GROUND TRUTH CROSS FIELD GUIDED MESHER-NATIVE BOX IMPRINTING FOR AUTOMOTIVE CRASH ANALYSIS

Nilanjan Mukherjee

Meshing & Abstraction, Simulation and Test Solutions, Siemens Digital Industries Software, SIEMENS.

2000 Eastman Dr., Milford, Ohio 45150 USA, mukherjee.nilanjan@siemens.com

ABSTRACT

This paper proposes a meshing strategy and a series of related algorithms of both computation and process used to imprint box shapes around washer holes for automotive carbody crash analysis. A uniquely lightweight, mesher-native, two-dimensional imprinting methodology is described. This imprinting methodology is mesh size and definition sensitive and does not alter or even interrogate geometry or surface triangulation. Two other methods are proposed for face axis determination, namely a minimum oriented bounding box (MOBB) approach and one based on a ground truth frame/crossfield (GTFF/GTCF). Box shapes around washer holes are oriented according to these direction fields. Virtual face decomposition is used to multiblock surfaces into box-with-hole and their boolean mesh areas. While a templatized mesh generator is used for the box-with-hole faces, a hybrid multizone mesher and along with loop-paver are used for the latter. An exhaustively complex mesh quality analysis for automotive body panels clearly justifies the strength of the present approach.

Keywords: ground truth, framefield, crossfield, multizone, shape imprinting, unified modeling language

1. PROBLEM DEFINITION

Finite element analyses of engineering products have assumed prime importance in design validation for over four decades now. In the twenty-first century these analyses and mesh model preparation processes have become more specialized and automated. Geometry simplification for meshing and mesh generation technologies have also been challenged with scalability and variability requirements. As geometry cannot be over-simplified for analysis, a single meshing algorithm is not adequate in providing every edge and surface feature characteristics analysis accuracy demands. Subdividing or zoning out mesh areas and trying unique meshers on them seem to hold great promise in this regard.

In the automotive industry, in particular, an important product design validation or engineering analysis function is crash/collision analysis. This analysis is usually done on a discretized finite element meshed model of the entire car assembly, especially the subassembly that is called *body-in-white* (BIW). In the automotive industry, *body-in-white* (BIW) refers to the fabricated (usually seam and/or tack welded) sheet-metal components that form the car's body. Body-in-white is a stage of the car body prior to painting and before the moving parts (doors, hoods, fenders etc.), the engine, chassis sub-assemblies, and trim (glass, seats, upholstery, electronics, etc.) have been mounted. Structured and regular quadrilateral-dominant meshes (with the majority of face interior nodes connected to four elements, i.e., possess a valency of 4) are created on these body panels for a variety of finite element analyses. Such a BIW panel is shown in Figure 1a with the quad-dominant surface mesh in Figure 1b. The panel is meshed employing a plethora of meshing strategies including the one proposed in this paper.

Crash analysis of the digital finite element model is usually a nonlinear, transient dynamic structural analysis under shock velocity and/or impact loading. This is performed in order to predict the stress, deflection and rupture of the automobile in a crash/collision situation. For results/ predictions to be accurate, crash analysis requires the finite element quadrilateral-dominant mesh to have many distinct characteristics namely, high quality quasi-structured meshes on features and around bolt/washer holes. Figure 2 shows two quad-dominant meshes where the mesh around bolt holes is unstructured and unpatterned around the washer holes in Figure 2a and the most ideally analysis-oriented in Figure 2b.

This paper deals with the development of two-dimensional shape imprinting strategies inside the mesh generator such that meshes can be given distinct characteristics in local zones required by the analysis type. The paper focuses on three classes of strategies/algorithms - a) mesher-native shape imprinting and multiblocking strategies, b) mesh direction field computation and c) box orientation and 2d meshing algorithm selection for the decomposed face subdomains which are called *virtual faces*.

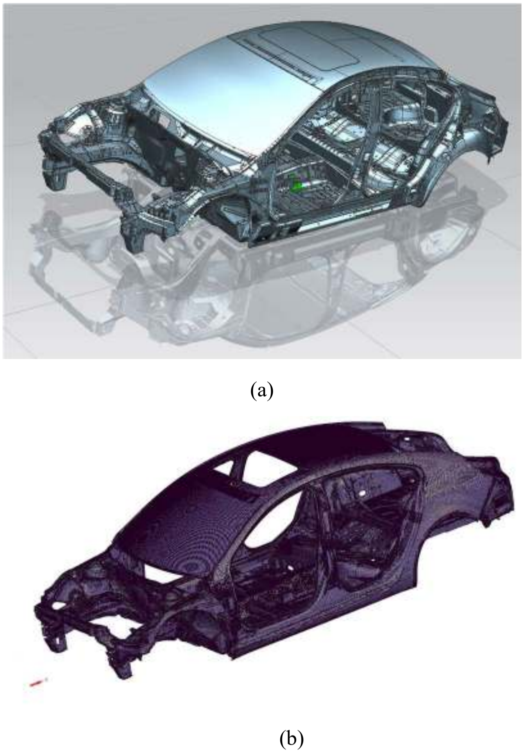


Figure 1. Quad-dominant mesh (b) on an automotive BIW panel (a) generated with the proposed meshing strategies

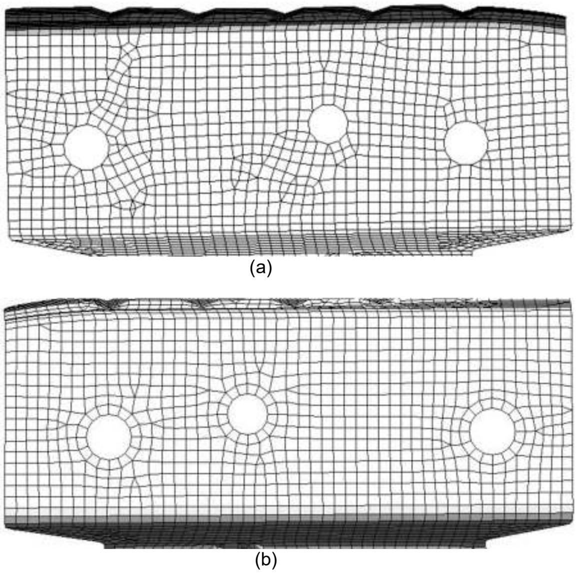


Figure 2. Mesh characteristic comparison of two quadrilateral meshes around washer hole features

2. PREVIOUS WORK

Shape-imprinting is classically seen as a CAD operator and is a well-researched and disseminated topic. While dealing with contemporary discrete geometry, shape imprinting research also finds interest in the field of computer graphics. However, when it comes to shape-imprinting tools embedded in finite element mesh generators, published research is quite limited. White and Saigal [1] and Clark et al [2] report some of the earliest investigations of CAD imprinting for the purpose of conformal mesh generation. Blacker's [3] "Cooper Tool" introduced the idea of shape imprinting in the sweep meshing context where source faces where internally imprinted on volume steps to continue sweeping in the subvolumes. In a similar vein but with completely different approaches Ruiz-Gironés, et al [4] used a least-squares approximation of affine mappings to decompose sweep meshing volumes with multiple source and target faces into subvolumes, while Cai and Tautges [5] used a novel edge-patch based imprinting technique on the sweep mesh layers to facilitate cage extractions. Lu [6] et al used a sketch-based approach with a geometric reasoning process to determine sweeping direction. Two types of sweepable regions are used which provides visual clues to the user in developing decomposition solutions. Outside of submapping and sweep meshing the author could not find any investigation on mesher-embedded, mesh-flow controlled shape imprinting for high quality quadrilateral mesh generation. Furthermore, no research paper or patent could be traced on the use of such techniques in the carbody crash analysis.

3. MESHER-NATIVE IMPRINTING STRATEGY

Shape imprinting is generally conceived as a CAD tool. Since geometry cannot be manipulated during meshing it is impossible to alter them during mesh generation. In this paper an attempt is made to shape-imprint inside a 2D surface mesh generator.

3.1 Design and Architecture for mesher-native shape imprinting

It is naturally imperative that a flexible, innovative architecture and design are prerequisite to the development of such functionality. This paper proposes an architecture and object-oriented design for imprinting 2D planforms in the parameter space of the face to mesh [7].

Appendix I describes the sequence of operations leading to the decomposition of the surface into these “zones” or "virtual faces". The UML (Unified Modeling Language) diagram shows the task begins with the creation of an Imprinter object for the face. The Imprinter creates an Imprint Shaper object. The latter has methods to either select a shape or use a user-driven shape. In the present context that user-driven shape is a square box. Next it orients the shape according to the face’s local or global mesh flow axis. The Imprint Shaper then imprints the chosen shape at the selected location on the parameter space. According to user-driven mesh controls, it applies element intervals or counts to the four sides of the box based on the template selected. The Imprint Shaper decides on the many parameters automatically, when user specification is missing. This complex algorithm will be reported in a separate paper.

