Triangular and Quadrilateral Surface Mesh Quality Optimization Using Local Parametrization (LA-UR-02-7190)

Rao V. Garimella [a], Mikhail J. Shashkov [a], Patrick M. Knupp [b]

[a] *MS B284, Los Alamos National Laboratory, Los Alamos, NM 87545.*

[b] *MS 0847, Sandia National Laboratories, Albuquerque, NM 87185.*

Abstract

A procedure is presented to improve the quality of surface meshes while maintaining the essential characteristics of the discrete surface. The surface characteristics are preserved by repositioning mesh vertices in a series of element-based local parametric spaces such that the vertices remain on the original discrete surface. The movement of the mesh vertices is driven by a non-linear numerical optimization process. Two optimization approaches are described, one which improves the quality of elements as much as possible and the other which improves element quality but also keeps the new mesh as close as possible to the original mesh.

Key words:
Unstructured surface mesh, triangles, quadrilaterals, numerical optimization, element quality, Jacobian condition number, Reference Jacobian Matrices

1 Introduction

Improvement of mesh quality is a very important problem for mesh generation and numerical simulation. The quality of a surface mesh heavily influences the ability of mesh generation algorithms to generate good quality solid meshes. Since surface meshes define external and internal boundaries of computational domains where boundary conditions are imposed, they also influence the accuracy of numerical simulations.

Email addresses: rao@lanl.gov (Rao V. Garimella), shashkov@lanl.gov (Mikhail J. Shashkov), pknupp@sandia.gov (Patrick M. Knupp).

Many researchers have investigated the issue of improving the quality (element shape and mesh gradation) of triangular, quadrilateral and mixed meshes in the plane [1–6]. However, optimization of surface meshes must address additional issues such as preserving the similarity of the mesh to the surface it represents. Considerable research has been conducted in the structured mesh community on using elliptic grid generation methods to smooth surface meshes generated by algebraic methods [7–9]. On the other hand, unstructured surface mesh improvement methods typically require the use of tools such as mesh modification and vertex (node) repositioning [10,11]. Mesh modification methods include edge swapping, vertex insertion (edge splitting, face splitting), vertex deletion (edge collapse) and local mesh retriangulation. Mesh modification methods change the topology of the mesh and therefore, may be more difficult to use in simulations requiring solution transfer from the original mesh to the improved mesh. Also, many mesh modification methods are only usable for simplicial (triangular and tetrahedral) meshes. Therefore, this paper only focuses on vertex repositioning for surface mesh quality improvement.

An important consideration during the improvement of surface mesh quality is to minimize changes in the discrete surface characteristics like discrete normals and curvature. Preservation of such characteristics is important for preventing drastic changes in the volume enclosed by the surfaces and in forces like surface tension that depend on surface properties. When improving surface mesh quality by vertex repositioning, changes in the surface properties can usually kept small by keeping the vertex movements small and by constraining the vertices to a smooth surface underlying the mesh or to the discrete surface described by the faces of the original mesh. The smooth surface may be defined in a geometric modeler or a locally smooth geometric support may be defined for each patch of elements in the form of a Bezier or polynomial patch [10].

An approach commonly used to constrain nodes to the underlying smooth surface is to reposition each vertex in a locally derived tangent plane and then pull the vertex back to the smooth surface [11,10]. Another approach to constrain the vertices to the smooth or discrete surface is to reposition them in a 2D parametrization of the surface. When the vertices are mapped back from the 2D parametric space to the real space, they are guaranteed to remain on the original surface. If the mesh has an underlying smooth surface, the parametric space of the surface is usually available from the geometric modeler in which the surface is defined or from the analytical definition of the surface. However, if such a smooth surface is not available, then the mesh itself must be used as a discrete surface from which to derive a parametric space. Several researchers have developed techniques to build a global parametric space from a given triangular mesh [12–16]. However, all these methods involve substantial computational cost since they often require solution of a system of nonlinear equations. Also, they cannot be used directly to parametrize closed surfaces. Instead, the closed surfaces must be cut into one or more pieces

which are then parametrized separately. Finally, since the methods operate only on triangular meshes, surface meshes with other element types must be preprocessed before the methods can be applied to them.

In this paper, an optimization-based vertex repositioning procedure is described to improve the quality of a surface mesh without an underlying smooth surface such that the essential mesh and surface characteristics of the original mesh are preserved. The method constrains the vertices to the discrete surface defined by the original mesh by repositioning them in a series of local parametric spaces derived from individual mesh elements (faces, edges). The repositioning procedure keeps track of the original mesh element that each vertex is moving in and if a vertex moves out of the parametric space of the element, the vertex is considered to have moved into the parametric space of an adjacent element. The optimization is then resumed after deriving the parametric coordinates of the vertex from the local parametric space of the adjacent element. When the repositioned vertices are mapped back to the real space, each vertex lies on the mesh element whose local parametric space it is in. The method imposes no restrictions on the nature of the discrete surface or how far vertices may move from their original positions on this discrete surface. The procedure has been implemented for surface meshes containing triangles and quadrilaterals. Using a recent publication on the barycentric mapping of general polygons, it is expected that the procedure can be extended to handle general mesh elements easily [17]. The repositioning of the vertices is driven by numerical optimization of some objective function that seeks to (a) improve the quality of mesh elements as much as possible or (b) improve the quality of all elements in the mesh while keeping the vertices as close as possible to their original locations.

The rest of the paper is organized as follows. Section 2 lists some of the notation used in this paper. Section 3 describes the method of optimizing an objective function with respect to local parametric coordinates. The section discusses the element based local parametrization, line search with respect to local parametric coordinates and moving vertices from one parametric space to another. Section 4 describes two methods for improving the quality of surface mesh faces using optimization with respect to local parametric coordinates. Section 5 presents several examples of optimization of triangular and quadrilateral meshes to demonstrate the features of both optimization methods.

2 Notation

F_i	i'th face in mesh
E_i	i'th edge in mesh
V_i	i'th vertex (node) in mesh
$\mathcal{V}$	Set of all vertices (nodes) in a mesh
$\mathcal{A}(B)$	Set of all entities of type A connected to or contained in entity B e.g., $\mathcal{F}(V_i)$ is the set of faces connected to vertex V_i, and $\mathcal{V}(F_i)$ is the set of vertices of face F_i
V_i^R	Reference position of vertex i
$(\mathbf{e}_k)_i$	Edge vector corresponding to edge E_k originating at V_i
$(\mathbf{e}_k^R)_i$	Reference edge vector corresponding to edge E_k originating at V_i^R
$\mathbf{J}$	Jacobian matrix of the mapping of an element to a parent element
$\mathbf{J}_{ji}$	Jacobian matrix of element F_j at vertex V_i
$\lvert\mathbf{J}\rvert_F$	Frobenius norm of $\mathbf{J}$ defined by $trace(\mathbf{J}^T\mathbf{J})$
κ	Condition number of Jacobian, $\kappa(\mathbf{J}) = \lvert\mathbf{J}^{-1}\rvert_F\lvert\mathbf{J}\rvert_F$
$\mathbf{x}_i$	3D coordinates of vertex i
$\mathbf{s}_i$	Local parametric coordinates of vertex i

3 Optimization with respect to Parametric Coordinates

Consider an objective function, $\Psi(\mathbf{x})$, defined in terms of the real coordinates, $\mathbf{x}$, of all the vertices of a surface mesh such that minimization of this function drives the mesh vertices to locations that improve the quality of the mesh with respect to some quality measure.

