SMOOTHING OF SHELL MESHES ON FACETED B-REP GEOMETRY

Harold J. Fogg[1] Jonathan E. Makem[1]

[1]*Meshing & Abstraction, Simulation and Test Solutions, Siemens Digital Industry Software, 112 Hills Road, Cambridge, UK. jonathan.makem@siemens.com*

ABSTRACT

Most smoothing methods are designed to move nodes in the interior of a domain whilst holding the nodes on boundaries fixed. By incorporating a stage to move nodes along edges a further improvement in mesh quality may be achieved. This is particularly true of meshes on complex B-rep models with many thin faces and curved edges. The work presented here describes a two-stage process for smoothing shell meshes on the edges and faces of a faceted B-rep geometry. The first stage involves the global smoothing of the mesh to generally improve its quality by Laplacian smoothing along edges with tangent line constraints and variational smoothing on faces. The second local optimisation smoothing stage is designed to make additional local adjustments to the mesh by targeting a specific element quality metric subject to constraints. Again, nodes on edges as well as on faces are smoothed. The approach is demonstrated on a selection of geometries of varying complexity.

Keywords: mesh smoothing, mesh optimisation

1. INTRODUCTION

On models of industrial complexity even the most advanced mesh generators will produce imperfect initial meshes that require post processing before they are fit for purpose. In order for the particular simulation to be performed efficiently with fidelity to the underlying physics and with the appropriate accuracy the mesh must fulfil certain quality criteria. Unfortunately, the quality criteria are problem dependent and an acceptable mesh for one simulation may be inadequate for another. Certain criteria are almost universally required, such as non-negative element Jacobians, and others are treated as "good rules of thumb". Typically, element quality metrics such as skew, aspect-ratio, taper, warp, distortion etc. (see e.g. [1]) are required to fall within a specified range.

There are two broad categories of approach for mesh quality improvement: 1) Topological improvement methods such as edge swap, element merge, etc. 2) Smoothing where nodes are repositioned. Both are needed in practice. This paper presents smoothing methods in the second category. Two smoothing methods are outlined, the first to efficiently move all the nodes of the mesh to improve its general quality overall and the second to make slight local adjustments that improve specific element quality metrics which are difficult to target globally.

2. RELATED WORK

Ruiz-Girones et al. [2] presented a hierarchical iterative approach whereby a two stage smoothing and untangling procedure is used to move interior nodes as well as boundary nodes. The technique is applied to analytic CAD geometries and cannot be directly applied to polygonalised faceted representations of geometry. The Gauss-Seidel scheme that the authors use for computing and updating the position of the new node locations may be slow to converge on certain geometries where nodal perturbation is substantial. The single objective function that the authors use will not improve element quality measures which are not strongly correlated with mesh distortion such as Taper for example. The models used to demonstrate the effectiveness of the approach are relatively basic in comparison to the models shown in this work (number of edges and faces $\sim$100 versus $\sim$1k). An overall running time in the order of minutes quoted by the authors suggests that the technique is computationally

expensive.

Garimella et al. [3, 4] reported a procedure to improve the quality of complex polygonal surface meshes on faceted geometries. Again an iterative approach is used and the mesh vertices are repositioned using a non-linear optimisation process. More precisely, two methods are used in the optimisation procedure. The first approach minimises a global condition number and the second method uses a Reference Jacobian Matrix (RJM) to improve mesh quality whilst keeping nodal perturbation to a minimum. The authors don't report exact computation times but instead compare one approach against the other to evaluate computational efficiency as a percentage. All the geometries used are quite simple closed surfaces (although complex for the time) and in some cases the results do not honour the boundary edges of the geometry.

Shivanna et al. [5] proposed a parametrisation and projection-based technique for optimisation of quad shell meshes on underlying triangulated surfaces. The main limitation of the approach is that the nodal perturbation is restricted to surfaces with no infrastructure in place to support node movement along boundaries. The authors use an iterative scheme to update the nodal position on a node-by-node basis. The computational efficiency of the approach has not been evaluated.

Escobar et al. [6] described a Gauss-Seidel approach for tri mesh quality improvement by minimising an objective function derived from algebraic quality measures of the local mesh in the immediate vicinity of the node being perturbed. Constraints are imposed to ensure the objective function restricts the node within a feasible region. The authors construct a local parameter space via orthographic projection but this is not ideal in the vicinity of G^1 discontinuities. The approach is not applied to quad meshes and nodal perturbation along edges is not possible. The effectiveness of the method on complex geometries with many faces and edges is not demonstrated.

Gargallo-Peiro et al. [7] developed a continuous optimization procedure to improve the quality of meshes on parametrised CAD surfaces by smoothing and untangling. Initially, the optimisation process did not consider the prescribed element size so the authors also used the size-shape distortion measure that combines the previous distortion measures to produce a mesh that preserves a prescribed element size field and generates well-shaped elements. Further work is required to extend the untangling capability on such meshes. The approach isn't extended to smooth nodes on edges.

The work presented here outlines a practical method for smoothing large shell meshes on faceted B-rep models of industrial complexity, unlike other methods which are demonstrated on simpler CAD models. The smoothing of nodes on piece-wise linear geometry edges is achieved without using parametrisation derivatives which are not necessarily available. A complementary local optimisation smoothing method is also described which can be applied to nodes on geometry faces and edges to target specific element quality metrics. These methods are straightforward to implement and have reasonable execution times when run on large meshes of complex models.

3. FACETED B-REP GEOMETRY

In 3D solid modelling or CAE software such as Simcenter 3D [8] a boundary representation (B-rep) of the modelling geometry is used. This comprises topological components (faces, edges and vertices) and the connections between them, along with geometric definitions for those components (surfaces, curves and points, respectively). The geometric definitions may be continuous mathematical equations (*e.g.* splines, NURBS) or facetings where simply connected triangles represent geometry faces and geometry edges are described by polylines composed of triangle edges. In Simcenter 3D the modelling geometry is always converted to a faceted B-rep known as a 'polygon geometry' which is then simplified and de-featured using merging operations to eliminate artefacts and details below the mesh size. The mesh is then generated on the polygon geometry. The continuous geometry to which the polygon geometry approximates is inferred when needed by using a variety of numerical techniques including triangular Bezier patches and Hermite interpolation.

The majority of shell meshing algorithms essentially work in 2D and to apply them to 3D faces a bijective parametrisation is required. For faceted geometry faces a range of practical and robust methods are available [9, 10].

4. GLOBAL SMOOTHING

The global smoothing procedure consists of five steps as shown in Fig. 1. These are iterated over until the solution converges or the number of iterations reaches a prescribed maximum number. Pseudo-code for the procedure is given in algorithm 1.