The Imprint Shaper uses a mesher-native domain decomposer and mesh topology operators to create virtual vertices, virtual edges and virtual faces using a virtual topology engine similar to one reported by Makem et al [8]. Thus, each imprinted shape becomes a virtual face, and the residual area of the faces becomes another boolean virtual face. These virtual geometries comprise of points and lines, they do not involve surface tessellations. These are not geometries but rather lines and points to mark out sub-regions of a face in its 2d parameter space which we will call "virtual face". However, they follow a strict Eulerian topological system of definitions, connections and operations. Once the virtual geometries are formed they are stored in the FEM (Finite Element Model) database. Each face has links to its virtual sub-geometries and is free to use or ignore them when needed.

The Shaper object also selects and builds mesh template objects, especially for the box shape around holes. The selection of a particular template for a particular box is also being developed and can become the content of another invention disclosure later. Presently, templates are auto-selected based on the number of elements the user asks for around each hole.

Finally, the surface meshers access these virtual faces and generate 2d meshes on them which are finally mapped/transformed back to the 3D surfaces [9,10] after topological mesh cleanup [11] and smoothing [12]. The flowchart in Figure 3 describes the overall mesher-native imprinting and mesh generation strategies.

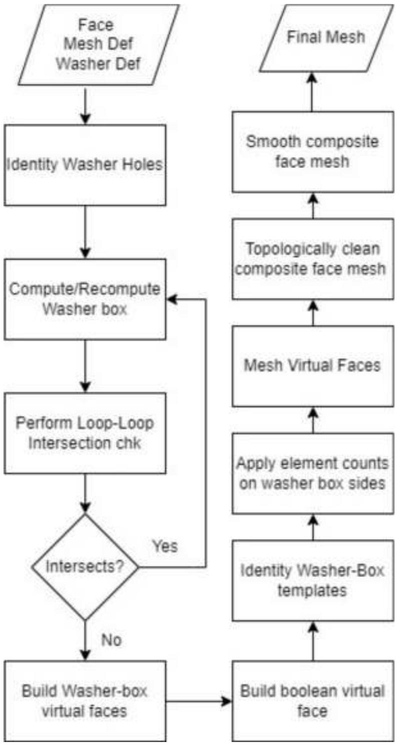

Face
Mesh Def
Washer Def
Identity Washer Holes
Compute/Recompute Washer box
Perform Loop-Loop Intersection chk
Intersects?
Yes
No
Build Washer-box virtual faces
Build boolean virtual face
Identity Washer-Box templates
Apply element counts on washer box sides
Mesh Virtual Faces
Topologically clean composite face mesh
Smooth composite face mesh
Final Mesh

Figure 3. Overall algorithm flowchart

3.2 Mesh Imprinting Box-with-Hole Shape

Face interior holes are regions of devout interest for structural analysts. Structure joining/fastening happens at these circular holes. Joints and fasteners like bolts, rivets, pins, locks pass through these holes. Many power and energy transferring and load bearing members like shafts, rods etc. are lodged at these sites making them potential areas of catastrophic stress, buckling, failure and long-term crack initiation. Finite element analyst always desires a model where the mesh around the holes is regular, structured and is made up of finite elements which deviate minimally from their best shapes (element included angle) – 60 deg. for triangles and 90 deg. for quads. Creating a box-shape around these holes, orienting them correctly with respect to reference direction leads to a well-patterned mesh. Such functionality is of great interest and importance to almost all structural engineers but most crucially the automotive crash analyst. Weldments, fasteners, bolts etc. are the most susceptible elements during car crash. Naturally, during finite element analysis great care is taken to reduce analysis error in these areas. Consequently, the shape and nature of the quadrilateral-dominant mesh in these localities assume paramount importance.

3.3 Shape Imprint driven Multiblocking

Surface meshes on carbody panels require all-quad, two-layered, patterned meshes around washer pads. Consequently, a shape imprint based multiblocking or face decomposition strategy becomes a natural choice.

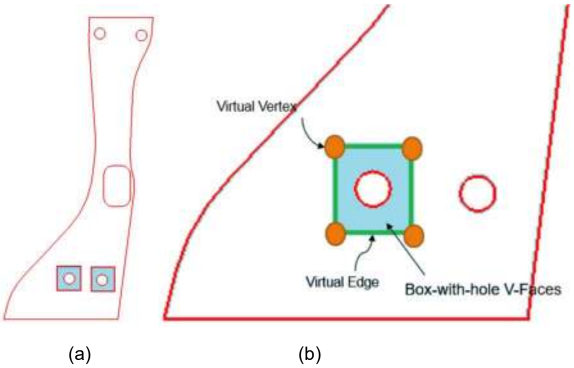


Figure 4. Face with washer holes multiblocked into box-with-hole virtual faces (a) detailed in (b)

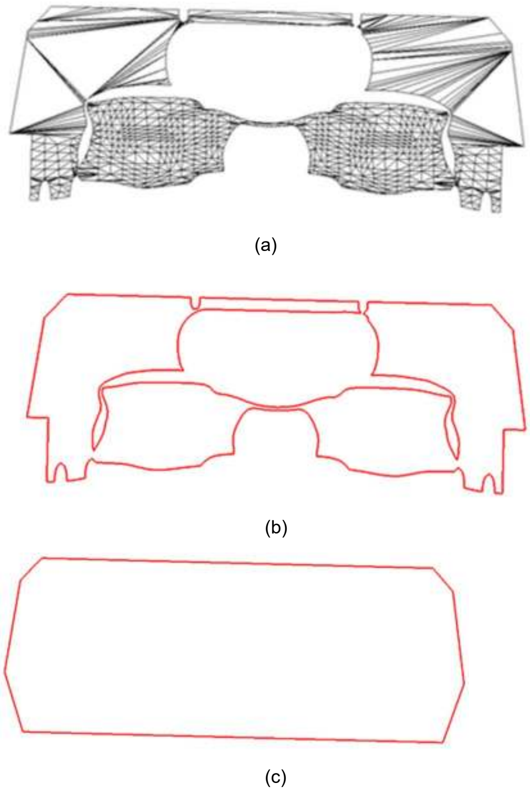
Figure 4 depicts an example where a body panel face with washer holes is box imprinted first. This mesher-native 2D decomposition technique breaks the face into three virtual faces or mesh areas - the two blue box-with-hole (BWH) virtual faces shown in Figure 4a and a residual region or BWH boolean face (white). Figure 4b illustrates typical virtual topology elements like virtual vertex and virtual edge which are 2d points and lines used to define a virtual face following a traditional Eulerian topology framework.

4. MESH DIRECTION FIELDS

Establishing a mesh flow direction for most classes of industrial problems is important. The mesh generated on the parts needs to approximately follow a direction. Therefore, it is equally important, especially for quadrilateral meshes, to orient the imprinted shape in this direction such that the imprinted shape (and thus, its local mesh) aligns with the global mesh. This paper proposes methods for boundary-aware direction vector computation for faces. It can be done in multiple ways. However, the three most common scenarios encountered by product users are automotive use cases, general mechanical/electronic use and aerospace applications. The most challenging amongst them is automotive BIW meshing.

4.1 Method I: Minimum Oriented Bounding Box based (MOBB)

Non-feature faces in an automotive body panel cover 50~90% of the body panel surface area. These faces are meshed either with the Multizone mesher [13] or the CSALF-Q (loop-paver) mesher [14]. For this class of faces a user-driven crash analysis global direction vector might be specified. If the crash vector is not specified, a natural mesh flow direction vector is computed for each face based on their shapes.



(a)

(b)

(c)

Figure 5. Sample tessellated face (a) , its 2D parameter space (b) and the convex hull of the face outerloop (c)

A method of determining a singular mesh flow direction vector in 2D is based on the natural shape of the face boundary in 2D. This technique is most general, robust and can serve most of the general mechanical and electronic industry finite element analysis requirements.

A sample face is shown in Figure 5a. The outer loop of the face in a flattened 2D domain is shown in Figure 5b. Meshing happens in this 2D domain or parameter space of the face. The face outer loop is first discretized with nodes. It thus makes a discretized closed polygon (Figure 4b). Next a 2D Convex Hull is created (Figure 5c) for the 2D face using the Gift-Wrap or Andrew algorithm [15]. A well-known rotating caliper method [16] is used to compute the minimum oriented bounding box of all orientations of the convex hull (Fig. 6).

