading edge for blending exponent P=2.

Similar plots for the trailing edge are shown in Figure 29 through Figure 31. In this case the P=0 result seems to produce some distortion ahead of the trailing edge, but better resolution of the region just behind the trailing edge. The P=2 result spreads the elements excessively. The P=1 result is a good compromise for both leading edge and trailing edge regions.

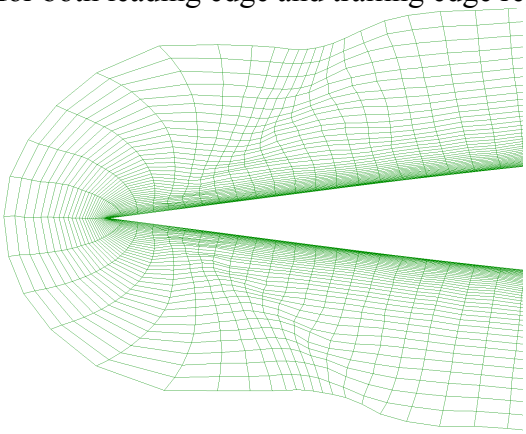

Figure 29. Symmetry plane mesh at trailing edge for blending exponent P=0.

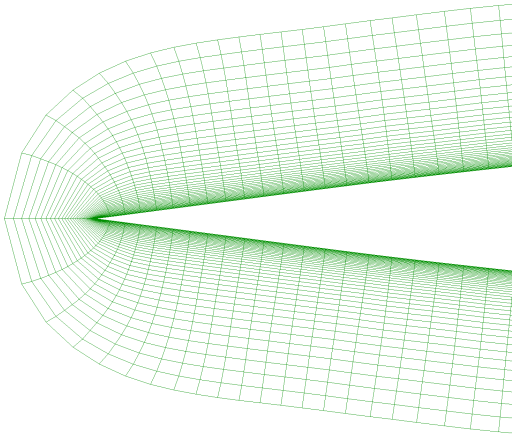

Figure 30. Symmetry plane mesh at trailing edge for blending exponent P=1.

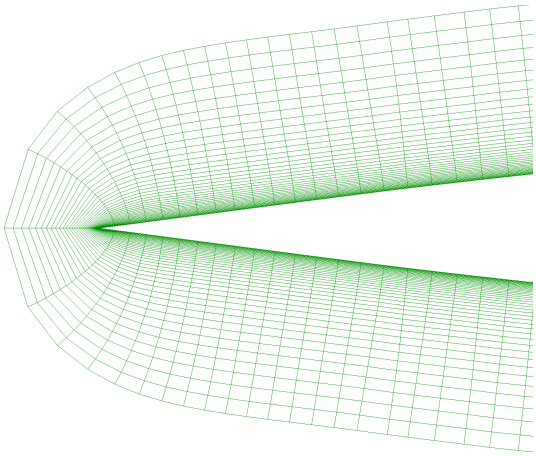

Figure 31. Symmetry plane mesh at trailing edge for blending exponent P=2.

C. Eglin Store

The last case is a popular validation case for moving body problems, the Eglin mutual interference experiment involving a wing with a pylon and finned store [21]. Only the store is meshed in this paper. The concave regions of the fins require the use of smoothing to create an extruded mesh. The geometry is shown in Figure 32. The surface mesh consisted of 62,690 triangles with some clustering to the leading edges of the fins. A series of extruded meshes were created with three different blending exponents, P. Each mesh used the same initial spacing equal to 0.001 with a geometric progression growth rate of 1.1 and a corner factor of 10. The number of iterations was set to 100 and the number of extruded layers was defined as 25. Figure 33 shows the 25th layer for P=0. All layers passed the quality constraints that could lead to premature termination of the extrusion process. Figure 34 shows the top layer for P=1 case. Again all layers passed the quality constraints. Less rounding effect is visible for the sharp edges on the fins and base region. Figure 35 shows the top layer for P=2 case. The last layer completed, but the code warned that nine elements were positive skewed. The problem elements were located near the trailing edges of the fins. That indicates that a corner of a prism was inverted.