If this objective function is minimized directly with respect to the real coordinates of the vertices, the search direction for the minimization may indicate vertex movement off the original surface mesh. Therefore, the optimization must be performed with respect to the coordinates of the vertices in a 2D parametric space derived from the discrete surface mesh. The optimization repositions vertices in the parametric space and when the vertices are mapped back to 3D space, they are guaranteed to lie on the original discrete surface. In this work, the repositioning of vertices is done in a series of local parametric

spaces derived from elements of the original mesh instead of a global parametric space derived at a much greater expense from the complete surface mesh. The optimization process moves each vertex in its appropriate local parametric space, which may change during the vertex movement. When mapped back to real space, the vertex lies on the original mesh element corresponding to the current local parametric space it is moving in. If the optimization process drives the vertex out of bounds of the parametric space of one element, the vertex is considered to have moved into the parametric space of an adjacent element. The optimization is then resumed after deriving the parametric coordinates of the vertex from the local parametric space of the adjacent element.

In the following sections, the process of optimizing any objective function with respect to local parametric coordinates is described in more detail including ideas of element based local parametrization, line search in local parametric spaces and parameter updating for repositioning vertices. The numerical method chosen for optimization is the well known non-linear conjugate gradient method [18,19]. The objective functions used to drive the optimization are described later in Section 4.

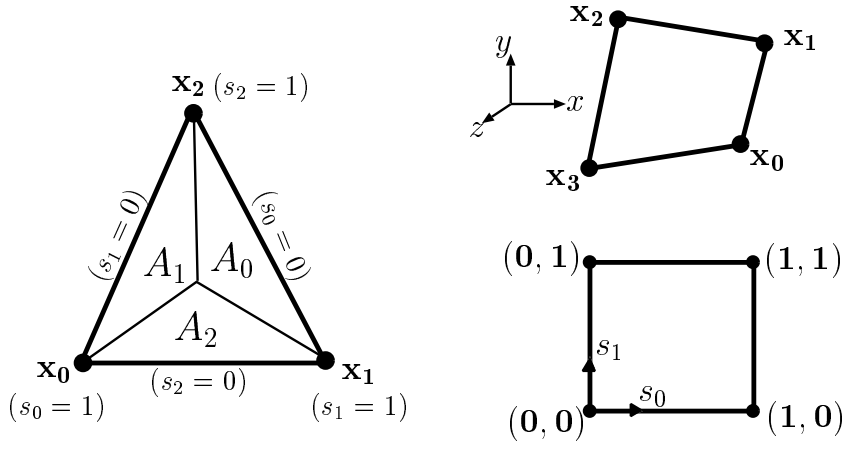
3.1 Element based Local Parametrization

The local parametric spaces used in the optimization and vertex repositioning procedure are derived from mappings of edges, and faces (triangles, quadrilaterals) to parent or canonical elements in 2D space commonly used in finite element methods.

Vertices on the boundary of the surface mesh (i.e., on a model edge) use the parametric spaces of boundary mesh edges of the original surface mesh. The parametric space of each boundary edge is derived by mapping it to a unit line segment along the X axis giving rise to parametric coordinate $0 \leq s_0 \leq 1$. Vertices in the interior of the surface mesh (i.e., on a model face) use the parametric spaces of the faces of the original mesh. The local parametric space for a mesh triangle is derived using a barycentric mapping [20], resulting in parametric coordinates $0 \leq (s_1, s_2) \leq 1$ as shown in Figure 1a. A local parametric space for a quadrilateral is derived using isoparametric mapping [20], giving rise to parametric coordinates $0 \leq (s_1, s_2) \leq 1$, shown in Figure 1b. Meyer et. al. [17] have proposed a new barycentric mapping method which can be used to extend this procedure for parameterizing general straight sided polygonal faces.

Any procedure repositioning vertices using the element based local parametrizations must keep track of which mesh element of the original mesh each vertex is moving in (referred to as the ***base*** element) and the coordinates of the vertex

in the parametric space of the base element. During the optimization process, all objective function evaluations are done after mapping the parametric coordinates of the vertex in the base element to real coordinates.



$$
\begin{aligned}
& A = A_0 + A_1 + A_2 \\
& s_i = A_i/A, \quad 0 \le s_i \le 1 \\
& s_0 + s_1 + s_2 = 1 \\
& \mathbf{x} \;=\; (x, y, z) \\
& \quad =\; s_o\mathbf{x_0} + s_1\mathbf{x_1} + s_2\mathbf{x_2}
\end{aligned}
$$

(a)

$$
\begin{aligned}
& 0 \le s_0, s_1 \le 1 \\
& \mathbf{x} \;=\; (x, y, z) \\
& \quad =\; (1 - s_0)(1 - s_1)\mathbf{x_0} + \\
& \quad\quad s_0(1 - s_1)\mathbf{x_1} + s_0 s_1 \mathbf{x_2} + \\
& \quad\quad (1 - s_0)s_1\mathbf{x_3}
\end{aligned}
$$

(b)

Fig. 1. (a) Barycentric mapping for triangle, (b) Isoparametric mapping for quadrilateral.

3.2 Line Search

In the optimization procedure, the gradient of the objective function with respect to the parametric coordinates is computed by numerical differentiation. This gradient is used to compute a line search direction in the local parametric space. The line search is used to find a distance α along the parametric search direction $\mathbf{d}$ such that the objective function is minimized or the constraints of the line search are encountered. For surface optimization with local parameterization, the line search is subject to two constraints, parametric bounds and mesh validity, as discussed next.

During a line search, a vertex that travels out of the parametric space of a base mesh face, moves out of the face and off the original surface mesh. In

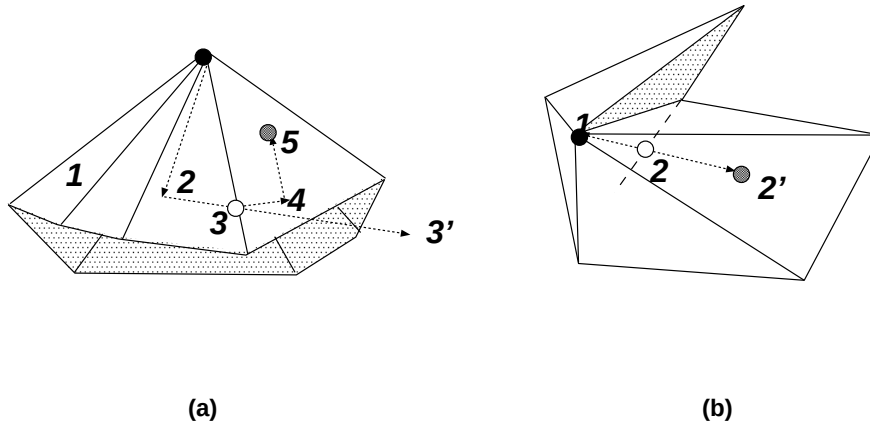


Fig. 2. Line search constraints: (a) Parameter bounds, (b) Validity constraint.

such a case, computing quantities such as the gradient of the objective function with respect to the current parametric coordinates of the vertex becomes meaningless. Therefore, if a vertex tries to move out of bounds of the local parametric space, the line search is stopped at the boundary of the base face. For example, in Figure 2a, the line search tries to proceed from point 2 to point 3', which is outside the triangle and off the surface triangulation. However, it encounters the parametric bounds of the triangle at point 3 (which is on an edge of the triangle) and therefore, the line search is stopped at that point.