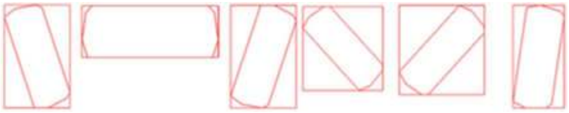


Fig. 6. A set of 2D oriented bounding boxes for different orientations of convex hull

The minimum oriented bounding box (MOBB) as shown in Figure 7, is defined by the oriented bounding box amongst all which has the minimum area. This means this bounding box encapsulated the 2D polygon which represents the discretized face loop best. The X-axis of this bounding box, described by the green 2D vector in Figure 7 defines the constant mesh flow direction vector. Angle α is the angle between the MOBB X-axis VMOBBx (green vector) and the global 2D axis of the face. In this particular case α = 4.4 deg. In order to ensure the orientation axis is dependable, a heuristic is set in terms of the area ratio A_r defined by the ratio of the face outer loop area and the MOBB area. This is defined in equation 1. An area factor threshold of 2.2 is obtained heuristically from the study of a complete carbody (Figure 1b) analysis.

$$\boldsymbol{A_r} = \frac{\boldsymbol{A_{lp}}}{\boldsymbol{A_{MOBB}}} \leq \boldsymbol{A_{FT}} \qquad (1)$$

where A_{LP} = 2d area of the face outer loop
A_{MOBB} = Area of the minimum oriented bounding box and
A_{FT} = Area factor threshold = 2.2

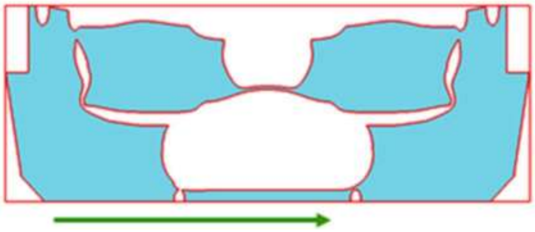


Figure 7. The Minimized Oriented Bounding Box for the face with the corresponding axis vector VMOBBX

4.2 Method II: Ground Truth Frame / Cross Field based (GTFF/ GTCF)

The face axis based on the MOBB (Minimum Oriented Bounding Box) of the face in its 2D parameter space is not always reliable for every single face of the geometry meshed. When the area factor threshold as defined by equation 1 is exceeded, a single face axis vector for shape imprinting and mesh generation is no longer reliable. To be able to orient the boxes reliably and locally on these faces without axis, a novel method is thus proposed.

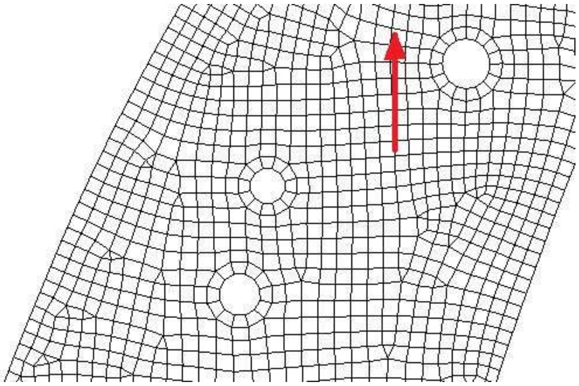


Figure 8. Axis-less face in 2D parameter space (u,v) where the boxes are imprinted at incorrect angles leading to a misoriented-flow quad mesh.

Figure 8 shows a detailed section of a face with many washer holes for which a single reliable mesh flow axis cannot be established using a MOBB face axis. Accordingly, the boxes, in the absence of any mesh flow axis gets created following the default uv-axes of the parameter space of the face. The red vector shows the V-axis of the parameter space along which all boxes are placed by default. Accordingly, the local mesh around the washer holes does not align with the overall mesh on the face. Such meshes are unsuitable for crash analyses. If one is lucky, the uv-axes of the face might be orthonormal to the crash direction. In such cases these can be used for box-imprinting. It is obvious, however, from even a casual review of the final mesh, the three boxes need to be oriented in different directions, so they align with the mesh better. In order to achieve this a novel algorithm (Algorithm I) is developed.

Algorithm I: Local Axes Determination for Box Orientation

1. First all face inner loops are suppressed and a coarse quad-dominant mesh (mostly quads with about 5% or less triangles) is generated.
2. A non-conforming Voxel mesh field is generated in the background for point tracking
3. A 2D local Frame field [17] is generated on the barycenter (centroid) of every element of that coarse mesh and transferred to element nodes.
4. The global field vector problem is addressed by solving the governing Laplace equation
5. A global crossfield direction field is constructed from the global frame field direction vector field
6. The imprintable boxes are placed atop the coarse quad-dominant mesh and the global crossfield field.
7. The elements containing each corner of each box are found by searching the box-packed field.
8. From each element containing a box corner the crossfield vector is interpolated at the corner point.
9. The box-corner crossfield vectors are averaged; each box is oriented to align with the average crossfield vector.

STEP 1 : A quad-dominant coarse background mesh is first generated on the face using a modified version of the CSALF-Q [14] mesher which will be used to generate the final mesh. The thought behind this is to determine an approximate mesh flow field for the "would-be" mesh on the entire face with inner features suppressed, so the boxes can be oriented in a reasonably accurate manner and can align well with the final mesh. Figure 9 shows the background mesh on the face with inner loops suppressed. The only face loop considered is the outerloop.

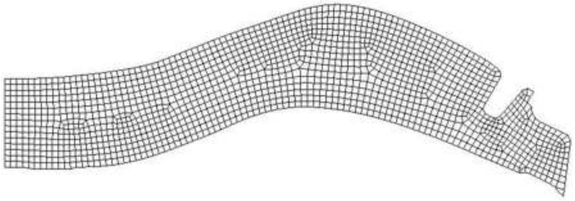


Figure 9. Background coarse quad mesh over the entire face represented by the outerloop only

STEP 2 : A non-conforming Voxel mesh field is generated next in the face parameter space. This field, shown in Figure 10, comprises non-conforming, unconnected and overlapping Voxel cells. Each cell represents the bounding box of one mesh element. Figure 10 illustrates a close-up view of the non-conforming Voxel mesh field near the upper side of the face. Equation 2 denotes, in a nutshell, the

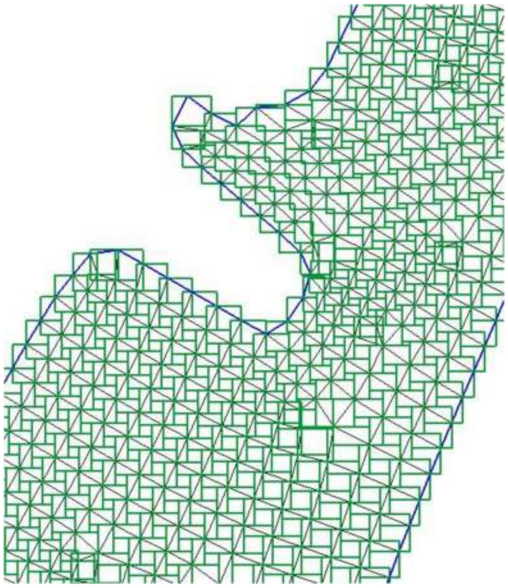


Figure 10. Close-up of the non-conforming Voxel mesh field of the face

non-conforming Voxel mesh (VX) for a 2D mesh with N elements as a union of the bounding areas of each element

$$\boldsymbol{VX} = \cup_{i=1}^{N} \boldsymbol{B}_i[\boldsymbol{O}(\boldsymbol{x}_i, \boldsymbol{y}_i,), \boldsymbol{\Omega}(\boldsymbol{x}_i, \boldsymbol{y}_i,)] \quad \forall\, \boldsymbol{x}, \boldsymbol{y} \ \in \boldsymbol{N} \quad (2)$$

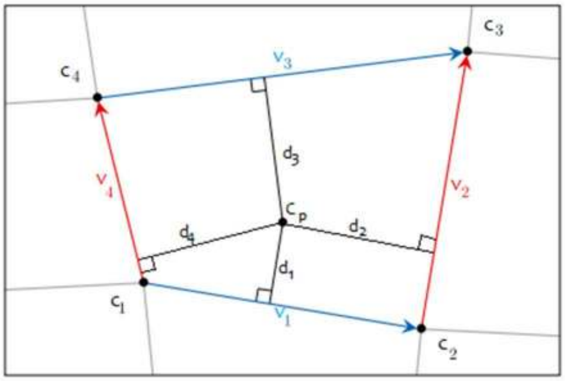


Figure 11. Side vectors and distance parameters used to construct element frame field vector

STEP 3 : Following a recent investigation on ground-truth frame field vector for a quadrilateral mesh [17], the element local frame field F_{uie} and F_{vie} vectors are computed for an i-th element.

$$\boldsymbol{F}_{uie} = \frac{\boldsymbol{d}_3}{(\boldsymbol{d}_1 + \boldsymbol{d}_3)}.\boldsymbol{v}_1 + \frac{\boldsymbol{d}_1}{(\boldsymbol{d}_1 + \boldsymbol{d}_3)}.\boldsymbol{v}_3 \qquad (3a)$$
$$\boldsymbol{F}_{vie} = \frac{\boldsymbol{d}_4}{(\boldsymbol{d}_2 + \boldsymbol{d}_4)}.\boldsymbol{v}_2 + \frac{\boldsymbol{d}_2}{(\boldsymbol{d}_2 + \boldsymbol{d}_4)}.\boldsymbol{v}_4 \qquad (3b)$$

Equation family 3 describes the computation used for the element local frame field vector at the centroid of the C_p element in terms of its opposite pair edge directions and the perpendicular distances of the sides from the element centroid. For a triangle, it is first split into 3 quads by joining the 3 corners of the triangle to the barycenter of the element. Next, the same vector formulation is used for the 3 quads and they are averaged out at the barycenter of the triangle. The element centroid frame field vectors are next transferred to its corner nodes. Each corner node thus gets the vector of each element it is connected to. The vector at each node is thus averaged.

STEP 4 : The global problem of the frame field can be expressed as Laplace's equation of steady state heat conduction in two-dimensions as shown in equation 4a. The 2D vector field (u) and F is a functional that needs to be minimized over the field. Dirichlet boundary conditions (4b) are applied and the equation is solved with a length-weighted iterative solver. The resultant global frame field vector field of the mesh is shown at the mesh nodes in Figure 12. This is representative of the flow-field of the mesh in terms of a frame field. A close-up detail of the GT frame field is shown in Figure 12b. Although the problem is still not globally solved, the nodal averaged vectors are beginning to indicate at what would be the mesh flow field in the end .