Algorithm 1: Global smoothing pseudocode

```
for i in range (num iterations):
    active edges = edges not converged
    smooth active edges

    active faces = faces not converged
    for face in active faces:
        smooth face
```

Geometry edges and faces are marked as converged if

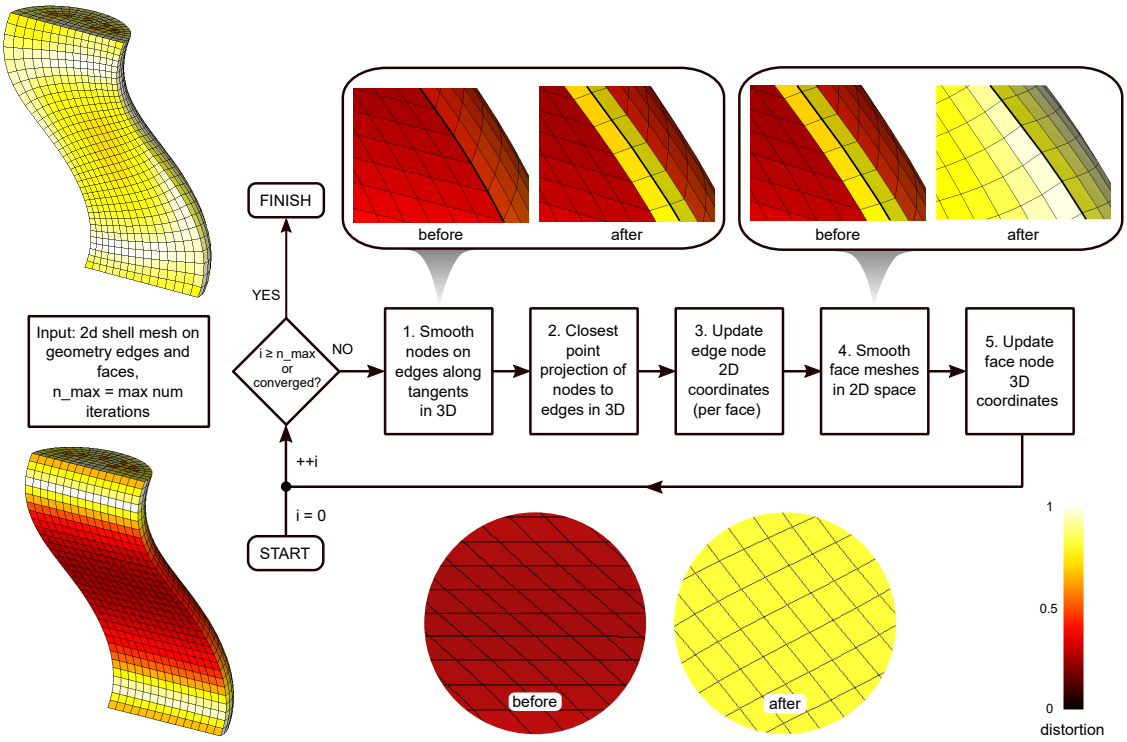

Figure 1: Global smoothing procedure

the maximum movement of a node in the previous iteration is below a small percentage of the local element size (*e.g.* 4%).

In Fig. 1 the meshes are coloured by element distortion [11], a general element quality metric for triangles and quadrilaterals where ideal elements have a value of 1 and degenerate elements a value of 0.

4.1 Smooth nodes on geometry edges along tangents in 3D

Laplacian smoothing is used with tangent constraints to smooth the nodes along a geometry edge. The unique solution is the distribution of node positions, $\mathbf{x}$, that minimises the sum of squared distances between neighbouring nodes (*i.e.* those that are connected by element edges),

$$\text{argmin}_{\{\mathbf{x}_i\}} \sum_{i=0}^{\#\text{mesh nodes}} e_i,$$

$$e_i = \sum_{j=0}^{\#\text{neighbours}_i} \frac{\alpha_{ij}}{2} ||\mathbf{x}_j - \mathbf{x}_i||^2 + \frac{\beta_i}{2} ||\mathbf{x}_i - \mathbf{x}_{i0}||^2. \quad (1)$$

The coefficient α_{ij} can be chosen to prioritise certain edges and also to reduce the dominance of long edges by setting the values as inversely proportional to the initial edge length squared. The second term of Eqn. (1) with coefficient β_i (e.g. =0.0005) is added to to penalise large displacements from the initial positions, $\mathbf{x}_{i0}$. This is a regularisation strategy. Since the initial positions are reset in each iteration nodes may move a long way from their pre-smoothing positions after a few iterations. Adding constraints to keep nodes on curved geometry edges would make the problem non-linear. Instead tangent line constraints are added to keep the nodes close to the geometry edges, but not exactly on them. The nodes are projected back to the edge in the subsequent step.

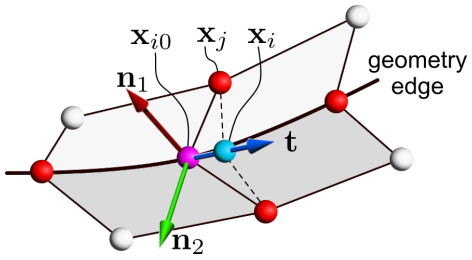

Figure 2: Laplacian smoothing of node on geometry edge

Tangents are approximated by the unitised displacements from the previous node to the next node, as show in Fig. 3 (a). The tangents that are computed are thus unaffected by small shape details of the ge-

ometry edge that are not captured by the initial mesh discretisation.

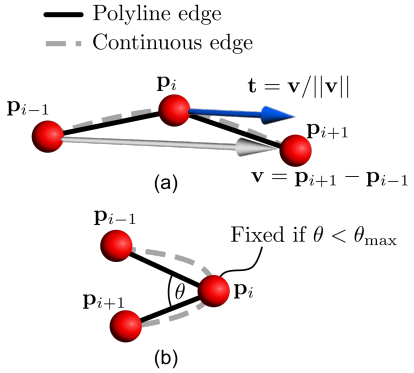

Figure 3: Approximating tangent vectors

If the angle between the previous node, the central node and the next node, θ, is below a tolerance, $\theta_{\max}$, then the node is fixed and not smoothed. Without doing this there is a risk that the local tangent approximation of the geometry edge is too crude and problems with mesh tangling could occur during the next projection stage. Experiments have found that using $\theta_{\max} = 80^\circ$ avoids such issues.