Also, it is possible that one of the elements connected to the vertex becomes invalid (inverted) due to movement of the vertex along the search direction in which case the line search must be stopped. This is shown in Figure 2b where the line search must be stopped at point 2 because further movement toward point 2' renders the shaded triangle invalid.

The line search procedure is implemented as an incremental stepping algorithm with step size control. The line search starts with a very small step size and checks if the function has decreased, the parameters are within bounds and if the mesh is valid. If so, the step size is increased and the process is repeated; if not, the step size is cut in half (up to a minimum) and the checks are repeated. The algorithm has additional refinements for zeroing in on the minimum with better accuracy.

3.3 Parameter Update and Parametrization Change

Once the line search along a direction has terminated, the step size, α, obtained from it is used to update the parametric coordinates of the vertex as $\mathbf{s}_{new} = \mathbf{s}_{old} + \alpha\mathbf{d}$. If the line search terminates normally at a minimum or

because further movement in the search direction would have made the mesh invalid, a new optimization iteration is started with a new gradient calculation. However, if the line search terminates because the parametric bounds were reached, then it is assumed that the vertex is trying to move out of the current mesh face. In such a case, the optimization iteration is terminated and the vertex is considered to have moved into the parametric space of an adjacent mesh face. Since the vertex is moved into a different parametric space, the optimization procedure is restarted from the parametric location of the vertex in the new face, discarding the previous search direction and any saved gradient information (in a conjugate gradient method).

At the start of the optimization, the base mesh face for initial movement of a vertex is chosen arbitrarily from the set of faces connected to the vertex. Therefore, it is possible that the objective function does not decrease along any direction in the chosen face and that a line search in the face will terminate without any movement from the current location. In such a case, an adjacent face connected to the vertex is chosen as the base face and the optimization iteration is performed in that face. The process is repeated until a viable base face is found for moving the vertex. If the vertex cannot be moved in any base face connected to the vertex without increasing the objective function value or encountering an optimization constraint, the vertex is taken to be at its optimal location.

During the search, if a vertex is at a common edge of two base mesh faces of the original mesh, it is possible that the gradient with respect to one face points into the adjacent face and vice versa, leading to the search switching infinitely between the two faces. This condition is recognized in the algorithm and resolved by moving the vertex along the edge. The line search direction along the edge is taken to be the one closer to the negative of the gradient direction.

Figure 3 illustrates the movement of vertices during an optimization with respect to parametric coordinates for a planar triangulation. The mesh was improved by minimizing an objective function based on the condition number quality measure (See Sec. 4.2) over the entire mesh. Note the vertex movements across several elements of the original mesh as well as movements along edges.

3.4 Global Optimization by Local Iterations

Consider the minimization of a global objective function, i.e., an objective function that involves the coordinates of all the vertices of the mesh. It would be most efficient if a global procedure could be used to minimize this objective function so that all the mesh vertices could be moved toward their optimal

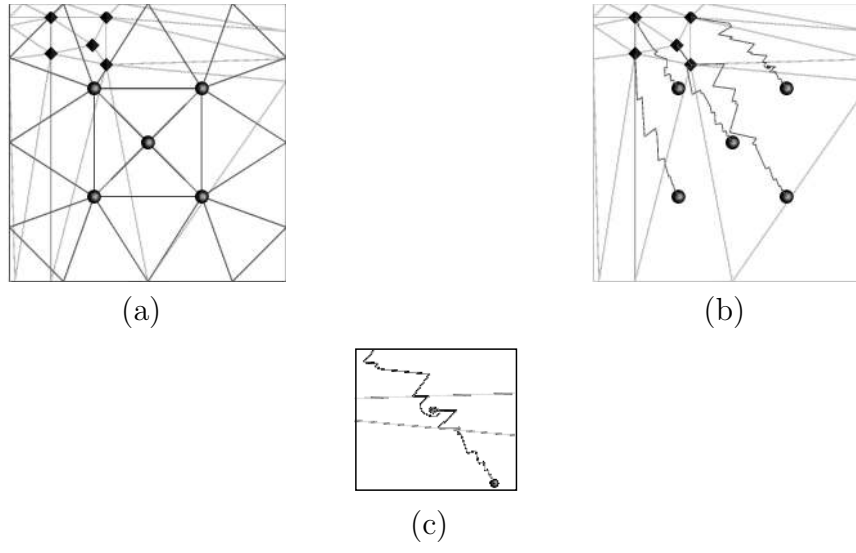


Fig. 3. (a) Original (light lines) and final (dark lines) mesh, (b) Paths taken by vertices from their original positions (shown as $\diamondsuit$) to their final positions (shown as $\bullet$), (c) Zoom-in of one of the paths.

position simultaneously. However, the use of local parameterization for vertex movement imposes strong constraints on a global optimization process. The line search necessary in the global optimization seeks a single step size for the parametric coordinates of all the mesh vertices. However, if a parametric coordinate for even a single vertex goes out of bounds, the line search must end for all the parameters in the problem and the optimization restarted, making the optimization very inefficient.

To increase the efficiency of the optimization, the approach used here is to minimize the global objective function by iteratively minimizing a local component of the global function at each mesh vertex. The local component of the global function at a given vertex is constructed so that every term in the global function involving the vertex is accounted for in the local function. The optimization loops over all the mesh vertices several times until the optimization converges to a solution. The criteria for convergence is that the movement of all the vertices is negligible for several iterations.

4 Optimization of Surface Mesh Quality

4.1 Condition Number Shape Measure for Mesh Faces

Of the many measures for evaluating the shape (or quality) of triangular and quadrilateral elements, the Condition Number Shape Measure [11] is one with a strong mathematical foundation. This measure is derived from the Jacobian matrix of an element mapping as described below.

Consider a vertex V_i, connected to a set of of edges, $\mathcal{E}(V_i)$, and faces, $\mathcal{F}(V_i)$ as shown in Figure 4. Assume that one of the faces $F_j \in \mathcal{F}(V_i)$ has edges $E_p \in \mathcal{E}(V_i)$ and $E_q \in \mathcal{E}(V_i)$ connected to vertex V_i. The triangle formed by edges E_p and E_q can always be mapped to a right triangle in 2D space with V_i mapped to the origin, a unit vector representing E_p along the x-axis and a unit vector representing E_q along the y-axis. Then, the *Jacobian matrix*, $\mathbf{J}_{ji}$, of the mapping of the triangle to the right triangle in 2D space, evaluated at vertex V_i, is given by $\mathbf{J}_{ji} = [\mathbf{e}_p\, \mathbf{e}_q]$ where, $\mathbf{e}_p$ and $\mathbf{e}_q$ are edge vectors representing edges E_p and E_q, of lengths l_p and l_q respectively. The condition number of the Jacobian matrix is defined as $\kappa(\mathbf{J}_{ji}) = |\mathbf{J}_{ji}^{-1}|_F |\mathbf{J}_{ji}|_F$ where $|\cdot|_F$ is the Frobenius norm of its matrix operand.