STEP 5 : From the smoothed GT frame field a GT crossfield is constructed directly by orthogonalizing the nodal vectors. A crossfield vector ($\boldsymbol{v}_i$) comprises a cross of four orthogonal vectors meeting at a point. Such a vector set can be described by equation 4c and is illustrated in Figure 13.

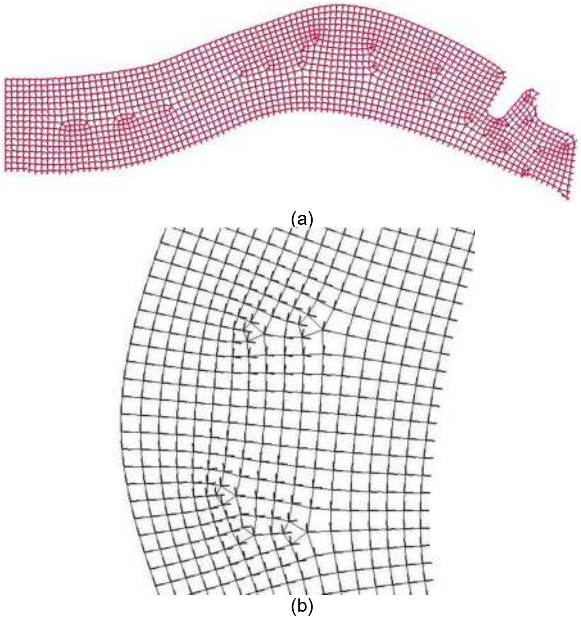


Figure 12. Laplace solved final frame field of the mesh (a) on the face outerloop including a close-up detail (b).

$$\boldsymbol{F}(\boldsymbol{u}) = \Delta\boldsymbol{u} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \qquad (4a)$$
$$\boldsymbol{u}(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{u}_0(\boldsymbol{x}, \boldsymbol{y}) \qquad (4b)$$
$$\boldsymbol{v}_i = [\mathbf{cos}\,(\boldsymbol{i\theta}), \mathbf{sin}\,(\boldsymbol{i\theta})]^T \quad \boldsymbol{\theta} = \frac{\pi}{2} \quad \mathbf{0} \le \boldsymbol{i} \le \mathbf{3} \qquad (4c)$$

The crossfield is necessary here as it is a more simplistic representation of the mesh as its u & v axes are assumed to be orthogonal at each point in the field. This is not true for the frame field vector. Here, the problem at hand is that of orienting a box – an orthogonal shape – therefore a crossfield is more convenient than a frame field.

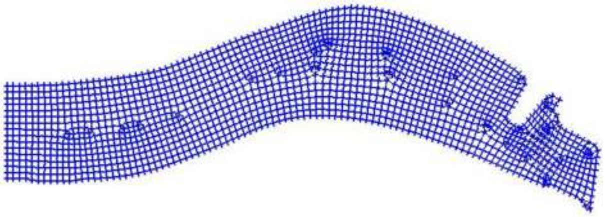


Figure 13. Solved GT crossfield of the mesh for the entire face.

5. BOX-WITH-HOLE ORIENTATION

The algorithm for orienting the imprintable box such that it aligns with the GT crossfield is discussed by steps 6 - 9 of Algorithm I.

STEP 6: The imprintable boxes around the washer holes are placed atop the coarse quad-dominant mesh and the global crossfield. These boxes and the elements of the background mesh that contain the box corners are shown in Figure 14.

STEP 7: The elements containing each corner of each box are found by searching the nonconforming voxel field described in equation 2. Each voxel cell corresponds to an element of the background quad mesh. A band search is performed which is quite efficient as it scans through the voxel cells by comparing the box corner coordinates with the min/max x and y of the voxel cells. This reduces the search to a small subset of elements. A point-in-polygon check is finally performed on these elements. As soon as one element is found the search stops. The background mesh element containing a box corner is thus found.

STEP 8: Each one of the originally computed box corners must lie in one element of the background mesh. The task at hand at this point is to compute the GT crossfield at each box corner. Each element of the background mesh stores the GT frame/cross field vectors at their corner nodes. Let us assume V_r and V_b are the horizontal axes of the original box (red) and the to-be-rotated box (black) respectively. This is illustrated in Figure 14. Red color refers to the original face orientation in 2d and therefore V_r is equivalent to the u-axis of the parameter face. In order to compute V_b the unit crossfield vectors of the 4 corners of the original imprintable box shape (red) must be found first. The background mesh elements containing these corners (as illustrated in Figure 14) are first determined by means of a linear search of the non-conforming voxel mesh field (Figure 10). The crossfield vector V_{cfck} at any box corners C_k can be interpolated from the nodal crossfield vectors of the element i containing the corner. If N_{ij} represents the four shape functions of a background isoparametric quadrilateral element i and V_{cfij} are the crossfield vector of its j-th corner the box corner field vector V_{cfck} is given by

$$\boldsymbol{V_{cfck}} = \sum_{j=1}^{4} \frac{1}{4} \boldsymbol{N_{ij}}(\xi_k, \eta_k).\boldsymbol{V_{cfij}} \qquad (5)$$

However, in the above equation the isoparametric element coordinates ξ_k, η_k at the box corner C_k are unknown. If the global coordinates of the 4 corners of element i are given by P_{il} (x,y) | l = 1,2,..4, box corner C_k can be expressed as

$$\boldsymbol{C_k} = \sum_{l=1}^{4} \boldsymbol{N_l}(\boldsymbol{\xi_k}, \boldsymbol{\eta_k}) . \boldsymbol{P_l} \qquad (6)$$

where C_k and P_l are known.

Equation (6) poses an inverse problem of solving for ξ_k, η_k. This is solved by a traditional Newton-Raphson technique. Figure 15 shows a close-up of a box corner $\boldsymbol{C_k}$ inside a background element E. The coordinates of $\boldsymbol{C_k}$ are known. In order to determine the element natural coordinates ξ_k, η_k at this point, we start off with paired functions $f(\boldsymbol{\xi}, \boldsymbol{\eta})$ and $g(\boldsymbol{\xi}, \boldsymbol{\eta})$ for a bilinear isoparametric element. The following equation can be conventionally written

$$\begin{bmatrix} \mathbf{J_{11}} & \mathbf{J_{12}} \\ \mathbf{J_{21}} & \mathbf{J_{22}} \end{bmatrix} \begin{Bmatrix} \Delta\boldsymbol{\xi} \\ \Delta\boldsymbol{\eta} \end{Bmatrix} + \begin{Bmatrix} \boldsymbol{f} \\ \boldsymbol{g} \end{Bmatrix} = \mathbf{0} \qquad (7a)$$

where the Jacobian derivatives are

$$\mathbf{J_{11}} = \frac{\partial f}{\partial \xi} \quad \mathbf{J_{12}} = \frac{\partial f}{\partial \eta} \quad \mathbf{J_{21}} = \frac{\partial g}{\partial \xi} \quad \mathbf{J_{22}} = \frac{\partial g}{\partial \xi} \qquad (7b)$$

Using a standard Newton-Raphson iterative procedure and upon further simplification the solution for i-th iteration can be written for a point k as

$$\boldsymbol{\xi_{ki}} = \boldsymbol{\xi_{ki-1}} - (\boldsymbol{J_{11i-1}f_{i-1}} + \boldsymbol{J_{12i-1}g_{i-1}}) \qquad (8a)$$
$$\boldsymbol{\eta_{ki}} = \boldsymbol{\eta_{ki-1}} - (\boldsymbol{J_{21i-1}f_{i-1}} + \boldsymbol{J_{22i-1}g_{i-1}}) \qquad (8b)$$

$$\boldsymbol{\varepsilon_n} = \left(\frac{\xi_{ki} - \xi_{ki-1}}{\xi_{ki-1}}\right)^2 + \left(\frac{\eta_{ki} - \eta_{ki-1}}{\eta_{ki-1}}\right)^2 \qquad (8c)$$
$$\boldsymbol{\epsilon_n} < \boldsymbol{\epsilon_{tol}} \qquad (8d)$$

An initial guess of $\boldsymbol{\xi_k} = \mathbf{0}$, $\boldsymbol{\eta_k} = \mathbf{0}$ is reasonably good. An error norm ε_n computed as shown in equation 8c is used to test convergence (8d). Solution is assumed to converge when the error norm falls below error tolerance ϵ_{tol}, usually in less than 10 iterations.