For a node on a geometry edge, as shown in Fig. 2, at initial position $\mathbf{x}_{i0}$ with tangent vector $\mathbf{t}$, Laplacian smoothing with a tangent constraint satisfies

$$\text{argmin}_{\mathbf{x}_i} e_i \tag{2}$$

$$\text{s.t. } g_{i1} = (\mathbf{x}_i - \mathbf{x}_{i0}) \cdot \mathbf{n}_1 = 0, \tag{3}$$

$$g_{i2} = (\mathbf{x}_i - \mathbf{x}_{i0}) \cdot \mathbf{n}_2 = 0, \tag{4}$$

where $\mathbf{n}_1$ and $\mathbf{n}_2$ are two mutually perpendicular unit vectors are perpendicular to the edge tangent vector $\mathbf{t}$. A possible choice for $\mathbf{n}_1$ and $\mathbf{n}_2$ are the the normal and bi-normal in a Frenet-Serret frame. Using Lagrange multipliers the Lagrangian is

$$L_i(\mathbf{x}_i, \lambda_1, \lambda_2) = e_i(\mathbf{x_i}) - \lambda_1 g_{i1}(\mathbf{x_i}) - \lambda_2 g_{i2}(\mathbf{x_i}). \tag{5}$$

The solution is an extremum,

$$\nabla_{\mathbf{x}_i, \lambda_1, \lambda_2} L_i = 0 \tag{6}$$

$$\Rightarrow \begin{cases} \frac{\partial L_i}{\partial \mathbf{x}_i} = \frac{\partial e_i}{\partial \mathbf{x}_i} - \lambda_1 \frac{\partial g_i 1}{\partial \mathbf{x}_i} - \lambda_2 \frac{\partial g_i 2}{\partial \mathbf{x}_i} = 0, \\ \frac{\partial L_i}{\partial \lambda_i} = -g_i = 0. \end{cases} \tag{7}$$

The equation $\frac{\partial L_i}{\partial \mathbf{x}_i} = 0$ can be expanded and simplified to

$$\sum_{j=0}^{\#\text{neighbours}_i} (\alpha_{ij} + \beta_i)\mathbf{x}_i - \sum_{j=0}^{\#\text{neighbours}_i} \alpha_{ij}\mathbf{x}_j + \lambda_1 \mathbf{n}_1 + \lambda_2 \mathbf{n}_2 = \beta_i \mathbf{x}_{i0}. \tag{8}$$

The nodes on edges may be smoothed one at a time in a local iteration scheme or alternatively the nodes on edges can be smoothed together in one shot in a global linear system. Equations (3), (4) and (8) give a linear system of 5 equations with 5 unknowns $(x_i, y_i, z_i, \lambda_{i1}, \lambda_{i1})$ for a single node. The nodal contributions can be assembled together into a global linear system for the entire mesh and decomposed by Schur complement to give a sparse system of dimension 5 times the number of free nodes. For a row corresponding to a free node the number of non-zero entries is three times the number of free nodes that are adjacent to the node.

4.2 Closest point projection of nodes to geometry-edges

A simple closest point algorithm is used to project the smoothed edge nodes back onto the polyline geometry edge. This algorithm has quadratic complexity so it becomes prohibitively expensive if the polyline has many points. If the polyline has more points than $O(10^2)$ then spatial trees (e.g. R-trees) can be used to reduce the expense of the algorithm.

After the global smoothing has been completed the nodes are lifted off the polyline geometry edges and repositioned on the inferred continuous geometry edge by cubic Hermite interpolation.

4.3 Update geometry edge node 2D positions (per face)

Since the 3D positions of the nodes on the geometry edges have been moved in steps 1 and 2 the 2D positions have to be updated. This can be done efficiently using the closest point projection data of step 2. The closest polyline segment has been identified and the 2D positions of the closest point along the segment can be calculated by linearly interpolating between the 2D positions of the facet vertices at the start and end of the segment. For every node on an edge its 2D positions of every connected face (1 for a free edge, 2 for a manifold edge, +2 for non-manifold connections) are updated. On seam edges (see e.g. Fig. 9) nodes have two corresponding 2D positions and both must be updated.

4.4 Smooth face meshes in 2D space

In response to the updated positions of the nodes on edges the interior nodes on the faces are smoothed with the edge nodes held fixed. All the interior nodes may be smoothed or to improve computational efficiency a subset of nodes in proximity to edges can be smoothed. The subset can be defined based on:

- adjacency: a specified number of layers from the edges (0 – no face smoothing, 1 – the neighbours

of the edge nodes, 2 – the neighbours of neighbours of the edge nodes, etc.), or
- distance: the nodes within an offset distance from the edges.

For the smoothing of the mesh in 2D the variational smoothing scheme of Knupp [12, 13] is used. It is assumed that there is a twice differentiable map, $\mathbf{x}$, for every local region of the mesh to go from 2D reference space (ξ, η,) to 2D physical space (u, v) (which is a face's parametric space). The element shapes in

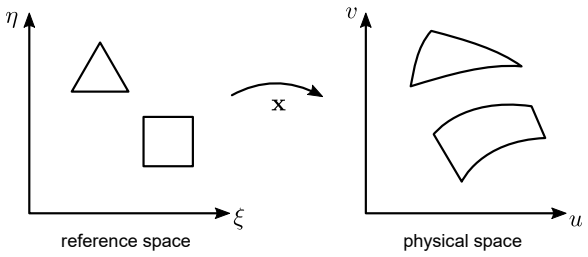

Figure 4: Reference to physical space map

reference space are ideal and unit in length, e.g. equilateral triangles and squares. A variational principle is a rule for evaluating an entire mesh to give a single real number. A smoothed mesh minimises some variation principle, typically of the form

$$I[\mathbf{x}] = \iint H d\xi\, d\eta. \tag{9}$$

The metric tensor, G, of the map from reference space to physical space and it governs the shapes of the elements in physical space,

$$G = \begin{bmatrix} g_{11} & g_{12} \\ g_{12} & g_{22}, \end{bmatrix} \tag{10}$$

$$g_{11} = \mathbf{x}_\xi \cdot \mathbf{x}_\xi \tag{11}$$

$$g_{12} = \mathbf{x}_\xi \cdot \mathbf{x}_\eta \tag{12}$$

$$g_{22} = \mathbf{x}_\eta \cdot \mathbf{x}_\eta \tag{13}$$

Of practical interest are variational principles that are expressed in terms of the metric tensor and relate to geometric properties. Some important integrands are listed in Tab. 1.

Principle	H
Length Squared	$\mathrm{tr}G$
Area Squared	$\det G$
Winslow	$\frac{\mathrm{tr}G}{\det G}$
Orthogonality	g_{12}^2
Combinations	$\sum_i \alpha_i H_i$

Table 1: Some metric tensor based variational principle integrands

The inverse map $\mathbf{x}^{-1}$ at a node n is assumed to equally spread the neighbouring nodes m_i around a unit circle as shown in Fig. 5. Finite difference stencils are

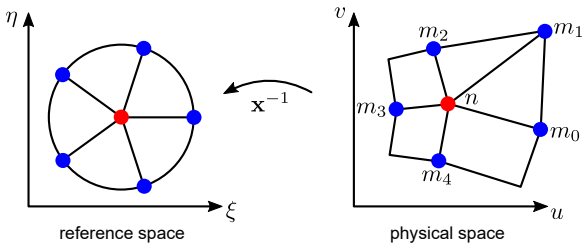

Figure 5: Reference to physical space inverse map

derived for approximating the first and second derivatives of the map, $\mathbf{x}_u$, $\mathbf{x}_v$, $\mathbf{x}_{uu}$, $\mathbf{x}_{uv}$ and $\mathbf{x}_{vv}$, by linear expressions using the positions of the neighbouring nodes. For the case of a node with four adjacent elements the diagonal nodes of adjacent quad elements are also used in the stencil.