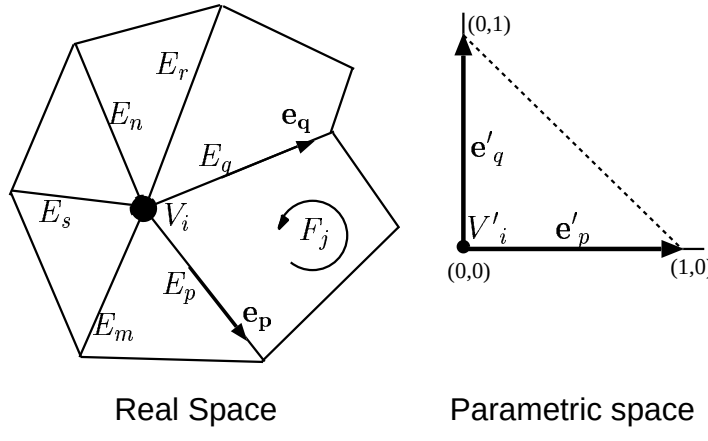


Fig. 4. Definition of edge vectors, $\mathbf{e}_p$, $\mathbf{e}_q$ for calculating the Jacobian of an element F_j at vertex V_i.

Since $\mathbf{J}_{ji}$ is a 3x2 matrix for a triangle in 3D, its condition number has to be calculated by singular value decomposition methods. On the other hand, the Jacobian matrix of a triangle in 2D space is a 2x2 matrix whose condition number can be calculated more easily as

$$\kappa(\mathbf{J}_{ji}) = \frac{(l_p^2 + l_q^2)}{2A_j} \tag{1}$$

where A_j is the area of the triangle formed by E_p and E_q [21,11]. This condition number is only a function of triangle lengths [1]; therefore, it is invariant with rotation of the triangle in the plane. Since there always exists a coordinate system in which an arbitrarily oriented triangle lies on one of its coordinate planes, it suggests that the condition number is also useful for measuring the quality of arbitrarily oriented triangles in space.

The condition number shape measure as described above can be used for measuring the deviation of an element corner from a right angle corner formed by unit edge vectors. In a given mesh, the quality of any element can then be measured by a suitable combination of the Jacobian condition numbers at the element corners. Also, the quality of the mesh at any vertex may be measured by a suitable combination of the Jacobian condition numbers of the element corners incident upon that vertex.

4.2 Condition Number Based Optimization

Consider the minimization of a function defined as the sum of condition numbers of the face corners incident at a given vertex, V_i, as given below:

$$\psi_i^c(\mathbf{x}_i) = \sum_j \kappa(\mathbf{J}_{ji}(\mathbf{x}_i)) = \sum_j \frac{l_p^2(\mathbf{x}_i) + l_q^2(\mathbf{x}_i)}{A_j(\mathbf{x}_i)}, \qquad j \in \{j \mid F_j \in \mathcal{F}(V_i)\} \tag{2}$$

where l_p and l_q are the lengths of the respective edges E_p and E_q of face F_j connected to vertex V_i and $\mathbf{x}_i$ is the coordinate vector of V_i. Note the presence of area A_j in the denominator as a barrier function which discourages vertex movements that tend to make the triangle formed by E_p and E_q degenerate. Note, however, that it is still important to check explicitly for degeneracy or invalidity of elements in the line search process described in Section 3.2 since it is possible for some line search techniques to jump to the other side of the degeneracy barrier.

The minimization of ψ_i^c attempts to smooth the distribution of face angles and edge lengths around a vertex since all the edge vector pairs are trying to reach equal length and form a right angle. Based on this property, a strategy

[1] A_j is a function of the lengths of the triangle sides

can be formed for improving the quality of a mesh by minimizing a global condition number based objective function, Ψ^c, defined as:

$$\Psi^c = \sum_i \psi_i^c, \quad i \in \{i \mid V_i \in \mathcal{V}\} \tag{3}$$

where $\mathcal{V}$ is the set of all mesh vertices.

As discussed in Section 3.4, the global function Ψ^c is minimized by minimizing a local function, $\widetilde{\psi_i^c}$, at each vertex. $\widetilde{\psi_i^c}$ at a vertex V_i is composed of all terms of Ψ^c that involve the coordinates of V_i. Therefore, $\widetilde{\psi_i^c}$ is formed by visiting each element F_j connected to vertex V_i and adding the Jacobian condition number of the element at V_i and the Jacobian condition numbers at both its edge connected neighbors in that element (See Figure 5). Mathematically, this is written as

$$\widetilde{\psi_i^c} = \sum_j \sum_k \kappa(\mathbf{J}_{jk}),$$
$$j \in \{j \mid F_j \in \mathcal{F}(V_i)\}, \qquad k \in \{k \mid V_k \in \mathcal{V}(F_j) \ \cap \ \mathcal{V}(\mathcal{E}(V_i))\ \} \tag{4}$$

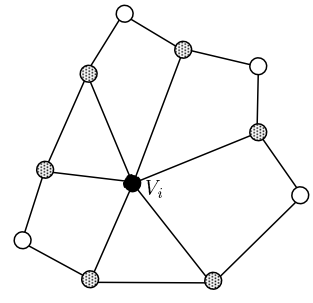


Fig. 5. Vertices involved in the local objective function expression, $\widetilde{\psi_i^c}$, for V_i. The shaded circles along with the black circle (V_i) represent the vertices at the Jacobian is computed for use in $\widetilde{\psi_i^c}$. The white circles represent vertices whose real locations contribute to the Jacobians at the vertices with shaded circles.

The optimization procedure visits each vertex of the mesh in turn and optimizes the position of the vertex using the local objective function, $\widetilde{\psi_i^c}$. The local optimization can be done by any optimization method such as the non-linear conjugate gradient method. For surface meshes, the optimization is conducted with respect to local parametric coordinates as described in Section 3. Several optimization iterations are made over all the vertices of the mesh leading to a minimization of the global function, Ψ^c. The iterations are stopped when the movement of all vertices becomes negligible.

4.3 Reference Jacobian based Optimization Method

4.3.1 Motivation

The global condition number minimization procedure allows mesh vertices to move along the surface as much as necessary to minimize the objective function, Ψ^c. However, in certain situations such as Arbitrary Lagrange-Eulerian (ALE) simulations [22–24], it is of interest to keep the vertices of the original mesh as close as possible to their original locations while improving the shape of the mesh elements. In ALE methods, the Lagrangian step dictates a certain movement for the vertices based on the physics of the problem. This can cause the mesh to be distorted enough that the simulation cannot proceed unless the quality of the elements is improved. After the mesh is improved, the solution from the distorted mesh must be transferred to the improved mesh before continuing the simulation. Since the accuracy of the solution transfer strongly depends on the similarity of the two meshes, it is important to devise a procedure that improves mesh quality but also limits the extent that vertices can move from their original locations. Such an optimization procedure, referred to here as *Reference Jacobian Matrix (RJM) based Optimization*, has been described earlier by Shashkov et. al. [21,24] for planar meshes. In the current work, the RJM optimization procedure has been combined with optimization with respect to local parametrizations, resulting in a strategy for improving surface mesh quality while keeping the vertices of the mesh on the faces of the original mesh and close to their original positions.

The RJM mesh improvement is a two stage procedure, consisting of a series of local condition number based optimizations and a global RJM optimization as described next.