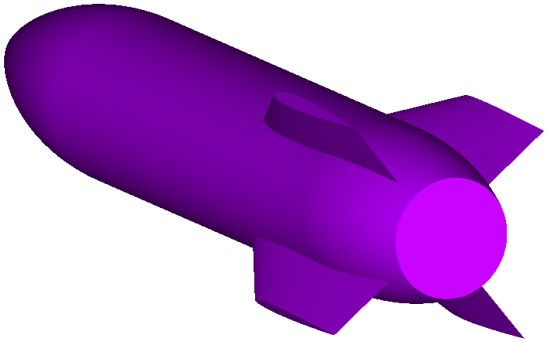

Figure 32. Store geometry with four fins.

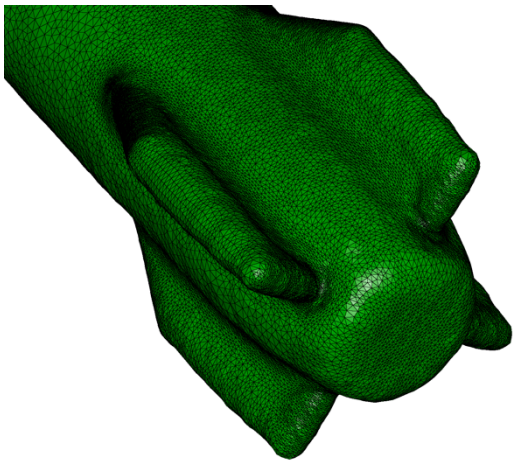

Figure 33. Top layer for blending exponent of 0.

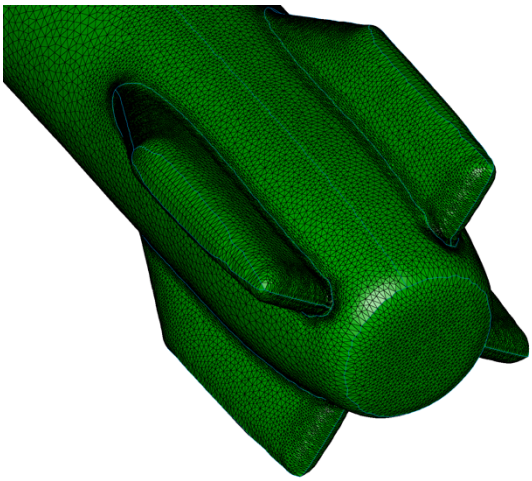

Figure 34. Top layer for blending exponent of 1.

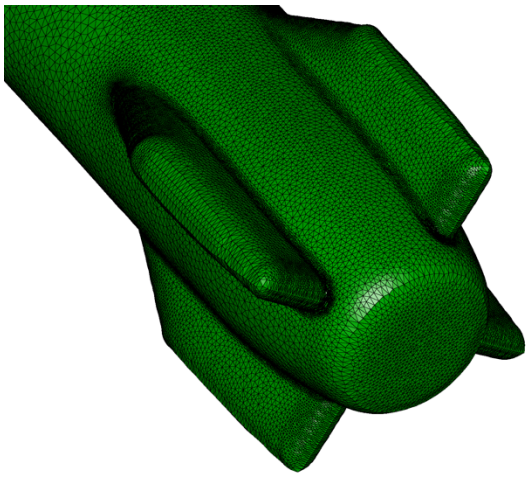

Figure 35. Top layer for blending exponent of 2.

A crinkle cut through the meshes at X=0.35 is shown Figure 36. The P=2 result is the smoothest, but has less turning of the elements out of the corners.