As described in [13], the Euler-Lagrange equation which minimises the variational principle can be expressed in the form

$$T_{11}\mathbf{x}_{uu} + T_{12}\mathbf{x}_{uv} + T_{22}\mathbf{x}_{vv} = 0. \tag{14}$$

T_{11}, T_{12} and T_{22} are 2×2 matrices with entries that only involve $\mathbf{x}_u$ and $\mathbf{x}_v$ and partial derivatives of H with respect to g_{11}, g_{12},g_{22} and $\det G$. The method of Picard iterations can be used to solve this non-linear system by treating T_{11}, T_{12} and T_{22} as constants to give a linearised system of dimension 2×2. This is solved and followed by an update of the matrices T_{11}, T_{12} and T_{22}. This is repeated for a few iterations until the solution converges.

The node contributions of Eqn. (14) can be assembled into a global system for a face. The dimension would be the number of free nodes if the variational principle is uncoupled with respect to the u and v coordinates (Length Squared and Winslow) and twice that otherwise. For a row corresponding to a particular free node the number of non-zero entries is the number of free nodes that are adjacent to the node for an uncoupled variational principle and twice that otherwise.

Combining variational principles is useful to achieve a compromise between the properties controlled by each. No ideal combination exists in general so empirical testing is required to find effective weights for the kinds of meshes that are expected. It has been found that for quad-dominant body panel meshes an effective combination of variational principles is Length Squared, Area Squared and Orthogonality with the weights 1.0, 0.01 and 0.01 at a reference length of 100. Using a dominant weight for Length Squared means that the numerical scheme is stable, whilst the small weight for area prevents the mesh inverting and the small weight for Orthogonality improves the element shapes without causing instability.

4.5 Update face node 3D positions

The 3D positions of the face nodes that were smoothed in the previous step must be updated. The bijective parametrisation is stored simply as 2D positions at every facet vertex. The standard operation for mapping a point from 2D to 3D involves first finding the containing or closest facet and then using the three 3D positions of its vertices to interpolate the 3D point. The first task can be done efficiently using bounding boxes and spatial hashing.

4.6 Results

Example results of the global smoother are shown in Figs. 9, 7 and 6. Their timing data is given in Tab. 2 where the following notation is used:

- $n_{\text{edge/face}}$ – the number of nodes on edges/faces that were smoothed.
- t_{init} – the time to initialise data structures.
- $t_{\text{edge/face}}$ – the time taken to smooth the nodes on edges.
- t_{proj} – the time taken to do all node projections on edges.
- t_{total} – the total time taken.

All results were generated using a desktop machine with a Intel Xeon CPU E3-1240 v6 at 3.70GHz (8 CPUs) with 64GB RAM. The algorithms are executed in serial.

A swept hex mesh example is shown in Fig. 6. The wall-faces (i.e the four-sided faces connecting the source and target faces) are transfinite mapped meshed which results in a high degree of distortion in this case. By applying the global smoother to the wall-face shell meshes before the generation of the volume mesh the quality of the final hex mesh is greatly improved.

In Fig. 7 the global smoother is applied to a quad dominant mesh on an Opel body panel [15] that was generated in Simcenter 3D for a transient dynamic crash simulation. Elements in some thin fillet regions which have been mapped meshed exhibit large distortion in the initial mesh. This is significantly alleviated by smoothing. Similarly, the skewed elements in the series of concentric annulus faces around an inner hole are effectively straightened after smoothing.

Fig. 9 shows an example of smoothing a quad mesh on a curved pipe face. The pipe face was constructed by revolving a circular profile curve around two orthogonal axes. A seam edge connecting the edge loops was automatically added in the CAE software to facilitate the parametrisation of the face. Each node along the seam has two 2D positions and both must be smoothed and updated. Global smoothing significantly improves the mesh quality and in this case a

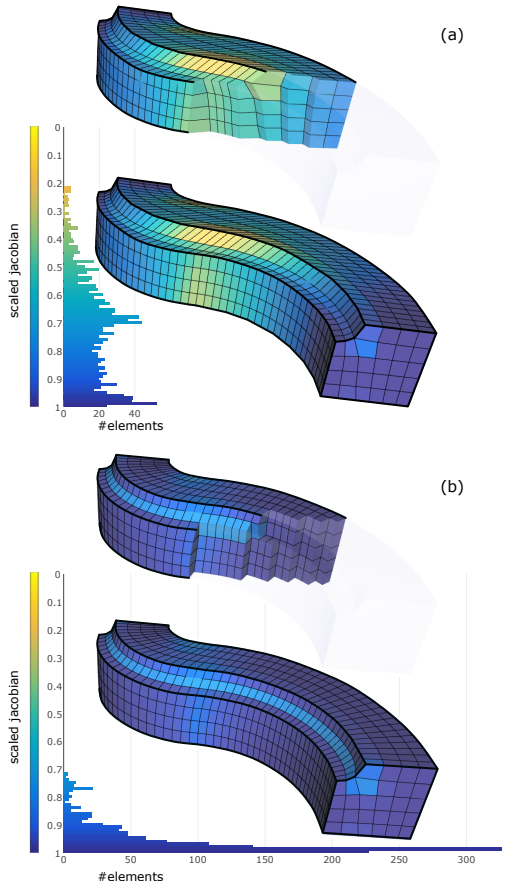

Figure 6: Swept hex mesh without (a) and with (b) smoothing the wall face meshes. Mesh sections are shown on top. (Images generated using Hexalab [14].)

global system scheme is found to be more effective than a local iteration scheme. This will be further discussed in the next section.

4.6.1 Local iteration versus global system

There are two possible schemes for performing the global smoothing of the mesh:

1. Local iteration – Sequentially visit each free node and solve its new position treating the adjacent nodes as fixed. If the node position is updated immediately it is a *Gauss-Seidel* scheme.
2. Global system – Solve a global system (iteratively if it is non-linear) for the positions of all free nodes together in one shot.

The global system can be solved using a sparse system solver, e.g. [16]. In our implementation a SuperLU direct solver is used for matrices with less than 10^6 entries. For larger matrices the memory usage may

model	#edges	#faces	n_{edge}	n_{face}	t_{init}	t_{edge}	t_{face}	t_{proj}	t_{total}
S Offset *(Local iteration) (Fig. 1)*	6	4	136	1260	0.19s	0.32s	1.01s	0.08s	1.61s
Sweep *(Local iteration) (Fig. 6)*	15	6	247	782	0.16s	0.28s	0.47s	0.14s	1.06s
Body panel 1 *(Figs. 7 and 8 (a))*	3138	1140	22120	39472					
- Local iteration					12.29s	9.78s	5.87s	0.26s	28.20s
- Global system					12.32s	4.18s	5.73s	0.42s	22.65s
Body panel 2 *(Fig. 8 (b))*	1170	423	8912	16308					
- Local iteration					4.64s	3.37s	2.33s	0.12s	10.47s
- Global system					4.88s	1.25s	1.57s	0.14s	7.84s
Body panel 3 *(Fig. 8 (c))*	2118	789	16245	28070					
- Local iteration					8.19s	6.15s	3.57s	0.23s	18.13s
- Global system					8.70s	2.38s	1.63s	0.30s	13.01s
Pipe (Fig. 9)	3	1	121	1452					
- Local iteration					0.19s	0.07s	0.77s	0.01s	1.03s
- Global system					0.20s	0.12s	1.43s	0.02s	1.77s

Table 2: Timings for global smoothing.

approach the heap memory limits and an indirect bi-conjugate gradient stabilized solver is used which is up to ten times slower.