4.3.2 Local Condition Number based Optimization (Step I)

This is the first stage of the RJM optimization strategy. In this step, the locally optimal position of each mesh vertex is computed with respect to the fixed position of its neighbors. The objective function for optimization is the local condition number function, $\widetilde{\psi_i^c}$, described in Eq. 4, Section 4.2. However, in this step, the vertex is not moved to its locally optimal position. Rather, the optimal position of each vertex, described by a base face and the parametric coordinates of the vertex in the base face, is stored as a virtual position for use in the second stage of the mesh improvement procedure.

4.3.3 Reference Positions, Reference Edges and the Reference Jacobian Matrix

The locally optimal position computed and stored for each vertex in the first stage of the procedure is known as the *reference position* for the vertex. After reference positions are calculated for all mesh vertices, two *reference edge vectors* are calculated for each edge in the mesh; each reference edge vector goes from the reference position of one vertex of the edge to the original position of the other. The idea of reference edges is illustrated in Figure 6, where E_m is an edge with vertices V_a and V_b. The reference positions of V_a and V_b are V_a^R and V_b^R respectively. The two reference edge vectors for E_m are $(\mathbf{e}_m^R)_a$ and $(\mathbf{e}_m^R)_b$, where the outer subscript indicates which of the vertices is at its reference position.

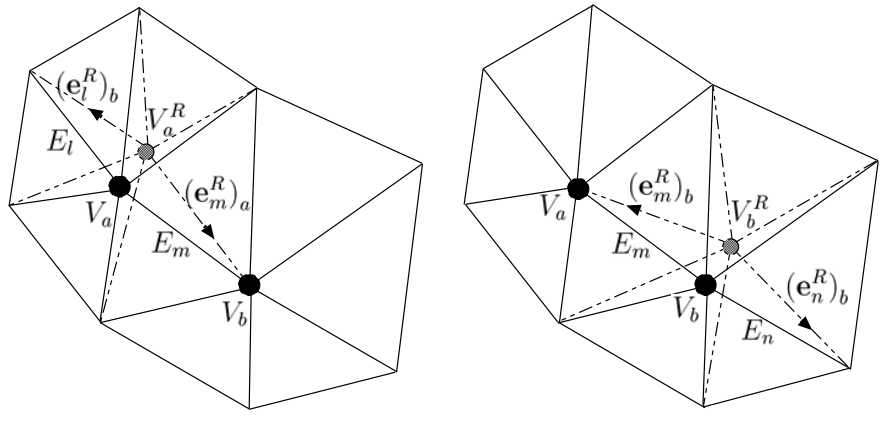


Fig. 6. Reference positions and reference edge vectors.

Using the concept of reference edge vectors, it is now possible to define *Reference Jacobian Matrices* (RJMs) just as Jacobian matrices were defined for a mesh without reference positions. Therefore, if the edges of F_j connected to vertex V_i are E_p and E_q, their reference edges are E_p^R and E_q^R, and their reference edge vectors are $(\mathbf{e}_p^R)_i$ and $\mathbf{e}_q^R)_i$ respectively, then the reference Jacobian of F_j at V_i is defined as $\mathbf{J}_{ji}^R = \left[(\mathbf{e}_p^R)_i \; (\mathbf{e}_q^R)_i\right]$.

4.3.4 Global Optimization based on Reference Jacobian Matrix (Step II)

The second stage of the mesh improvement procedure is a global optimization based on the definition of reference Jacobian matrices. The goal of this step is to find a valid mesh configuration such that each edge is in a compromise configuration between its pair of reference edges. It is expected that such a configuration for the edges will improve mesh quality, since the reference edge

vectors were formed by locally improving mesh quality at each mesh vertex. It is also expected that the optimized mesh will not deviate drastically from the base mesh, since each reference edge vector has one of its vertices at its original position and the other at the locally optimal position.

The objective function for the global optimization quantifies the difference between the Jacobian matrices of the current mesh configuration and the reference Jacobian matrices as shown below:

$$\Psi^R = \sum_i \sum_j \frac{\left|\mathbf{J}_{ji} - \mathbf{J}_{ji}^R\right|_F^2}{A_j / A_{ji}^R}, \qquad i \in \{i \mid V_i \in \mathcal{V}\},\ j \in \{j \mid F_j \in \mathcal{F}(V_i)\} \quad (5)$$

where, $\mathcal{V}$ is the set of all mesh vertices, A_{ji}^R is the area of the triangle formed by edge vectors, $(\mathbf{e}_p^R)i$ and $(\mathbf{e}_q^R)i$. Note that, similar to the objective function for local optimization, the objective function includes a barrier term A_j in the denominator in the form of the triangle area to prevent mesh invalidity. Since the Jacobian matrix and the reference Jacobian matrix are formed from the mesh edges and the reference edges respectively, optimization of Ψ^R makes the edges of the final mesh as close as possible to their respective reference edge vectors.

As discussed in Section 3.4, the global objective function, Ψ^R is minimized by iteratively minimizing a local component of the global function at each mesh vertex. The local component of the global objective function that involves the real and reference positions of V_i is given as:

$$\widetilde{\psi_i^R} = \sum_j \sum_k \frac{\|\mathbf{J}_{jk} - \mathbf{J}_{jk}^R\|^2}{A_j / A_{jk}^R}, \qquad j \in \{j \mid F_j \in \mathcal{F}(V_i)\}, \qquad k \in \{k \mid V_k \in \mathcal{V}(F_j) \ \cap \ \mathcal{V}(\mathcal{E}(V_i)\}$$

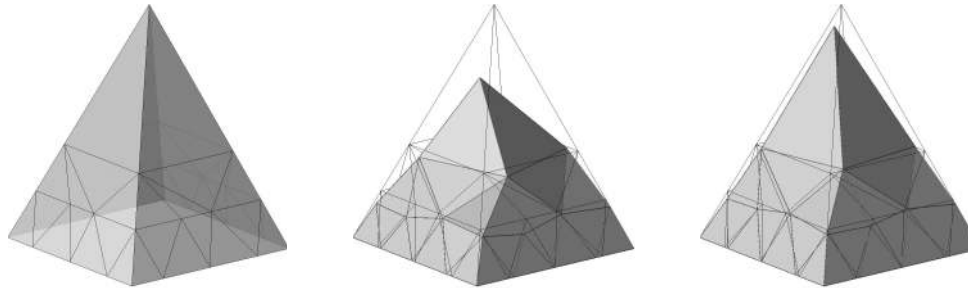
In the expression, the outer sum is over all faces connected to the vertex and the inner sum is over all vertices of a face that include V_i itself or are edge-connected to V_i.

Thus, the second stage of optimization visits each mesh vertex, V_i, and conducts a minimization of the local function, $\widetilde{\psi_i^R}$ by repositioning V_i. Minimization of the local function results in a reduction of the global function, Ψ^R. The procedure loops over all the mesh vertices several times until the optimization converges to a solution. The criteria for convergence is that the movement of all the vertices is negligible for several iterations. It can be seen that the

first and second stage optimizations are similar except for the use of different objective functions.

5 Results

Figure 7 shows a simple example to illustrate the effects of a condition number optimization (**CN Opt.** or **CNO**) and reference Jacobian based optimization (**RJ Opt.** or **RJO**) on a non-planar surface mesh. Figure 7a shows the original pyramid shaped mesh on which the two optimization techniques are applied. Figure 7b shows the effect of optimizing the CN objective function and Figure 7c shows the effect of optimizing the RJ objective function. In both cases, the apex vertex lies on the left lateral surface of the original pyramid. It can be seen that the CN optimization improves the shapes of the triangles more than the RJ optimization. On the other hand, the RJ optimization results in lesser movement of the apex vertex from its original position.