STEP 9: When ξ_k, η_k are inverse solved, the GT crossfield vectors are computed at the four box corners C_k and the box centroid C_g using equation 5. The five crossfield vectors are finally averaged to determine V_b

$$\boldsymbol{V_b} = (\sum_{k=1}^{4} \boldsymbol{V_{cfck}} + \boldsymbol{V_{cfg}})/\mathbf{5} \qquad (9)$$

As explained before V_r is constant over the face domain and is equivalent to the u-axis of the parameter space. The smallest angle φ, the angle between the red (original) and black (oriented) boxes, can thus be computed as

$$\boldsymbol{\varphi} = \mathbf{cos^{-1}}\left(\frac{\boldsymbol{V_r}.\boldsymbol{V_b}}{|\boldsymbol{V_r}|\,.|\boldsymbol{V_b}|}\right) \qquad (10)$$

The original box (box) is now rotated about its center so as to align with this averaged crossfield axis. This can be expressed as

$$\boldsymbol{B_n}(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{T}(\boldsymbol{\varphi}).\boldsymbol{B_o}(\boldsymbol{x}, \boldsymbol{y}) \qquad (11a)$$

where B_n denotes new box coordinates, B_o represents old box coordinates and T, the transformation matrix which is a function of angle of rotation/turn φ and is given by

$$\mathbf{T}(\boldsymbol{\varphi}) = \begin{bmatrix} \boldsymbol{cos\varphi} & \boldsymbol{sin\varphi} \\ -\boldsymbol{sin\varphi} & \boldsymbol{cos\varphi} \end{bmatrix} \qquad (11b)$$

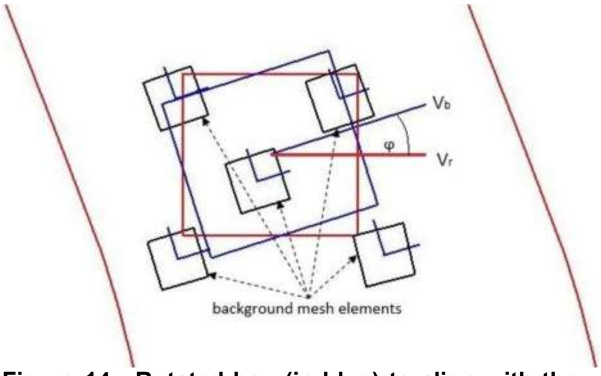


Figure 14. Rotated box (in blue) to align with the mesh flow

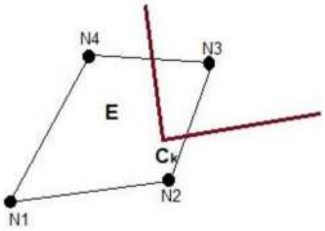


Figure 15. Box corner C_k falling inside a background quadrilateral element E with nodes N_1-N_4.

Figure 16 shows the virtual faces or mesh areas the face is decomposed into after imprinting the oriented boxes. The box-with hole faces (holes are not drawn inside the boxes for the sake of simplicity) are shaded in blue while the residual area, called the boolean virtual face is in pale green and red for the two comparison cases. Figure 16a illustrates the boxes not aligned with the GT crossfield. They are parallel to the v (or u) axis of the parameter space. In Figure 16b the virtual faces are rotated to align with the local crossfield. It is evident how the box-with-hole virtual faces align with the mesh flow in Figure 16b. All of them are nearly parallel to their nearest boundary tangent. The turning vector for each box is unique as determined from the crossfield.

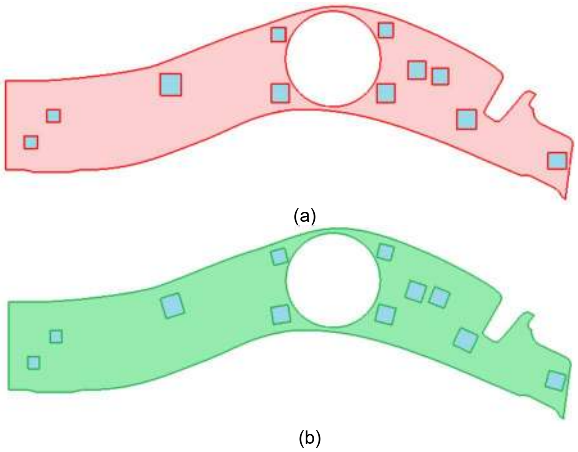


Figure 16. Decomposed faces with oriented box-with-hole virtual faces and the boolean virtual face

A section of the final mesh after rotating all imprintable boxes in this process is shown in Fig. 17b in comparison to the original unrotated box (Fig. 17a). A comparison of the two meshes by standard quad element quality measures and even by visual examination clearly indicates the mesh no longer "twists" around the holes. Furthermore, same studies show the mesh around the holes after proper rotation of the box is perfectly structured – meaning nearly all face interior quad nodes are tetra-valent i.e. connected to 4 neighboring quads and all element angles are fairly close to right angles. The number of resulting triangles is also lesser. It is to be noted that a small number of triangles result in these meshes for two reasons - firstly, the element size on the holes is user-controlled and typically lesser than the global mesh size which requires mesh size transition; secondly all-quad mesh transitions are not allowed in crash analysis meshes as they violate mesh flow. Hence, careful transitions need to be created in the mesh with a small number of triangles.

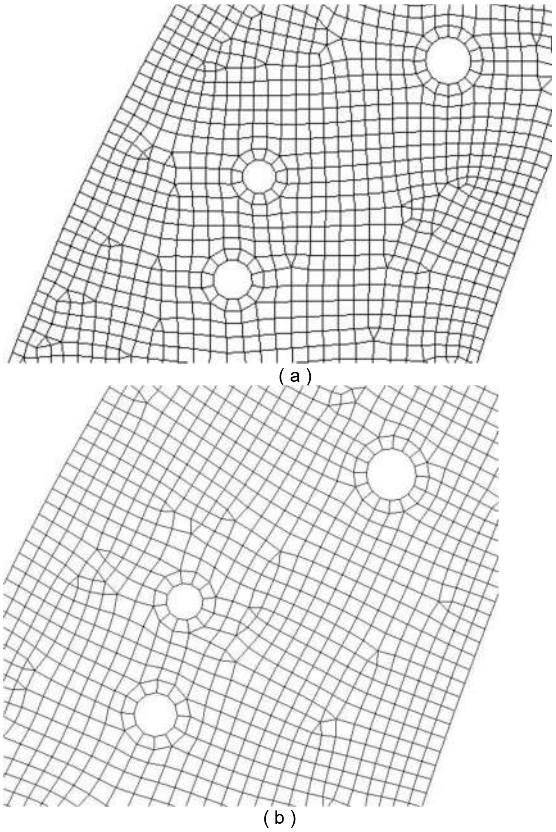


Figure 17. Improvement in mesh directionality after correctly aligning boxes with mesh flow on the face in Figure 8.

6. HOLE ORIENTATION INSIDE BOX

While the box-with-hole virtual face is oriented according to the mesh direction field, the circular hole loop also needs to be oriented with respect to the box. This becomes necessary to ensure the paved layer mesh around the hole has at least two element edges (diametrically opposite) parallel or perpendicular to the box edges. Their mutual orientation requirement is a property of the box-with-hole mesh template selected. For car crash analysis, the box needs to be first oriented along the face local crash direction vector of V_b as described in equation 9, section 5. Transformation of each box with respect to the face outer boundary is described in equation 11a and 11b.

The washer rings should ideally be oriented parallel to the tangent vector at the closest point on the box boundary. In order to achieve correct ring orientation near the box boundary so as to reduce stress computation errors, the paved ring (first layer of elements around the hole) needs to be appropriately rotated. This is achieved by clocking or rotating the geometry vertex of the circular face loop representing the hole by a parametric offset and by creating a virtual vertex at the new location. This virtual vertex is a ghost representation of the real geometry vertex. During meshing no node is created at the real vertex location but at the ghost vertex location. The node, however, is associated with the real vertex. Figure 18 explains the vertex rotation algorithm.

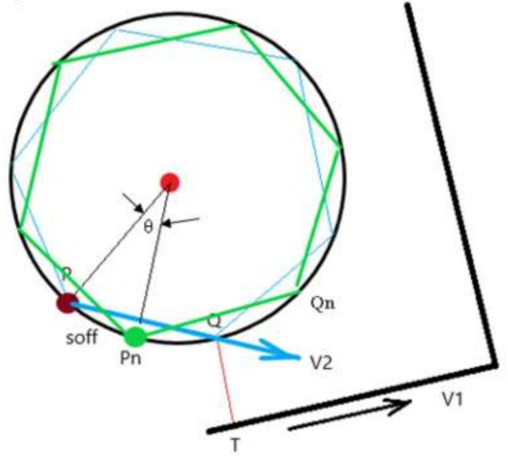


Fig. 18 Schematic explaining loop vertex rotation to make paved element edge parallel to boundary.