The two main advantages of the local iteration Gauss-Seidel scheme are the ease of its implementation and that it is straightforward to ensure that no element quality regressions (e.g. negative Jacobian) are caused by smoothing. This is done by performing checks after a local smoothing iteration and avoiding updating the position if necessary. On the other hand the global system scheme may be faster. In Fig. 8 (top) the convergence behaviour of global smoothing using both the local iteration and global system schemes on three large meshes (∼10k) on body panel geometries with many faces and edges (∼1k) are shown. Both schemes reach convergence in less than 7 global iterations. In Fig. 8 (bottom) it can be seen that the distortion quality improvements of the mesh are similar. However, as reported in Tab. 2 the execution times are marginally faster for the global system scheme.

For cases where smoothing must move the nodes relatively far from their initial positions to reach convergence the global system scheme may outperform the local iteration scheme. For example in Fig. 9 a quad mesh on a pipe face is smoothed to convergence using a local iteration scheme and a global system scheme. A comparison of the histograms for element distortion indicates that the local iteration scheme converged prematurely and that a superior result was produced by using the global scheme.

Overall for most B-rep models resembling body panels with many edges and faces the advantages of the local iteration scheme outweigh those of the global iteration scheme. The ability to maintain valid-in valid-out meshes at every stage of the smoothing process is of major benefit to the algorithm robustness in the general case.

5. LOCAL OPTIMISATION SMOOTHING

The global smoothing procedure is capable of large-scale changes to the mesh to improve its general quality. It can do this robustly and quite efficiently. However, some element quality metrics might be important to the analyst but they may not be expressible in the form a variational principle in Eqn. (9). Moreover, they might not be strongly correlated with the chosen variational principle that is used for global smoothing. To deal with this problem another phase of smoothing is applied to the mesh.

Local optimisation smoothing is designed to solve the following problem: For a *target* element quality metric and a specified range, smooth the nodes of a mesh so that the fewest elements have a target metric value outside of the range (i.e. they *fail*). An alternative objective could be to minimise (or maximise) the highest (or lowest) target element quality metric value. Constraints can also be added to keep other *constraint* element quality metrics within their specified ranges.

An element quality metric is a function that evaluates the node positions of an element to give a value. Many are routinely used for assessing mesh quality such as skew, warp, jacobian etc. [1]. Often their response surface plots have discontinuous features which can cause problems in gradient based optimisation methods.

Taper is an element quality metric that often sees many failures as it is not tightly coupled with more intuitive element quality metrics such as min/max angles and aspect ratio. Using the NX Nastran definition [1] (other slightly different formulae exist), the taper of a quad element $ABCD$ is computed by

$$taper = \frac{A_{\max} - Q}{Q}, \tag{15}$$

where $A_{\max}$ is the largest of the triangle areas ABD, BCA, CBD and DAC and $Q = 0.5 \times$ total quad area. It does not apply to triangle elements. The ideal taper value is 0.0 and the worst value is 1.0 so it should be minimised in optimisation. For certain structural

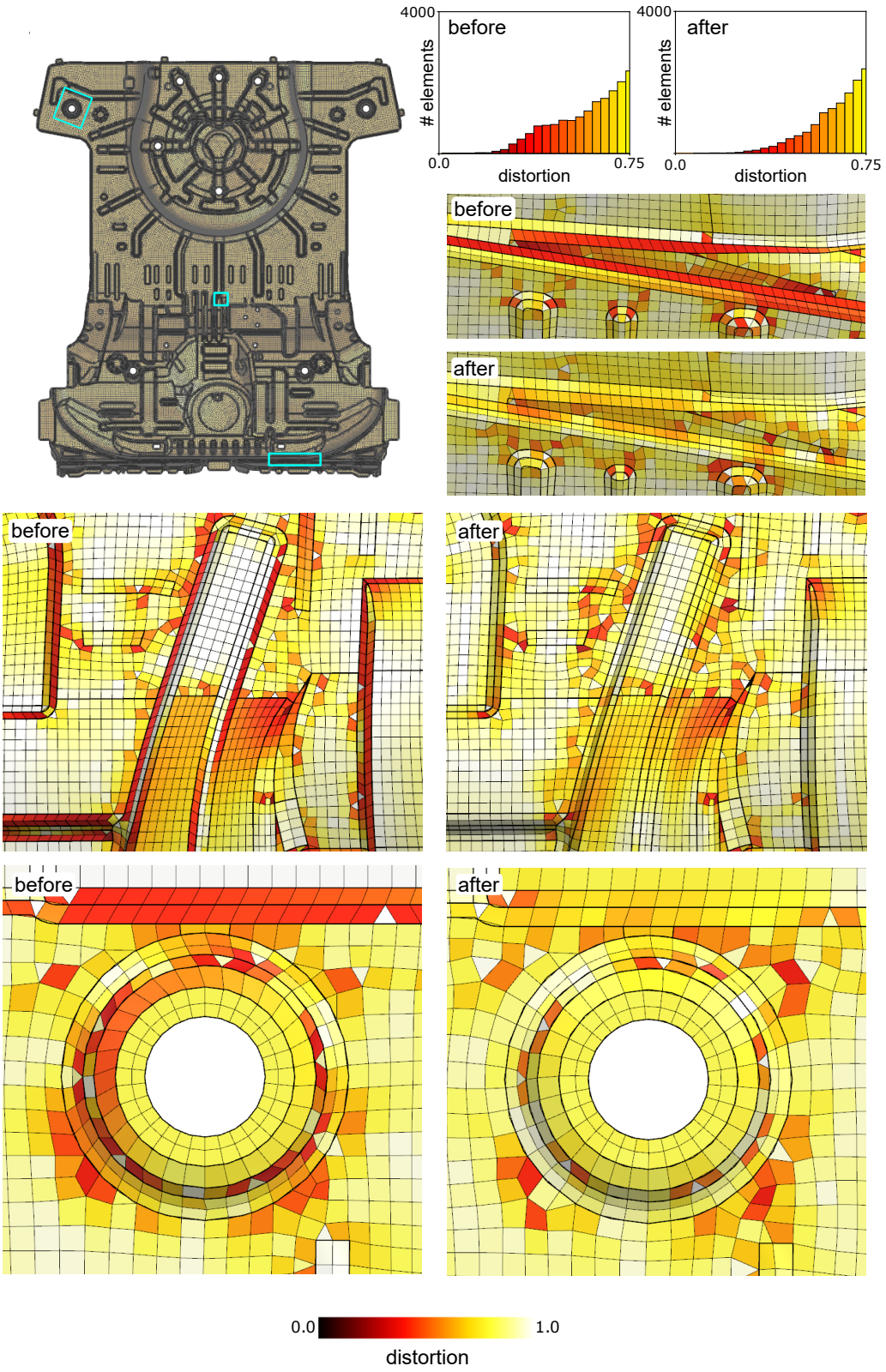

Figure 7: Global smoothing of a quad dominant shell mesh on an body panel.