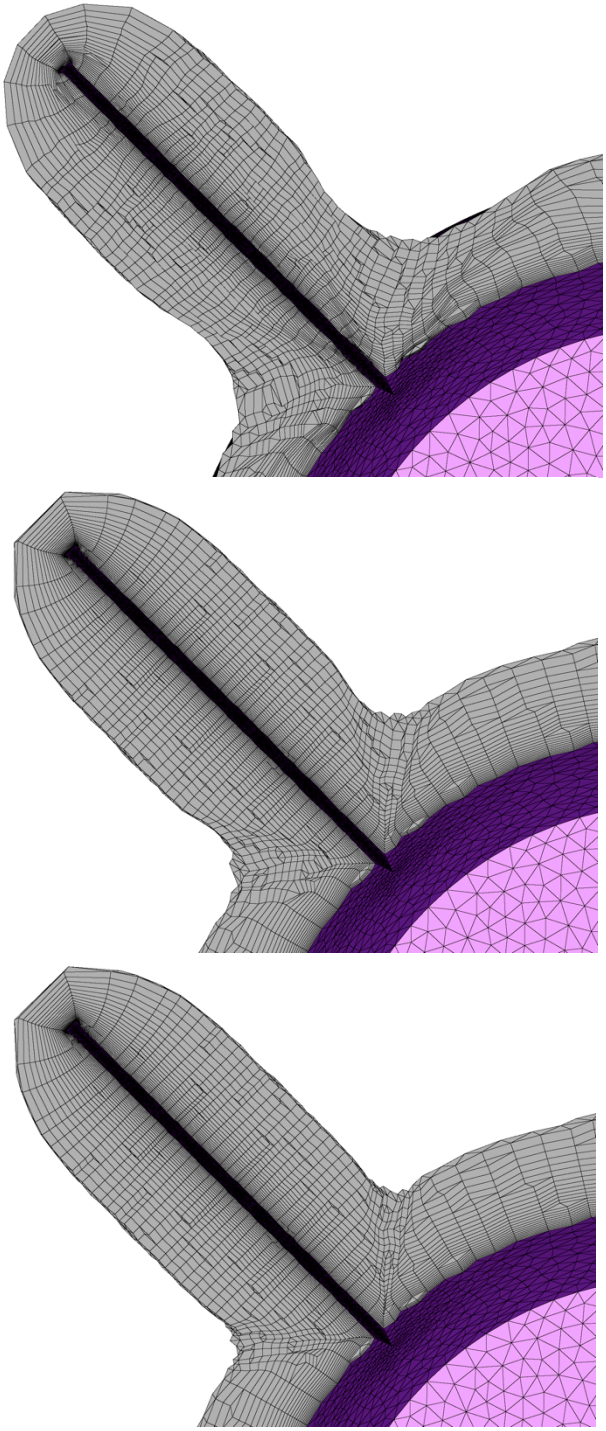

Figure 36. X=0.35 crinkle cut through meshes for P=0 (top), P=1 (middle) and P=2 (bottom).

The P=0 result has more turning, but is less smooth. Once again the P=1 result seems to be a good compromise between smooth and cell quality.

V. Conclusions

A smoothing method for quality control of extruded meshes has been presented that attempts to reposition the mesh node locations on each extruded layer to optimize a cost function. The cost function is based on element condition numbers. The basic scheme moves each node in the mesh based on the sensitivity of the element cost function with respect to the coordinates of each corner of the element. Nodal perturbation vectors are constructed from weighted averages of element cost sensitivities. Users can alter the behavior of the method by controlling the exponent P in the cost and sensitivity averaging equations. P equal to zero uses the worst cost element to control the node perturbation. P equal to two will favor the average cost and sensitivity vectors.

The condition number includes a weight matrix that maps the right-angled reference corners to a desired shapes and edge lengths. The desired normal distance was used to control the height of the elements. The other edge lengths used in constructing the weight matrices were obtained from the base of the current layer. The corner angles were specified as the existing angles for the base of the current layer. The angles for the top of the current layer were specified as $\pi/3$ for triangular prisms and $\pi/2$ for hexahedra.

Several example cases were included that demonstrate the control exerted through the weight matrix to produce high quality extruded meshes. Setting the cost and sensitivity averaging exponent to one produced the best overall results with smoothness and adequate turning at sharp corners.

