Medial Axis Based Bead Feature Recognition for Automotive Body Panel Meshing

Jonathan E. Makem[1], Harold J. Fogg[1] and Nilanjan Mukherjee[2]

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

[1] Francis House,112 Hills Road, Cambridge, UK. CB2 1PH
[2] 2000 Eastman Dr., Milford, Ohio 45150 USA

Abstract. As a feature sensitive meshing investigation, this paper focuses on *beads* which are tangent continuous, high curvature, raised surfaces meant to stiffen and enhance the durability and specific strength of automotive body panels. An improvised and enhanced medial axis based strategy is proposed for identifying three broad types of bead features. Appropriate boundary discretisation, inclusion of zero medial vertex case for annulus identification, medial axis topology modifications to eliminate undesirable pathologies, T-junction squaring with cubic filtering smoothing highlight some of the improvisations to the medial axis technology employed. *Ridge curves* representing the crest lines of the bead are extracted and inserted on the face. A combination of multi-blocking, clamping and face node-loop insertion, followed by boundary connection strategies are used to generate high fidelity, feature sensitive, quasi-structured meshes.

Keywords: Medial Axis, Medial Object, Quad Meshing, Bead Feature, Feature Recognition

1 Introduction

Beads constitute the most important feature in contemporary automotive vehicle bodies. These are tangent continuous, medium to high curvature feature faces, usually single or two-looped, raised above the flat panel zones to provide structural stiffness, enhance durability and optimise panel weight, as shown in Fig. 1. Beads are manufactured by a deep-drawing process and their shapes and distribution are determined based on structural stress, NVH behaviour, crash response etc.

Beads are usually defined by many geometry parameters like curvature, form geometry, arrangement, height etc. It is imperative that the finite element mesh model capture these characteristics with high fidelity to ensure desired accuracy in the plethora of analyses performed on these panels. In the various analyses on the body panel different qualities and behaviours are targeted. For example, the structural engineer focuses on panel strength, the crash analyst on shock absorption, local buckling strength and crack propagation and the NVH engineer attempts to recover panel stiffness and lower levels

*Corresponding author.
E-mail address: jonathan.makem@siemens.com

of air and structure-borne noise and vibration leading to more efficient and lighter automotive body panels. Bead identification and modelling play a key role in meeting these goals.

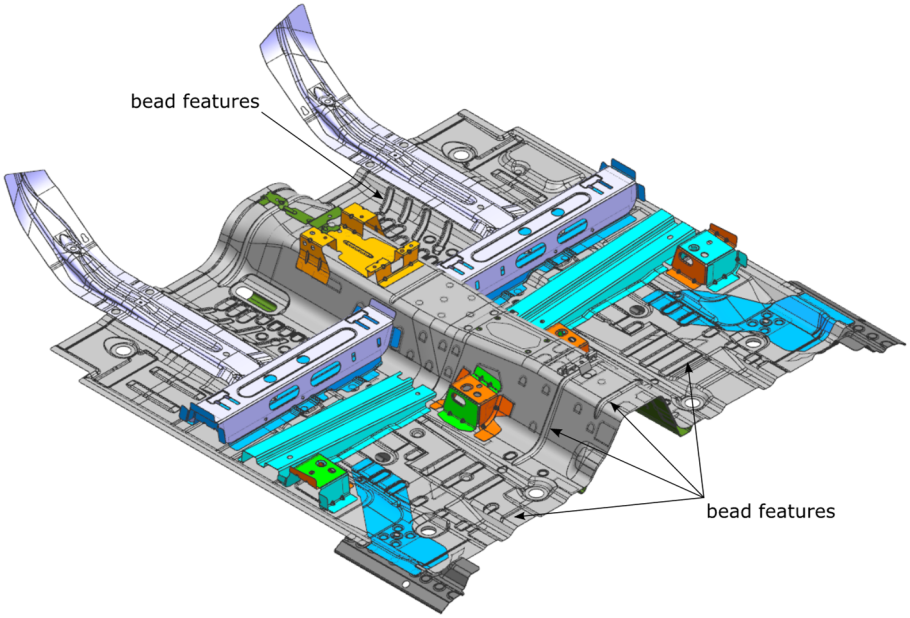


Fig. 1. Automotive Body Panel Assembly, courtesy of GM-Opel [1].

This paper describes a medial axis based strategy for bead identification and classification. An efficient and robust framework for computing the medial axis from a 2-D Delaunay tessellation is proposed. Simple strategies are incorporated for robustly optimising the medial axis topology and for improving its geometric shape. This enables its use for robust feature recognition. Based on the bead feature type and the ridge curve data generated to represent the crest line of the feature, the surface mesher employs feature specific methods to generate a desired mesh whose local parameters can be user-driven.