(a) (b) (c)

Fig. 7. (a) Original Mesh, (b) Mesh optimized with condition number objective function, (c) Optimized with reference Jacobian objective function. Note that in both cases, the apex vertex is on the lateral surface of the original pyramid.

Figure 8a shows the triangular mesh of a pig, and Figures 8b and 8c show the results of the CN optimization and RJ optimization on the mesh respectively. It is again clear from the example that the CN optimization improves the shape of mesh elements more than the RJ optimization, but it also causes much more movement of the vertices. In particular, note that the CN optimization destroys much of the anisotropy in the midsection of the pig and smooths away the local refinement around the pig's mouth while the RJ optimization preserves these characteristics of the mesh.

The mesh optimization procedure has been implemented and tested for mixed triangular and quadrilateral meshes as shown in Figure 9. Figure 9a shows a mixed mesh of the pig, and Figures 9b and 9c show the CN optimized mesh and RJ optimized mesh respectively.

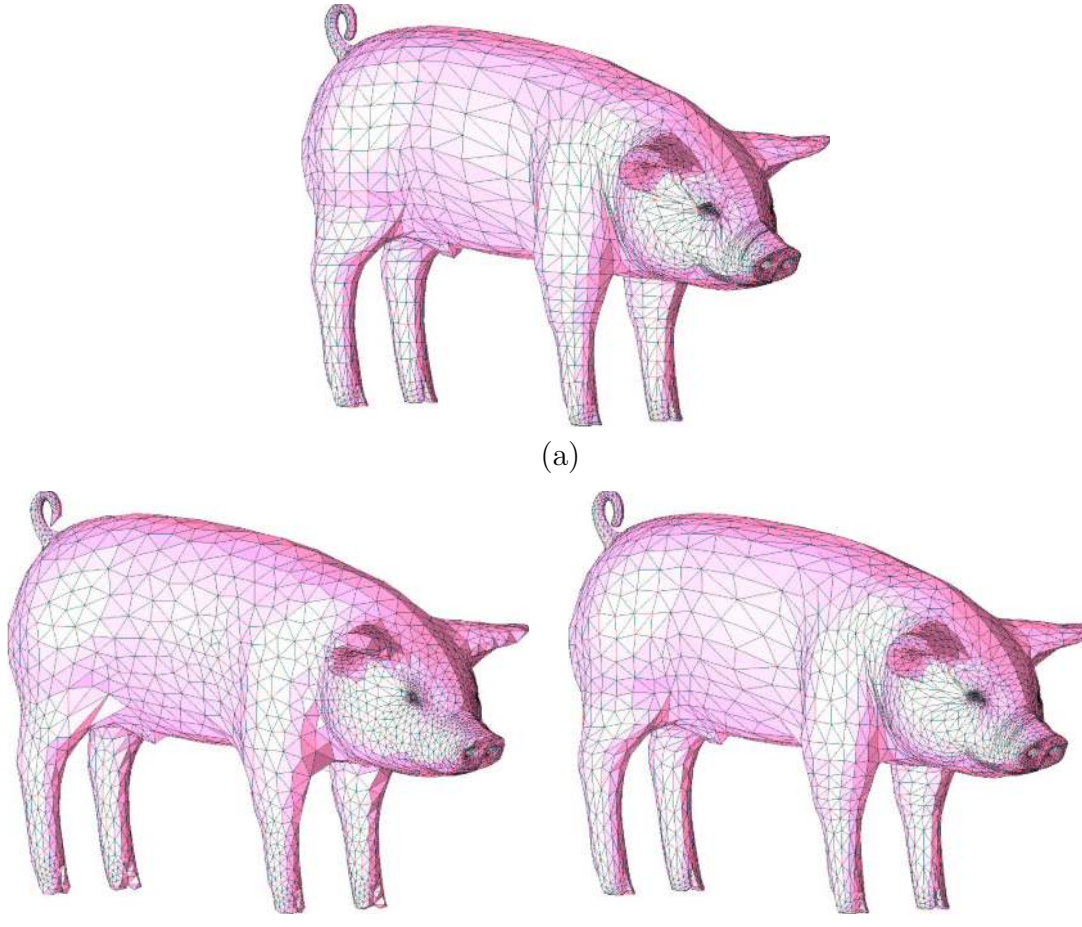


Fig. 8. (a) Mesh of pig with anisotropy and local refinement, (b) Mesh optimized with global condition number function, (c) Mesh optimized with reference Jacobian function.

The differences between the results of CN optimization and the RJ optimization of the triangular and mixed meshes of the pig can be quantitatively demonstrated by the mesh data presented in the following tables (Table 1 and Table 2).

Table 1 shows the improvement in the distribution of *normalized average condition number*, $\bar{\mathcal{K}}$, of elements in the triangular and mixed meshes with the two types of optimization. The normalized average condition number for an element is defined as the mean of the condition numbers at the vertices of an element, normalized so that an equilateral triangle or square quadrilateral will produce a value of 1.

Table 2 shows various quantities computed to measure the change in the meshes and the discrete surfaces using the two methods of optimization. In the

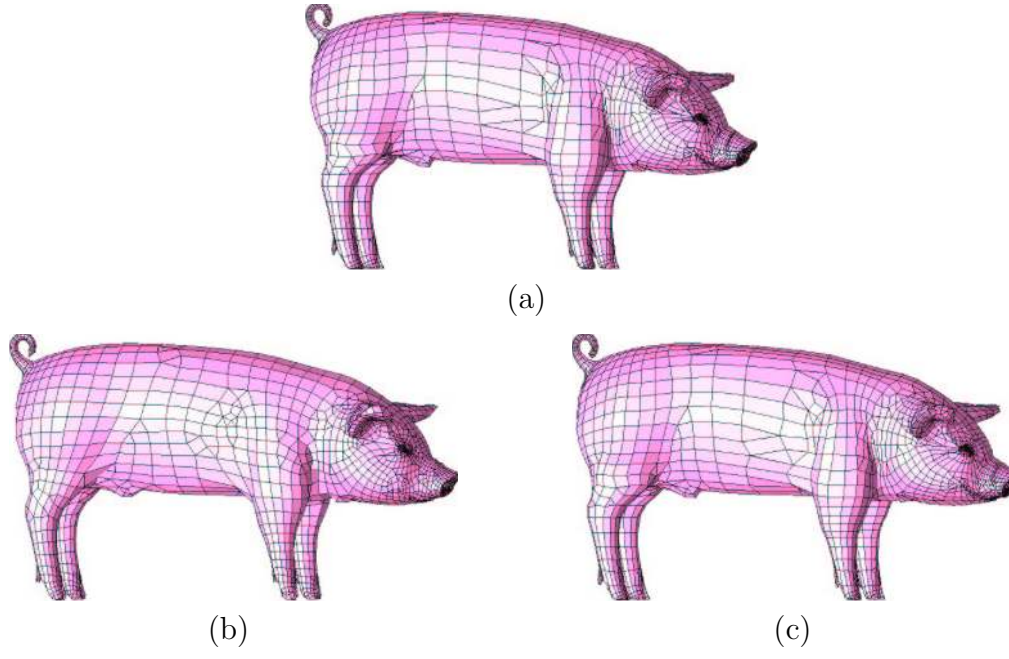


Fig. 9. (a) Mixed mesh (triangular and quadrilateral elements) of pig with anisotropy and local refinement, (b) Mesh optimized with global condition number function, (c) Mesh optimized with reference Jacobian function.

table, the normalized Hausdorff distance is computed by finding the minimum distance from each vertex of the original mesh to the new mesh, taking the maximum of these distances [25,26] and normalizing it by the problem size. The problem size is defined as the maximum length of the domain along the three coordinate directions. The maximum vertex movement is the maximum distance traveled by any vertex from its original position and the average vertex movement is the mean of the distance traveled by all vertices from their original positions; these are also normalized by the problem size.