P denotes the vertex on a circular edge-loop running clockwise inside a box imprinted around the hole. The box is first oriented in the face local mesh flow direction V_1. This vector for a MOBB face is $VMOBB_x$ as explained in section 4.1 or V_b for the crossfield as explained in equation 9. In this particular case the user had applied a mesh control on this washer hole asking for 6 nodes. The blue hexagon inside the circle represents the default discretization on the hole with one node at its original vertex P. Element edge PQ is the edge closest to the box boundary with Q being the closest node. Q is projected to the nearest box boundary edge at point T. The tangent vector to the edge at T is V_1 which as explained before. The direction of the most box boundary proximate element edge PQ is represented by vector V_2. For the paved ring to be parallel to the box boundary, vectors V_1 and V_2 must be parallel. In other words,

$$\cos^{-1}\left(\frac{V_1.V_2}{|V_1|.|V_2|}\right) = \theta \qquad (12)$$

wher*e* ideally $\theta = 0$ deg. for the vectors to be parallel

Typically, they are not parallel. So, in order that the closest paved element edge is parallel to the nearest boundary edge, vertex **P** needs to be relocated such that $\mathbf{P_n}$ becomes its new location, and the green hexagon represents the new discretization on the hole. The green element edge $\mathbf{P_nQ_n}$, nearest to the boundary now ends at $\mathbf{P_n}$ instead of **P** and the edge becomes parallel to $\mathbf{V_1}$ and thus satisfies equation 12. To ensure this the radius vector at **P** needs to be clocked in a direction opposite to the circular edge-loop (counter - clockwise) by a parametric offset $\mathbf{s_{off}}$. This required vertex offset is expressed as

$$s_{off} = r\,.\cos^{-1}\left(\frac{V_1.V_2}{|\,V_1\,|.|\,V_2\,|}\right)/l_{loop} \qquad (13)$$

where r = radius of hole and l_{loop} = length or perimeter of the circular edge-loop

Once the boxes are oriented, the circular face loops (or holes) need to be clocked as explained in eqn. system 4, such that the paved first layer of quad elements around them are parallel to the box. This is illustrated in Figure 19.

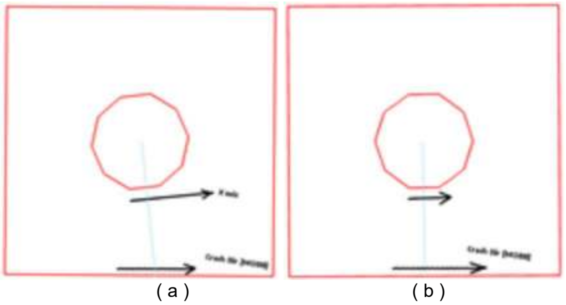


Fig. 19 Orienting the hole with respect to the box (a) to a parallel configuration (b) .

The decagonal discretization on the hole is initially unparallel to the box boundary (Figure 19a). After offsetting the circular edge's vertex, the orientation is corrected, and the hole is parallel to the box. The final mesh on the mesh will now be flowing in a unique direction both inside and outside the box as shown in Figure 1b. More examples of such meshes are shown in Figures 20 and 23.

7. MESHING ALGORITHMS FOR THE VIRTUAL FACES

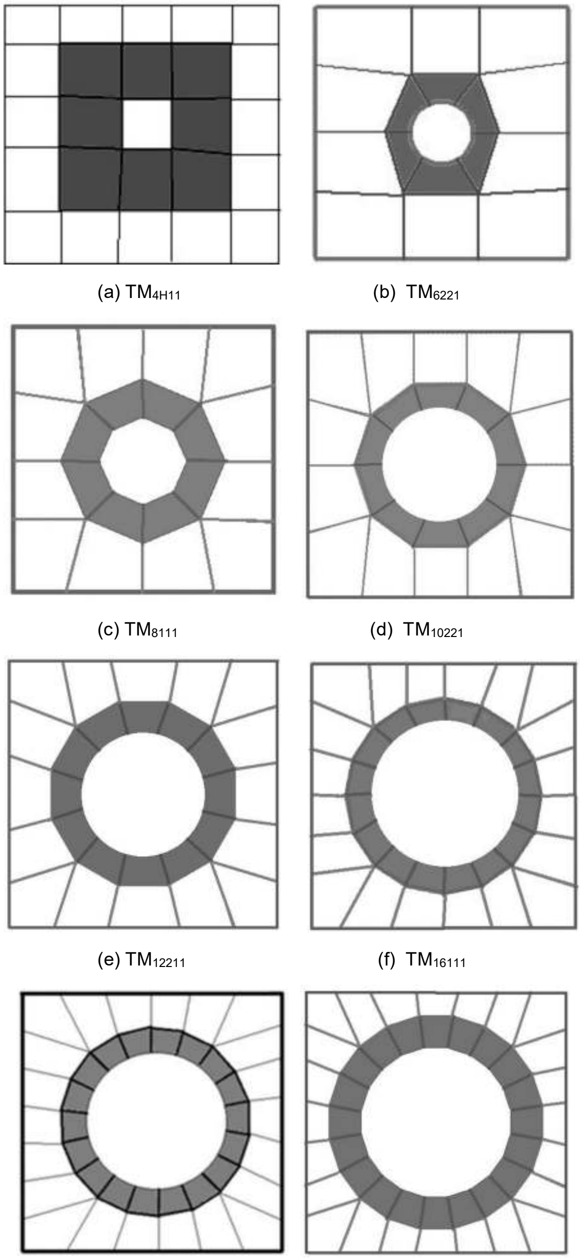
As described before, faces with washer holes are decomposed into two types of mesh areas (virtual faces) - the BWH virtual face and the BWH boolean virtual face. Delicately handled specialized mesh generators and meshing process algorithms have been developed to mesh these virtual faces.

7.1 Washer Mesh Control

Crash mesh finite element analysts typically require finer control on the washer regions. The washer holes represent bolt, lug, pin loading areas where the highest stresses occur making their locality critical sites of possible rupture during vehicle collision. According to Griffith's criteria of crack propagation the speed of crack propagation is directly proportional to the length of an initial crack. Naturally, a lot of care is taken to design and analyze washer sites. User-controlled patterned meshes are thus imperative. A washer mesh control provides that desirable user control. Washer mesh control is typically applied by hole radius range and is defined by three key parameters, namely number or count of washer elements (W_{ec}), number of washer element layers (W_{nl}) and thickness of washer layers (W_{lthk}). Usually, crash analysis meshes require a single layer, i.e. W_{nl} = 1 of even numbered elements.

7.2 Templatized Meshers for BWH Face

The second layer of elements in a box-with-hole virtual face is determined by the templatized meshers. The element count W$_{ec}$ on the inner loop prescribed by the user via washer mesh control defines the template meshers used.



(a) TM$_{4H11}$ (b) TM$_{6221}$

(c) TM$_{8111}$ (d) TM$_{10221}$

(e) TM$_{12211}$ (f) TM$_{16111}$

(g) TM$_{18121}$ (h) TM$_{20211}$

Fig. 20 Some meshing templates used for box-with-hole virtual faces from W$_{ec}$ =4, 20 where is W$_{ec}$ is even.

These templatized meshers (TM) are illustrated in Figure 20 and designated by

$$\mathbf{TM}_{NHPQ} \quad (14)$$

where N = element count (4- 20) on the hole; H = hole orientation (1-3 - perpendicular (1), parallel (2), other (3)); P = parity of element count type (1-3 divisible by 4(1), even (2), odd (3)) and Q = quad dominance (1-3 all quad (1), quad-dominant (2), triangular (3)).

The shaded regions in each templatized mesher represent the washer mesh-controlled areas where the user decides the three key parameters W$_{ec}$, W$_{nl}$, W$_{lthk}$.

7.3 Box Sizing

Imprintable boxes are placed at the center of washer holes. Box orientation algorithms have also been discussed. This sub-section deals with the determination of box size.

The generation of a quality mesh inside the box-with-hole imprinted virtual face encounters several challenges and conflicts. One of them is mesh quality defined by a number of quality metrices. The templatized mesh must meet these thresholds. Over and above this, a significant challenge is posed by the conflict of 3 sizes, namely –

i) The size **d$_{ec}$** defined by the user-driven element count on the washer hole; ii) The average size **d$_{b}$** defined by the element counts on the box boundary (determined by the template under consideration) and iii) The global mesh size **d$_{g}$.** The minimum and maximum permissible mesh sizes are denoted by $\boldsymbol{d_{mel}}$ and $\boldsymbol{d_{max}}$ respectively.

The following inequalities define the practical permissible ranges of sizes related to the imprintable box.

$$\boldsymbol{d_g \geq d_b \geq d_{ec}}$$

$$\boldsymbol{d_g \leq d_{max}}$$

$$\boldsymbol{d_{mel} \leq d_b \leq d_{max}}$$

$$\boldsymbol{d_{ec} > d_{mel}} \quad (15)$$

If the user-driven total thickness or offset of the mapped hole is designated by **d$_{lo}$**, **d$_{box}$** the side length of the box, **d$_{h}$** the diameter of the washer hole; the most ideal box dimension range (assuming square shape) can be expressed as

$$\boldsymbol{2k_{tol2}d_{max} > d_{box} > 1.25k_{tol1}d_{mel}} \quad (16)$$

where **k$_{tol1}$** and **k$_{tol2}$** are tolerance factors **k$_{tol1}$ = 1.1; k$_{tol2}$ = 0.909**

7.4 Mesher Selection Algorithm

As stated before, each face and as well their decomposed virtual faces are meshed by different meshers. Mesher choice algorithm is illustrated in Figure 21.

As faces are cycled, the map-meshable face is send to the transfinite mesher. A map-meshability check is run to ensure the face is worthy of a transfinite mesh. For faces with washer holes a MOBB is first generated. If the area ratio A$_{r}$ is equal or less than the area factor threshold, the orientation or box-turning V$_{ori}$ axis for all i holes becomes the x- (or u) component of the MOBB. Boxes are oriented accordingly and imprinted.