References

1. Smith, R. "Transfinite Interpolation (TFI) Generation Systems," *Handbook of Grid Generation*. CRC Press, 1999, pp. 3-1 - 3-15.
2. Thompson, J. F., Warsi, Z. U. A., and Mastin, C. W. *Numerical Grid Generation: Foundations and Applications*. New York: Elsevier, 1985.
3. Steger, J. L., and Chaussee, D. S. "Generation of Body-Fitted Coordinates Using Hyperbolic Partial Differential Equations," *SIAM Journal of Scientific and Statistical Computing* Vol. 1, No. 4, 1980, pp. 431-437.
4. Chan, W. M., and Steger, J. L. "A Generalized Scheme for Three-Dimensional Hyperbolic Grid Generation," *10th Computational Fluid Dynamics Conference*. Honolulu HI, 1991.
5. Chan, W. M., Chiu, I. T., and Buning, P. G. "User's Manual for the HYPGEN Hyperbolic Grid Generator and the HGUI Graphical User Interface." NASA TM-108791, 1993.
6. Chan, W. H. "Hyperbolic Methods for Surface and Field Grid Generation," *Handbook of Grid Generation*. CRC Press LLC, 1999, pp. 5-1 - 5-26.
7. Steinbrenner, J., Wyman, N., and Chawner, J., "Development and Implementation of Gridgen's Hyperbolic PDE and Extrusion Methods", AIAA-2000-0679, 38th Aerospace Sciences Meeting and Exhibit, Reno, NV, 2000.
8. Thompson, D. S., and Soni, B. K., "Semistructured Grid Generation in Three Dimensions using a Parabolic Marching Scheme", AIAA-2000-1004, 38th Aerospace Sciences Meeting & Exhibit, Reno, NV, 2000.
9. Thompson, D. S., and Soni, B. K. "Semistructured Grid Generation in Three Dimensions Using a Parabolic Marching Scheme," *AIAA Journal* Vol. 40, No. 2, 2001, pp. 391-393.
10. Karman, S. L. "Unstructured Viscous Layer Insertion Using Linear-Elastic Smoothing," *American Institute of Aeronautics and Astronautics Journal* Vol. 45, No. 1, 2007, pp. 168-180. doi: 10.2514/1.23283

11. Marcum, D. L., "Generation of Unstructured Grids for Viscous Flow Applications", AIAA-95-0212, 1995.
12. Lohner, R., "A Parallel Advancing Front Grid Generation Scheme", AIAA-00-1005, 2000.
13. "Pointwise." Pointwise, Inc., http://www.pointwise.com.
14. Sahasrabudhe, M., Karman, S., and Anderson, W. K., "Grid Control of Viscous Unstructured Meshes Using Optimization", AIAA-2006-0532, 44th Aerospace Science Meeting and Exhibit, Reno, NV, 2006.
15. Freitag, L. A., and Knupp, P. M. "Tetrahedral Element Shape Optimization via the Jacobian Determinant and Condition Number," *8th International Meshing Roundtable*. South Lake Tahoe, CA, 1999.
16. Karman, S. L., "Adaptive Optimization-Based Smoothing for Tetrahedral Meshes", AIAA-2015-2038, 53rd AIAA Aerospace Sciences Meeting, 2015.
17. Aubert, P., Cesare, N. D., and Pironneau, O. "Automatic differentiation in C++ using expression templates and application to a flow control problem," *Computing and Visualization in Science* Vol. 3, 2001, pp. 197-208.
18. Phipps, E., and Pawlowski, R. "Efficient Expression Templates for Operator Overloading-based Automatic Differentiation." Cornell University Library, 2012.
19. Schmitt, V., Charpin, F. "Pressure Distributions on the ONERA-M6-Wing at Transonic Mach Numbers," *Experimental Data Base for Computer Program Assessment*. AGARD AR 138 ed., AGARD Fluid Dynamics Panel Working Group 04, 1979.
20. NASA. "ONERA M6 Wing," *NPARC Alliance Validation Archive*. http://www.grc.nasa.gov/WWW/wind/valid/m6wing/m6wing.html.
21. Heim, E. R. "CFD Wing/Pylon/Finned Store Mutual Interference Wind Tunnel Experiment." Calspan Corporation/ AEDC Operations, Arnold Engineering Development Center, 1991.