Finally, a complex mesh of a sculpture is presented in Figure 10 to illustrate the effectiveness of this procedure on large surface meshes. The original mesh for this model was obtained from the Cyberware, Inc. [2] which was then coarsened and converted into a mixed mesh using software from the Scientific Computation Research Center at Rensselaer Polytechnic Institute. The coarsened mesh (Figure 10a) was used to obtain the optimized meshes shown in the example. A CN optimization resulted in the mesh shown in Figure 10b and a RJ optimization yielded the mesh shown in Figure 10c.

The condition number histograms for the three meshes are presented in Table 3 and the measures for change in surface characteristics are presented in Table 4.

[2] `http://www.cyberware.com/samples`

$\bar{\mathcal{K}}$			Triangle Mesh			Mixed Mesh		
			Original	CN Opt.	RJ Opt.	Original	CN Opt.	RJ Opt.
1.0	–	1.5	3921	6830	5124	2540	3745	2971
1.5	–	2.0	1734	156	1257	736	71	586
2.0	–	3.0	917	48	525	349	9	232
3.0	–	4.0	247	3	100	94	0	31
4.0	–	5.0	102	0	22	45	2	4
5.0	–	7.5	93	2	7	46	0	1
7.5	–	10.0	11	1	1	6	0	2
10.0	–	15.0	12	0	4	9	0	0
15.0	–		3	0	0	2	0	0

Table 1
Histograms of Normalized Average Condition Number of elements in Original and Optimized Meshes for triangular and mixed meshes of a pig (Figure 8).

	Triangle Mesh		Mixed Mesh	
Measure (% of problem size)	CN Opt.	RJ Opt.	CN Opt.	RJ Opt.
Hausdorff Distance	2.7%	0.6%	2.38%	0.8%
Max. Vertex Movement	11.1%	3.1%	8.0%	1.3%
Ave. Vertex Movement	1.7%	0.3%	1.4%	0.2%

Table 2
Quantitative measures of the change in the mesh and discrete surface characteristics for CN optimization and RJ optimization for triangular and mixed meshes of a pig (Figure 8); distances are presented as a percentage of the problem size.

6 Conclusions

A procedure was presented to improve the quality of complex surface meshes without an underlying smooth surface using numerical optimization. The optimization is designed to improve the quality of the mesh faces without distorting the discrete surface too much. The vertices are kept on the original surface mesh using movement in local parametric spaces of mesh faces. Two methods were proposed for improving the quality of the surface mesh. The first method improved the quality of mesh elements as much as possible by minimizing a global condition number objective function. The minimization of the global function was achieved by minimizing a local component of the

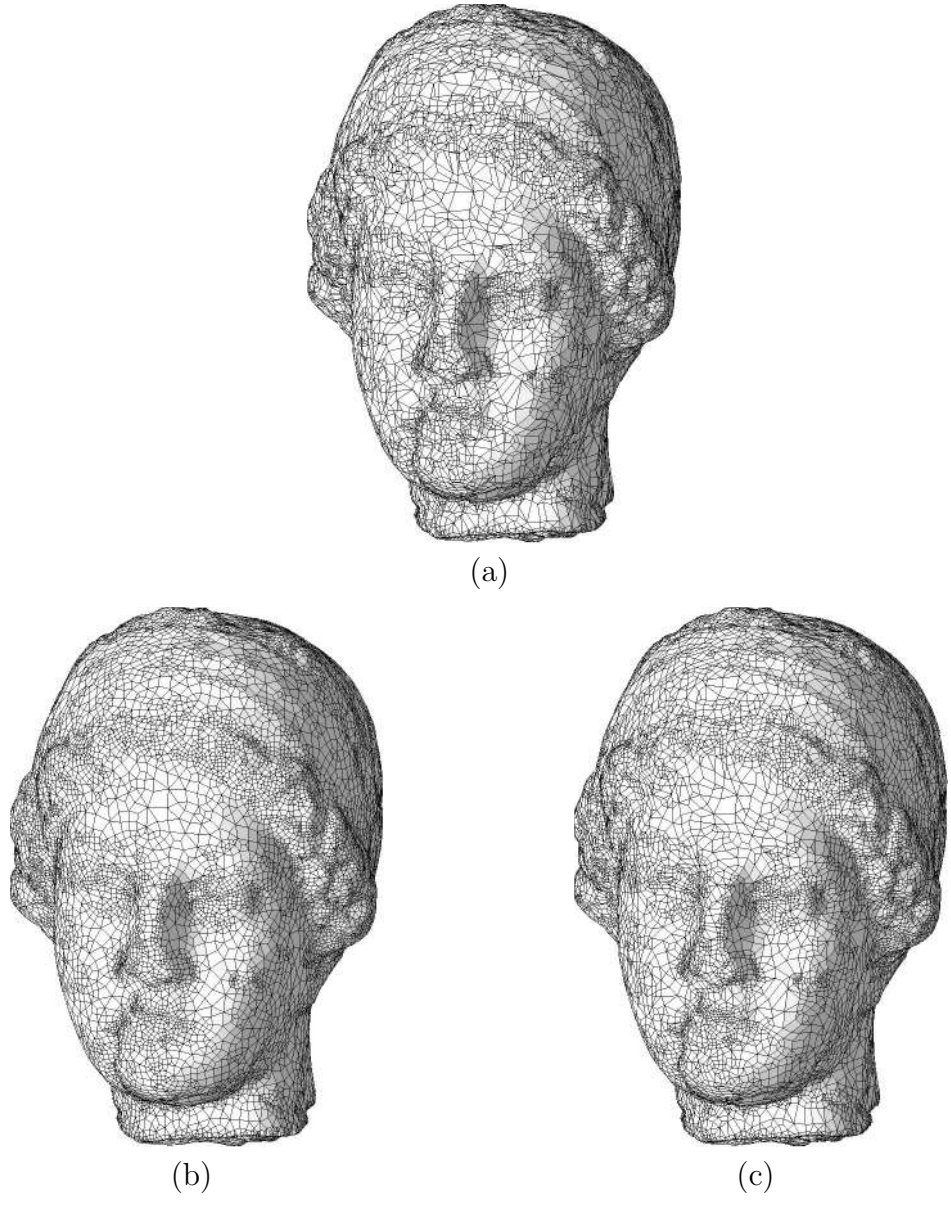


Fig. 10. (a) Mixed mesh of the Igea artifact (from Cyberware, Inc.), (b) Mesh optimized with CN objective function, (c) Mesh optimized with RJ objective function.

global objective function at each mesh vertex. The second method was the two stage reference Jacobian matrix or RJM based method, which was designed to improve the mesh quality as well as minimize the movement of vertices from their original locations. In the first stage of this method, minimization of a local condition number obective function at each mesh vertex was used only to calculate the locally optimal, virtual position for that vertex. These virtual or reference positions were then used to form a global RJM based objective function which was also minimized by minimizing a local component of the global function at each vertex.