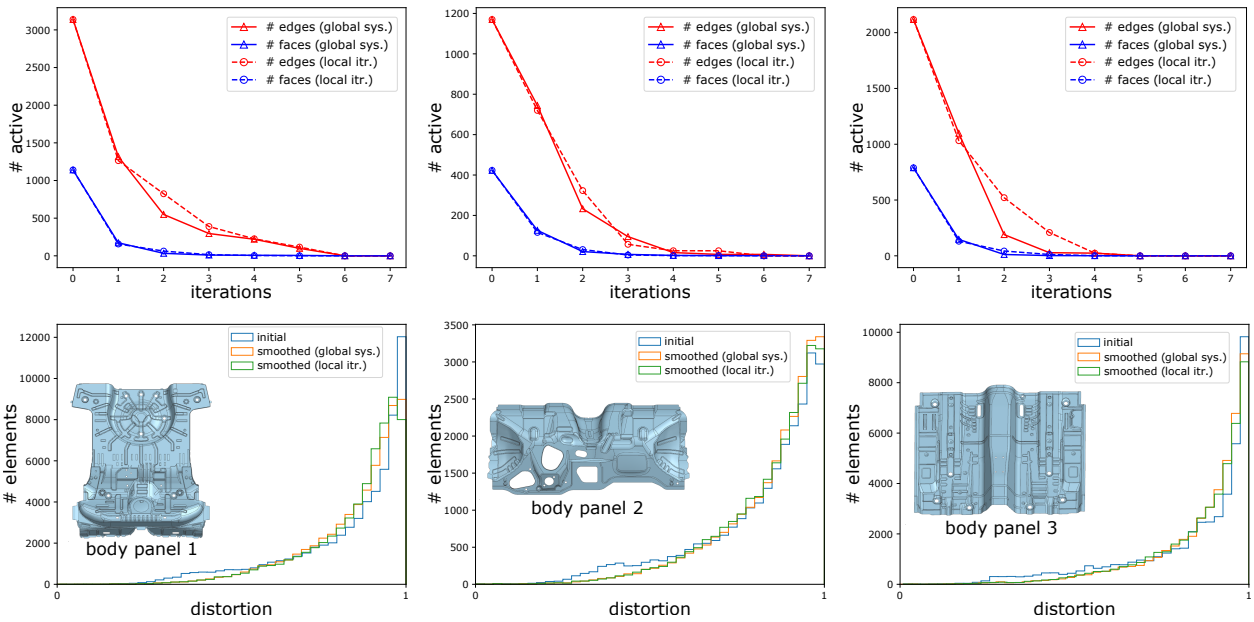

Figure 8: Comparison between the global system and local iteration methods for three body panels. Graphs showing the number of unconverged (active) edges and faces per iteration for global system and local iteration methods (top). Histograms showing mesh distortion improvements (bottom).

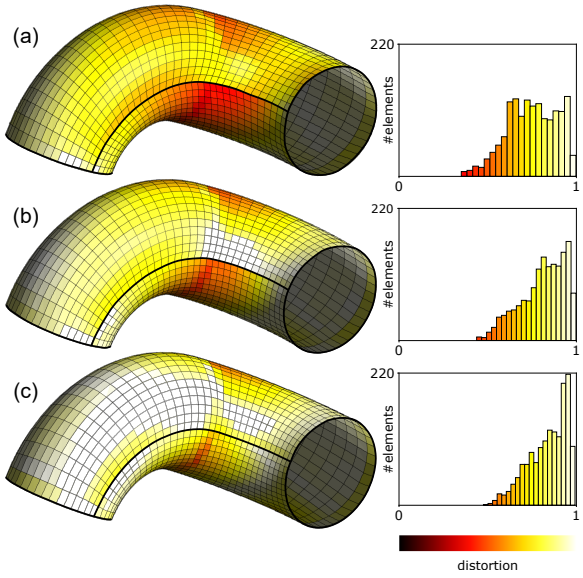

Figure 9: Smoothing a quad mesh on a pipe model. (a) Initial mesh. (b) Smoothed mesh using local iteration scheme. (c) Smoothed mesh using global system scheme. The histograms show the distributions of element distortion.

mechanics analyses it is required that the taper values of all quad elements are less than 0.5.

Local optimisation smoothing is performed in a iteration scheme. The whole process is outlined in algorithm 2.

Algorithm 2: Local optimisation smoother

```
for i in range(0, n_max):
    get failing elements
    if no failing elements:
        break
    for node in failing elements:
        optimise node position
```

First all the elements that fail the target quality metric are identified. Next all the positions of the nodes of the failing elements are optimised with respect to their one-rings (adjacent elements and nodes). Nodes on faces are visited first before nodes on edges. This is done repeatedly for a fixed number of iterations or until there are no failing elements.

Algorithm 3 describes the process for node optimisation.

Algorithm 3: Optimisation of a node position

```
if node on edge:
    get previous and next nodes
    locally parametrise edge (node position
        ↪ = pos(t))
    optimise t
    if t is valid:
        update node position
else: #node on face
    locally parametrise one ring (node
        ↪ position = pos(u,v))
    optimise (u,v)
    if (u,v) is valid:
        update node position
```

A node's position is optimised in a local parametric space that is established immediately before the optimisation process. For an interior node of a face

the parametrisation method that is used is an orthographic projection onto a local tangent plane. The angle-weighted normal of the one-ring elements is used for the normal. Minimal distortion is expected in this parametrisation as it is applied locally, except for extreme curvature cases. The search space of uv coordinates is the area of the projected one-ring elements.

Nodes on edges are parametrised locally in one of two ways with the choice depending on the relative positions of the node, its previous node and its next node along the edge. If the positions are almost co-linear then the edge is parametrised as a straight line, as shown in Fig. 10 (a). In this case the search space is $t \in (0, ||\mathbf{p}_{\text{next}} - \mathbf{p}_{\text{prev}}||)$. Otherwise a circle is fitted to

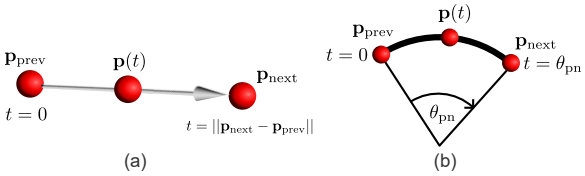

Figure 10: Local edge parametrisation. (a) Straight line parametrisation; (b) Arc parametrisation.

the three nodes and the parametric value corresponds to a subtended angle as shown in Fig. 10 (b). In this case the search space is $t \in (0, \theta_{pn})$.