Templatized meshers are used for the box-with-hole virtual face while a multizone mesher [13] is used to mesh the box-with-hole boolean face. The multizone mesher is a hybrid quadrilateral mesher that combines paving, cartesian and subdivision algorithms in three distinctly different zones of the face.

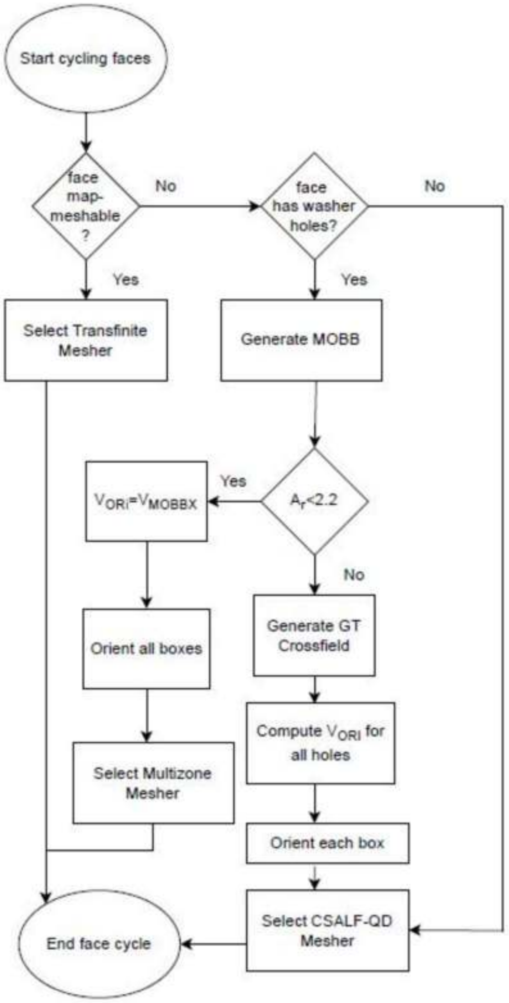


Figure 21. Mesher selection algorithm

If A_r exceeds the area factor threshold a GT crossfield is computed; V_{ori} axes for all i washer holes are uniquely computed. While box-with-hole virtual faces are meshed with templatized meshers as before the boolean face is meshed with the CSALF-QD (Combined Subdivision And Loop-Front - Quad-dominant version) mesher. The third category of faces are neither map-meshable nor have washer holes and are meshed with the CSALF-QD mesher.

7.4.1 Explaining Quad-dominance

Linear quad-dominant meshes have become the norm in automotive BIW crash analysis for several decades now. This choice needs to be explained as it is not obvious. Given the extremely large number of mesh-pattern, mesh flow and quality requirements, discussed in this paper, it is virtually impossible to generate an all-quad mesh that is able to honor all constraints, maintain a desirable mesh flow and meet all of the 12-16 quality goals discussed later in section 8. Less than 5 ~ 8% triangles become necessary to insert. However, care needs to be taken in the meshing algorithms to insert them such that one is able to minimize the following mesh irregularities - i) touching triangles ii) triangles on free geometry edges and critical feature lines and iii) triangles misaligned against the mesh flow direction. The meshing strategies/algorithms to achieve such orientations is complex and will be reported in a separate paper.

8. MESH QUALITY FOR CRASH ANALYSIS

Several monometric mesh quality measures [18,19] have been reported in the past which use a single preferably dimensionless, normalized metric to measure the overall quality of a quadrilateral (-dominant) shell mesh. All of these metrices have value in the sense they can be used to compare quad-dominant meshes reasonably well. However, crash analysis requires a far more thorough and stringent system of polymetric mesh quality evaluation. This industry does not rely on monometrices, but rather failure statistics. Table I depicts all relevant mesh quality parameters that are tracked for BIW meshes. The permissible thresholds are also reported as an industry average. The generic mesh quality goal is to limit failed elements in all categories to 0.005% or less. Of these a particular type of failure is not permissible at all - minimum element length (**d_{MEL}**). **E_g** denotes global element size

Table I: Crash Analysis Mesh Quality Measures

Symbol	**Quality Metric**	**Threshold**
d_{MEL}	Minimum Element Length	≥ 0.5 Eg
d_{MAXEL}	Maximum Element Length	≤ 1.5 Eg
J_R	Scaled Jacobian or Jacobian ratio	≥ 0.48
O_W	Warp	Solver dependent
σ_{SK}	Skew	Solver dependent
σ_{or}	Aspect Ratio	≤ 4.0
σ_T	Taper	Solver dependent
θ_{Qmax}	Maximum Quad Angle	$\leq 150\ deg.$
θ_{Qmin}	Minimum Quad Angle	$\geq 30\ deg$
θ_{Tmax}	Minimum Tria Angle	$\leq 140\ deg.$
θ_{Tmin}	Minimum Tria angle	$\geq 20\ deg.$
T_p	Percentage of Triangles	≤ 5.0

Crash analysis is a transient dynamic analysis where the critical time step is important in terms of achieving convergence. This critical time step depends on the speed of the longitudinal sound wave through the structural material. Equation 17 provides the relationship of the critical time step (Δt_c) and the minimum element length d_{MEL}

$$\Delta \boldsymbol{t_c} = \boldsymbol{d_{MEL}} = \frac{(\mathbf{1}+\alpha)\boldsymbol{A_e}}{\boldsymbol{c}} \qquad (17)$$

where area of the element is given by A_e, c denotes the characteristic length which for a quadrilateral is the longest diagonal and for a triangular element longest side length. Non-dimensional factor $\boldsymbol{\alpha} = 1$ for triangle and 0 for quadrilaterals. If the smallest element length in a panel mesh drops to less than $\boldsymbol{d_{MEL}}$, solution convergence becomes uncertain. Thus, a crash mesh becomes acceptable when all element quality failures listed in Table I are below the radar and no element fails $\boldsymbol{d_{MEL}}$.

A monometric measure, however, still becomes necessary to compare meshes. Accordingly, for a mesh of **N** elements of which $\mathbf{N_q}$ are quadrilaterals, a metric $\boldsymbol{\sigma_{cn}}$ called Mesh Condition Number is designed as

$$\boldsymbol{\sigma_{cn}} = \frac{w_1 + w_2 + w_3 + w_4}{\frac{w_1}{M_{ang}} + \frac{w_2}{M_s} + \frac{w_3}{M_{dist}} + \frac{w_4}{M_{qd}}} \qquad (18a)$$

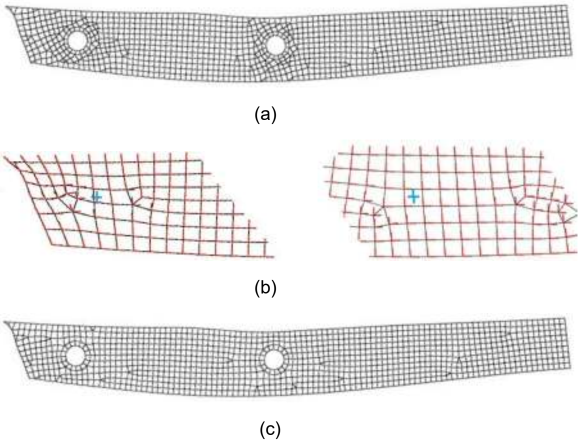
where angle metric

$\boldsymbol{M_{ang}} = \frac{N_{ni}}{N}$, N_{ni} being the number of elements whose included angles deviate from the ideal by > 10 degree

$\boldsymbol{M_{dist}}$ = harmonic mean of element scaled Jacobian

$\boldsymbol{M_s} = \frac{N_s}{N}$, where N_s = number of elements whose average size is > 90% of E_g.

$$\boldsymbol{M_{qd}} = \frac{N_q}{N} \qquad (18b)$$

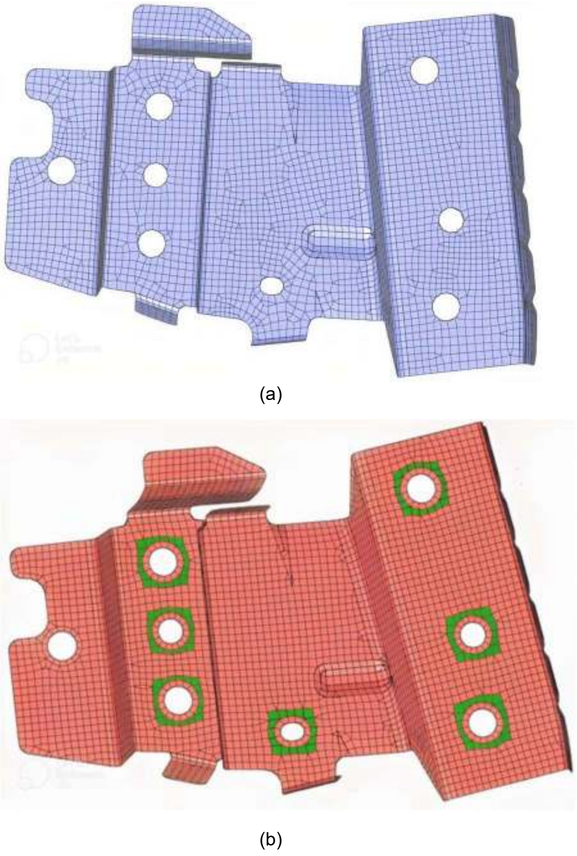

(a)

(b)

(c)

Figure 22. Comparison of meshes on a panel face with two washer holes with with/without box imprinting.