$\bar{\mathcal{K}}$			Original	CN Opt.	RJ Opt.
1.0	–	1.5	15984	23341	22021
1.5	–	2.0	6071	310	1537
2.0	–	3.0	1370	1	88
3.0	–	4.0	142	0	5
4.0	–	5.0	33	0	1
5.0	–	7.5	40	0	0
7.5	–	10.0	8	0	0
10.0	–	15.0	3	0	0
15.0	–		1	0	0

Table 3
Histograms of Normalized Average Condition Number in Original and Optimized Meshes for Igea artifact (Figure 10).

Measure	CN Opt.	RJ Opt.
Hausdorff Distance	0.5%	0.2%
Max. Vertex Movement	3.1%	1.3%
Ave. Vertex Movement	0.5%	0.2%

Table 4
Quantitative measures of the change in the mesh and discrete surface characteristics for CN optimization and RJ optimization for Igea artifact (Figure 10); distances are presented as a percentage of the problem size

The procedure has been successfully tested on a number of complex triangular and quadrilateral surface meshes. Several quantitative measures were presented to show that both types of optimizations do not distort the surface much. The RJM optimization strategy improves the mesh quality considerably but also keeps the vertices of the original mesh close to their original positions. On the other hand, the global condition number based optimization can cause considerable movement of the vertices from their original positions in order to provide a small improvement in mesh quality beyond what is possible by the RJM based method.

Future work will attempt to extend the procedure to general polygonal meshes.

7 Acknowledgments

The work of Rao V. Garimella and Mikhail J. Shashkov was performed at Los Alamos National Laboratory operated by the University of California for the US Department of Energy under contract W-7405-ENG-36. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness. The work of Patrick M. Knupp was funded by the Department of Energy's Mathematics Information and Computational Sciences Program (SC-31) and was performed at Sandia National Laboratory. Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the US Department of Energy under contract DE-AC04-94AL85000.

The authors also acknowledge use of software tools from the Scientific Computation Research Center, Rensselaer Polytechnic Institute, Troy, NY that enabled the testing of the ideas presented in this paper.

References

[1] D. Field, Laplacian smoothing and Delaunay triangulations, Communications in Applied Numerical Methods 4 (1988) 709–712.

[2] L. Freitag, M. Jones, P. Plassmann, An efficient parallel algorithm for mesh smoothing, in: Proceedings of the Fourth International Meshing Roundtable, Albuquerque, NM, 1995, pp. 47–58.

[3] P. Zavattieri, E. Dari, G. Buscaglia, Optimization strategies in unstructured mesh generation, International Journal of Numerical Methods in Engineering 39 (1996) 2055–2071.

[4] P. Knupp, Winslow smoothing on two-dimensional unstructured meshes, in: Proceedings of the Seventh International Meshing Roundtable, Park City, UT, 1998, pp. 449–457.

[5] S. Cannan, J. Tristano, M. Staten, An approach to combined laplacian and optimization-based smoothing for triangular, quadrilateral and quad-dominant meshes, in: Proceedings of the Seventh International Meshing Roundtable, Dearborn, MI, 1998, pp. 479–494.

[6] T. Zhou, K. Shimada, An angle-based approach to two-dimensional mesh smoothing, in: Proceedings of Ninth International Meshing Roundtable, pp. 373–384.

[7] A. Khamayseh, C. Mastin, Computational conformal mapping for surface grid generation, Journal of Computational Physics 123 (2) (1996) 394–401.

[8] B. Soni, S. Yang, Nurbs-based surface grid redistribution and remapping algorithms, Computer Aided Geometric Design 12 (1995) 675–692.

[9] P. Knupp, S. Steinberg, The Fundamentals of Grid Generation, CRC Press, 1993.

[10] P. Frey, H. Borouchaki, Geometric surface mesh optimization, Computing and Visualization in Science (1998) 113–121.

[11] P. M. Knupp, Achieving finite element mesh quality via optimization of the jacobian matrix norm and associated quantitites. Part I - a framework for surface mesh optimization, International Journal for Numerical Methods in Engineering (48) (2000) 401–420.

[12] K. Giannakoglou, P. Chaviaropoulos, K. Papailiou, Boundary-fitted parametrization of unstructured grids on arbitrary surfaces, Advances in Engineering Software 27 (1996) 41–49.

[13] M. S. Floater, Parametrization and smooth approximation of surface triangulations, Computer Aided Geometric Design 14 (1997) 231–250.

[14] A. Sheffer, E. de Sturler, Parameterization of faceted surfaces for meshing using angle-based flattening, Engineering with Computers 17 (3) (2001) 326–337.

[15] M. Desbrun, M. Meyer, P. Alliez, Intrinsic parameterizations of surface meshes, in: Proceedings of Eurographics 2002 Conference, Saaebrücken, Germany, 2002.

[16] I. Guskov, An anisotropic mesh parameterization scheme, in: Proceedings of the Eleventh International Meshing Roundtable, Ithaca, NY, 2002, pp. 325–332.

[17] M. Meyer, H. Lee, M. Desbrun, A. Barr, Generalizing barycentric coordinates for irregular n-gons, Journal of Graphics Tools To appear.

[18] J. Nocedal, S. Wright, Numerical Optimization, Springer, 1999.

[19] Reklaitis, Ravindran, Ragsdell, Engineering Optimization - Methods and Applications, Wiley Interscience, 1983.

[20] T. J. R. Hughes, The Finite Element Method: Linear Static and Dynamic Finite Element Analysis, Prentice Hall, 1987.

[21] M. J. Shashkov, P. M. Knupp, Optimization-based reference-matrix rezone strategies for arbitrary Lagrangian-Eulerian methods on unstructured grids, in: Proceedings of the Tenth Anniversary International Meshing Roundtable, Newport Beach, CA, 2001, pp. 167–176, `http://math.lanl.gov/~shashkov`.

[22] C. Hirt, A. Amsden, J. Cook, An arbitrary Lagrangian-Eulerian computing method for all flow speeds, Journal of Computational Physics 135 (2) (1997) 203–216, originally published in Journal of Computational Physics, 1974.

[23] D. Benson, Computational methods in Lagrangian and Eulerian hydrocodes, Computer Methods in Applied Mechanics and Engineering 99 (1992) 235–395.

[24] P. M. Knupp, L. G. Margolin, M. J. Shashkov, Reference Jacobian optimization-based rezone strategies for arbitrary Lagrangian Eulerian methods, Journal of Computational Physics 176 (2002) 93–128.

[25] P. Cignoni, C. Rocchini, R. Scopigno, Metro: Measuring error on simplified surfaces, Computer Graphics Forum 17 (2) (1998) 167–174.

[26] D. P. Huttenlocher, G. A. Klanderman, W. J. Rucklidge, Comparing images using the Hausdorff distance, IEEE Transactions on Pattern Analysis and Machine Intelligence 15 (3) (1993) 850–863.