The default objective function that is used is

$$f = \sum_{i=1}^{\#\text{ one-ring elements}} \text{target\_metric}(\text{element}_i)^2. \quad (16)$$

Through testing this has been found to be the most effective objective function for minimising the number of failing elements. If the overall goal is to to minimise (or maximise) the highest (or lowest) target element quality metric value then tests indicated that it is better to do all iterations using equation (16) as the objective function except for the final iteration for which the following objective function is used,

$$f = \min\big(\text{target\_metric}(\text{element}_i)\big) \mid i \in [1, \#\text{ one-ring elements}]. \quad (17)$$

Powell's method [17] is used as the optimisation method. No derivatives of the objective function are taken, which is convenient because the objective functions are typically not fully differentiable. However, it may not be as efficient as other non-linear optimisation methods that use derivatives. The initial node position is used as the initial starting point. Constraints are added to the optimisation method which are of the form:

$$\text{constraint\_metric}(\text{element}) < \text{threshold}. \quad (18)$$

For example, a constraint on the maximum corner angle of an element to be less than 150°.

5.1 Results

In Fig. 11 an example is shown of a close up region of a quad mesh between two connected faces. The local optimisation smoother was used to resolve two elements failing taper, i.e. elements with a taper element quality metric value greater than 0.5. The nodes of the elements were perturbed along the faces and along edges to improve taper.

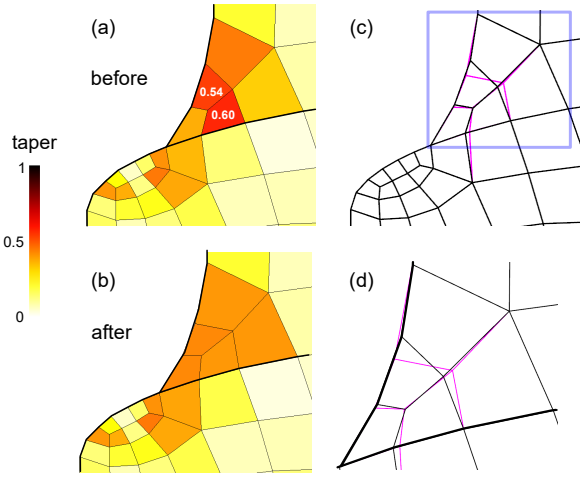

Figure 11: An example of optimisation smoothing for fixing taper failures (> 0.5). (a) The initial mesh with two failures; (b) The smoothed mesh with no failures; (c & d) Overlays of the meshes (initial mesh in black and smoothed mesh in magenta).

In Fig. 12 the local optimisation smoother was used to resolve taper failures in the quad dominant mesh of a body panel to which the global smoother was previously applied (Fig. 7). Three constraints are applied to keep the minimum and maximum corner angles and the minimum edge lengths of the elements within acceptable ranges. Twelve iterations are performed and the taper failures are reduced from 179 to 47, as shown in Fig. 13. Noticeably, most of the improvement is achieved in the first iteration and thereafter the convergence is slow. The timings are reported in Tab. 3. The mapping and inverse mapping times are negligible and most of the time is taken in the Powell optimisation solver, which typically converges in less than 5 iterations for each optimisation. The whole process is reasonably quick (3.2s) which demonstrates its effectiveness on a complex geometry.

6. LIMITATIONS

The global smoother has two theoretical limitations. Firstly, the variational principle is inconsistent for the separate stages of smoothing on edges and faces. Nodes on edges are smoothed by Laplacian smoothing (Eqn. (1)) in 3D whereas nodes on faces are smoothed according to another variational principle (Eqn. (9)) in

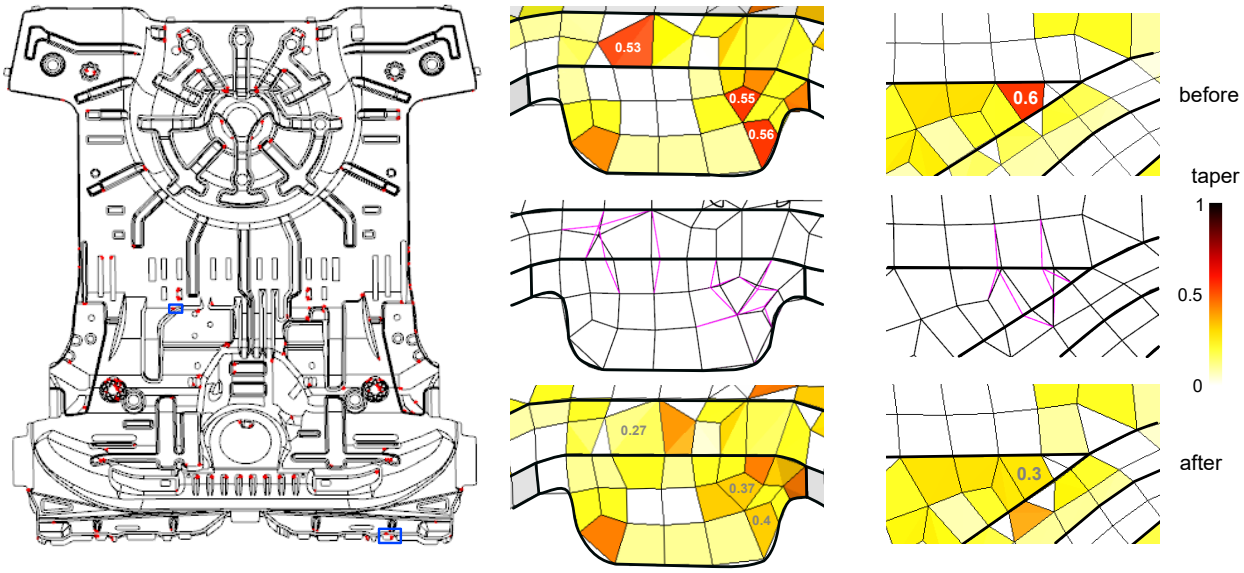

Figure 12: Optimisation smoothing of the quad dominant body panel mesh to resolve taper failures. To the left the whole body panel is shown with the the failing elements highlighted in red. To the right close ups of two regions are shown that demonstrate how the nodes are perturbed to resolve the failures.

Target metric	*Taper*
Constraint metrics	*Min Angle, Max Angle, Min Edge Length*
Num. edge optimisations	1621
Num. face optimisations	387
Edge optimisation time	0.712s
Face optimisation time	2.518s
Initialisation time	0.008s
Mapping time	<0.001s
Inverse mapping time	0.002s
Total time	3.240s

Table 3: Timings for the local optimisation smoother on the body panel mesh.