Figure 22 depicts the character of a mesh on a single face with washer-holes change from a grossly disoriented form (22a) to a flow-oriented mesh (22c). The ground truth frame field vectors in the localities of the hole centers (marked with blue crosses) are shown in Figure 22b. While all other quality measures pass in both, the monometric measure clearly separates the non-acceptable, disoriented mesh (where σ_{cn} = 0.895) from the oriented one (where σ_{cn} = 0.993). Figure 23 does a similar comparison on a smaller panel with/without washer imprinting. While both meshes pass all quality metrices listed in Table I, the mesh without washer imprinting has a mesh condition number of 0.879 while upon imprinting the norm is 0.955. A performance study is included in Appendix II.


(a)

(b)

Figure 23. Comparison of meshes without (a) and with (b) box imprinting.

9. CONCLUSION

This paper addresses a crucial industrial quadrilateral mesh generation problem for which no known solutions are available in open literature. For automotive carbody crash analyses it proposes a meshing strategy and related algorithms used to imprint box shapes around washer holes. Contrary to traditional methods of geometry modification which are permanent and can be damaging, a two-dimensional mesher-native, imprinting algorithm is developed. A box-with-a-washer layer shape is imprinted inside the 2D mesher. A minimum oriented bounding box and a ground truth crossfield based box orientation control function are developed. Virtual face decomposition is used to multiblock surfaces into box-with-hole and their boolean mesh areas. Detailed mesh quality comparison clearly justifies the strength of the present approach.

REFERENCES

[1] D.W. White and S. Saigal, “Improved imprint and merge for conformal meshing” *Proceedings of the 12th International Meshing Roundtable*, Ithaca, NY, pp.793-800 (2002).

[2] B.W. Clark, B.W. Hanks, C.D. Ernst, "Conformal Assembly Meshing with Tolerant Imprinting", *Proceedings of the 17th International Meshing Roundtable. Springer, Berlin, Heidelberg*, pp.267-281 (2008). https://doi.org/10.1007/978-3-540-87921-3_16

[3] T. D. Blacker, "The Cooper Tool", *Proceedings of the 5th International Meshing Roundtable*, pp.13-29 (1996).

[4] E. Ruiz-Girones, X. Roca, and J. Sarrate, "A new procedure to compute imprints in multi-sweeping algorithms", *Proceedings of the 18th International Meshing Roundtable*, pp. 281-299 (2009).

[5] Shengyong Cai and Timothy J. Tautges, "Surface Mesh Generation based on Imprinting of S-T Edge Patches", *23rd International Meshing Roundtable*, Procedia Engineering vol. 82, pp. 325 – 337 (2014).

[6] J.HC. Lu, I. Song, W.R. Quadros, "Geometric reasoning in sketch-based volumetric decomposition framework for hexahedral meshing". Engineering with Computers Vol. 30, pp.237–252 (2014). https://doi.org/10.1007/s00366-013-0332-z

[7] N. Mukherjee, "Imprint-based Mesh Generation for Computer Aided Design (CAD) Objects", Siemens Digital Industry Software patent application 2021P03555WO, (2021).

[8] J. E. Makem, H. J. Fogg, N. Mukherjee, "Automatic Feature Recognition Using the Medial Axis for Structured Meshing of Automotive Body Panels", Computer-Aided Design, 120, (2020). DOI: 10.1016/j.cad.2020.102845

[9] K. Beatty, N. Mukherjee, "Flattening 3D Triangulation for Quality Surface Mesh Generation". Proc. of the 17th International Meshing Roundtable. Springer, pp.125-139 (2008).

[10] K. Beatty, N. Mukherjee, "A Transfinite Meshing Approach for Body-In-White Analyses". Proc. 19th International Meshing Roundtable, Springer, pp.49-65 (2010).

[11] P. Kinney, “CleanUp: Improving Quadrilateral Finite Element Meshes”, Proc. 6th International Meshing Roundtable, pp.437-447, (1997).

[12] N. Mukherjee, “A hybrid, variational 3D smoother for orphaned shell meshes”, Proc. 11th International Meshing Roundtable, pp.379-390, (2002).

[13] N. Mukherjee, "Multizone Quadrilateral Mesh Generator for High Mesh Quality", Siemens Digital Industry Software Patent (pending) WO2020060561A1, (2018).

[14] N. Mukherjee, "CSALF-Q: A Bricolage Algorithm for Anisotropic Quad Mesh Generation’", Proc. XXth International Meshing Roundtable, Paris, France, Springer, pp. 489-510, (2011).

[15] A. M. Andrew, "Another Efficient Algorithm for Convex Hulls in Two.", Info. Proc. Letters 9, pp. 216-219, (1979).

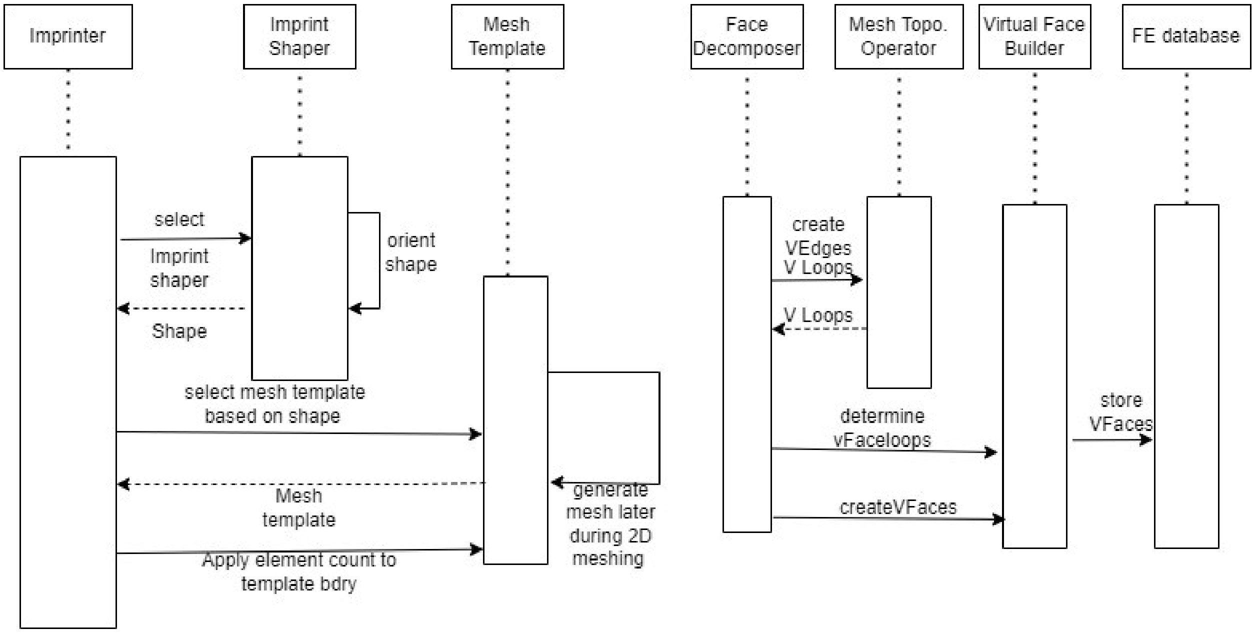
[16] G. T. Toussaint, "Solving geometric problems with the rotating calipers", Proc. MELECON '83, Athens (1983).

[17] A. Dielen, I. Lim, M. Lyon, L. Kobbelt, "Learning Direction Fields for Quad Mesh Generation, Eurographics Symposium on Geometry Processing", Ed. K. Crane and J. Digne, Vol 40(5) (2021).

[18] S. H. Lo, “Generating quadrilateral elements on plane and over curved surfaces”, Comput. Struct.,31, 421-426 (1989).

[19] S. A. Canann, J. R. Tristano, M. L. Staten, "An Approach to Combined Laplacian and Optimization-Based Smoothing for Triangular, Quadrilateral, and Quad-Dominant Meshes", Proc. 7th International Meshing Roundtable, pp. 309-323 (1998).

APPENDIX I



A UML sequence diagram of the proposed architecture for mesher-native shape imprinting

APPENDIX II

PERFORMANCE ANALYSIS

In terms of performance, the entire processing time is a natural function of the number of holes on the face the user wants a patterned mesh on. For a carbody panel with multiple faces, as shown in Figure 23, the total mesh generation (incudes parameter space generation, mesh data processing, size and direction map generation, meshing, topological cleaning, smoothing and mesh postprocessing) time is 12.7471 cpu secs. The mesher-native shape imprinting technique, as described by the entire content of this paper, takes 0.9325 cpu secs (i.e., 7.3154% of total meshing time). This indicates a typical performance rating. For an entire body-in-white crash analysis model shape imprinting time around washer holes is typically less than 8% of the total meshing time. Timing data is measured on a Windows x64, 16 processor desktop with the following configuration - Intel(R) Xeon(R) W-2245 CPU @ 3.90GHz.