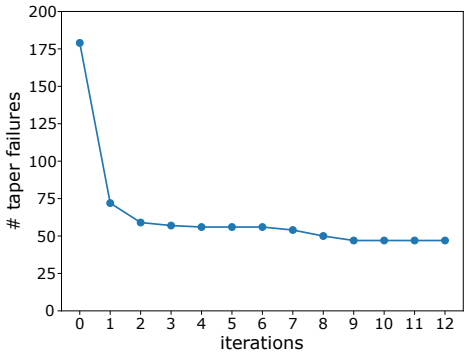

Figure 13: Number of elements failing Taper versus iterations in the optimisation smoother for the quad dominant body panel mesh.

2D. From a theoretical standpoint it may be preferable in edge smoothing to minimise the sum squared of the variational principles of the connected faces. However, this formulation would be more complex and less well behaved than the linear system of the employed formulation. In practice this inconsistency has not been found to be an issue and convergence is achieved in all tests within a few iterations.

Secondly, smoothing on faces is performed in 2D parametric space but the quality of the mesh is measured in 3D. Therefore any distortion of the parametrisation compromises the effectiveness of the smoothing. But typical parametrisations of faceted B-rep faces do not exhibit severe distortion and this limitation is not significant. Other authors have developed smoothing methods that can account for the parametric space distortion [2, 18]. But incorporating these adjustments would lead to non-linear equations which would be more difficult to solve. Also, they involve parametric derivatives which are not directly available in our case.

A practical limitation is that a varying target element size cannot be accommodated. This would be possible in the edge smoothing method by adding a penalty term to Eqn. (8) for deviation of edge length from the target element size. However, the system would then become non-linear so an iterative line-search solver would have to be used. Varying target element size could also be accommodated be in face variational smoothing by adding space-weighting terms to Eqn. (14). Implementing these will be the subject of future work.

7. CONCLUSION

In this work we have described a process for smoothing shell meshes on 3D faceted B-rep geometries. The approach comprises a two-stage method whereby the mesh is first globally smoothed for large-scale movement of nodes across faces and along edges to minimise a variational principle. Finally, the mesh is finedtuned using a local optimisation method to improve element quality metrics which may have non-smooth response surfaces and hence are not suitable for use in global objective functions. Constraints can be added to the local optimisation method to ensure that key element metrics are not regressed.

The global smoothing procedure involves iteratively smoothing nodes on edges and then smoothing nodes on faces. Smoothing on edges is done in 3D with tangent line constrains which avoids the need to parametrise the edges. Complex edges with G^1 discontinuities can be robustly handled with this approach. Although smoothing nodes on edges not a completely new concept it has not been widely adopted especially on faceted geometries. The smoothing on faces is done in parametric space and a variational method is used. The variational principle that is used may be adjusted to suit the mesh properties most desired. A weighted sum of the Length Squared, Area Squared, Orthogonality and Winslow principles with well chosen weights will tend to produce good quality inversion-free meshes.

Results indicate that both smoothers perform effectively on complex models with reasonable execution times. Both a local iteration scheme and a global system scheme have been tested in global smoothing. It has been found that the global system scheme can produce superior results when the nodes must travel relatively large distances along the faces. However, there is a theoretical risk that elements may be inverted although in practice this has not been observed. The advantage of the local iteration scheme is that checks can be easily performed after each local smoothing operation to ensure that certain element properties are preserved. For example it is important that element edge length is kept above a specified tolerance for transient dynamic analyses.

The combination of both smoothers gives a comprehensive process for generally improving the mesh quality whilst also targeting particular element quality metrics. The methods have been implemented in the commercial CAE software package Siemens Simcenter 3D.

References

[1] "NX Nastran User's Guide." `https://docs.plm.automation.siemens.com/data_services/resources/nxnastran/11/help/tdoc/en_US/pdf/User.pdf`. Accessed: 18-Jan-2022

[2] Ruiz-Gironés E., Roca X., Sarrate J. "Optimizing Mesh Distortion by Hierarchical Iteration Relocation of the Nodes on the CAD Entities." *Procedia Engineering*, vol. 82, 101–113, 2014. 23rd International Meshing Roundtable (IMR23)

[3] Garimella R.V., Shashkov M.J. "Polygonal surface mesh optimization." *Engineering with Computers*, vol. 20, no. 3, 265–272, Sep 2004

[4] Garimella R.V., Shashkov M.J., Knupp P.M. "Triangular and quadrilateral surface mesh quality optimization using local parametrization." *Computer Methods in Applied Mechanics and Engineering*, vol. 193, no. 9, 913–928, 2004

[5] Shivanna K.H., Grosland N., Magnotta V. "An Analytical Framework for Quadrilateral Surface Mesh Improvement with an Underlying Triangulated Surface Definition." *Proceedings of the 19th International Meshing Roundtable*. 2010

[6] Escobar J.M., Montero G., Montenegro R., Rodríguez E. "An algebraic method for smoothing surface triangulations on a local parametric space." *International Journal for Numerical Methods in Engineering*, vol. 66, 740–760, 2006

[7] Gargallo-Peiró A., Roca X., Sarrate J. "A surface mesh smoothing and untangling method independent of the CAD parameterization." *Computational Mechanics*, vol. 53, 587–609, 2013

[8] "Siemens Simcenter 3D." `https://www.plm.automation.siemens.com/global/en/products/simcenter/simcenter-3d.html`. Accessed: 21-Sep-2021

[9] Hormann K., Polthier K., Sheffer A. "Mesh Parameterization: Theory and Practice." *ACM SIGGRAPH ASIA 2008 Courses*, SIGGRAPH Asia '08, pp. 12:1–12:87. ACM, New York, NY, USA, 2008

[10] Jacobson A., Panozzo D., et al. "libigl: A simple C++ geometry processing library." `https://libigl.github.io/libigl-python-bindings/tut-chapter4/`, `https://libigl.github.io/`, 2018. Accessed: 28-Sep-2021

[11] Lee C., Lo S. "A new scheme for the generation of a graded quadrilateral mesh." *Computers & Structures*, vol. 52, no. 5, 847–857, 1994

[12] Knupp P.M. "Winslow Smoothing on Two-Dimensional Unstructured Meshes." *Proceedings of the 7th International Meshing Roundtable, Park City, UT*, pp. 449–457. 1998

[13] Knupp P. *The fundamentals of grid generation*. CRC Press, Boca Raton, 1993

[14] Bracci M., Tarini M., Pietroni N., Livesu M., Cignoni P. "HexaLab.net: An online viewer for hexahedral meshes." *Computer-Aided Design*, vol. 110, 24–36, 2019

[15] "Opel international." https://www.opel.com/. Accessed: 14-Oct-2021

[16] Guennebaud G., Jacob B., et al. "Eigen v3." http://eigen.tuxfamily.org, 2010

[17] Press W.H., Teukolsky S.A., Vetterling W.T., Flannery B.P. *Chapter 10. Minimization or Maximization of Functions, Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge University Press, USA, 3 edn., 2007

[18] Gargallo-Peiró A., Roca X., Peraire J., Sarrate J. "A distortion measure to validate and generate curved high-order meshes on CAD surfaces with independence of parameterization." *International Journal for Numerical Methods in Engineering*, vol. 106, 1100–1130, 2016