2 Previous Work

By far the most extensively used and practical form of medial axis computation is derived from a discretised point set of the boundary. These boundary-sampling based approaches generally approximate the medial axis from the Voronoi diagram or its dual, the Delaunay tessellation of the point set. Amenta *et al* [2] proposed a "Power Crust" method to extract the medial axis from the Voronoi vertices of a sample point set. An inverse transform is then applied to reconstruct the original shape. Dey *et al* [3] describe an algorithm which generates a sub-plot of the Voronoi diagram that approximates the medial axis from the Hausdorff distance. However, a major limitation with the approach

is that the sample points are only generated on smooth surfaces. Li *et al* [4] presented a "Q-MAT" technique which generates a piecewise linear approximation of the medial axis derived from an initial sample mesh using quadratic error minimisation for axis refinement. Its main drawback is its inability to preserve very sharp features on the boundary. Sun *et al* [5] propose a more efficient approximation error estimator that evaluates the 1-sided Hausdorff distance from the input shape to the boundary represented by the medial axis. The smoothness of the final medial mesh is questionable and the authors suggest further refinement by expensive re-meshing.

There are several alternatives to using the aforementioned geometric approximation error to govern the axis simplification process. Foskey *et al* [6] observe that angle-based filtration often causes significant changes in the topology of the medial axis. Chassard *et al* [7] devised the λ–medial axis which utilises the circumradius of the closest boundary point to a medial edge point. If the circumradius of a medial edge point is smaller than the λ tolerance, it is removed. Nevertheless, when the feature size changes dramatically, the approach is not consistent as small values of λ will not successfully prune medial points on larger features. Miklos *et al* [8] developed the Scaled Axis Transform (SAT) where all medial discs are scaled by a factor, $s > 1.0$. Any scaled disc which overlaps another is removed.

Aside from the above-mentioned boundary sampling techniques, a different method and indeed one of the earliest approaches [9] to computing a medial axis is referred to as "thinning". Given a voxel-based representation of the initial figure, the premise of this approach is to incrementally remove voxels from the boundary until enough voxels have been sufficiently removed to reveal the medial axis. In any case, an inherent limitation with thinning is that the final axis is normally off-center and generally exhibits a non-smooth, noisy definition [10]. A more advanced approach which yields a better solution is to use a distance transform [11,12]. However, these techniques have only been applied to relatively basic geometries.

Theoretically, it is possible to derive an exact definition of the medial axis for shapes defined by semi-algebraic sets, each set the solution to a finite system of algebraic equations and inequalities. Attali *et al* [13] observes however, that even for simple planar shapes bounded by primitive curves the algebraic complexities in the axis computation are severe. Aichholzer *et al* [14] implemented an approach that approximates the shape boundary to a series of bi-arcs. An algorithm is then applied to shapes which are bounded by circular arcs to generate the medial axis. But the approach was limited to single loop faces. Buchegger *et al* [15] improved this work by introducing a regularisation method aimed at simplifying the medial axis by smoothing the boundary of the domain and reducing the number of local curvature extrema while maintaining a low approximation error.

Peng *et al* [16] developed a locus method associated with the moving Frenet Frame [17] for generating the medial axis on B-rep models to simplify fillet features. A series of insertion points are established within close proximity of the fillet and are incrementally adjusted along a direction derived from the Frenet Frame formula. The algorithm finally converges when the point is equidistant from opposing boundary edges. The method is applied only to rectangular fillet faces.

3 Medial Axis Generation

Applications in the fields of motion planning [18,19], feature recognition [20], surface reconstruction [21,22] shape analysis [23] and (quasi-)structured meshing [24,25,26], to name a few, all utilise the medial axis to a high degree. Therefore, an extensively used tool such as this which is heavily relied on requires an efficient and reliable computation. Moreover, because the medial axis is inherently unstable, especially on geometries of industrial complexity, regularisation techniques must be employed to counteract this tendency.

3.1 Boundary Discretisation

An appropriately sized Delaunay mesh is first generated in 2-D. A reliable flattening algorithm [27] facilitates the creation of accurate medial axes on non-planar faces. The circumcentres of triangles of the Delaunay mesh are approximately located on the medial axis. For regions which are densely tessellated, circumcentres of adjacent triangles may be super-imposed which subsequently requires a complex system of collapse and update operations to produce a viable medial topology. This highlights the fundamental problem of adequate boundary discretisation. Dense point sampling on the boundary produces a more accurate approximation of the medial axis at the expense of more triangles which will also have a downstream impact on the computational efficiency of the topology march. Conversely, a sparse discretisation on the boundary will result in a coarse tessellation and a poorly defined medial object. Hence, the problem of producing an appropriate boundary sampling remains non-trivial as any improvement in accuracy by modifying the sample size will be negated by an increase in computational cost.

In industry, particularly in Computer Aided Engineering (CAE) applications, complex CAD geometries are often represented by facetted models which don't have a precise mathematical definition. Consequently, algebraic methods [14] for computing an adequate discretisation are not a feasible solution. Alternatively, as the underlying polygon faceting has already been sufficiently sized within acceptable bounds to respect the original parent CAD geometry, it is an acceptable compromise to use this information to determine an appropriate sample size for the Delaunay mesh. This avoids more expensive alternatives which seek to minimise geometric approximation errors by creating multiple medial objects [4,5].

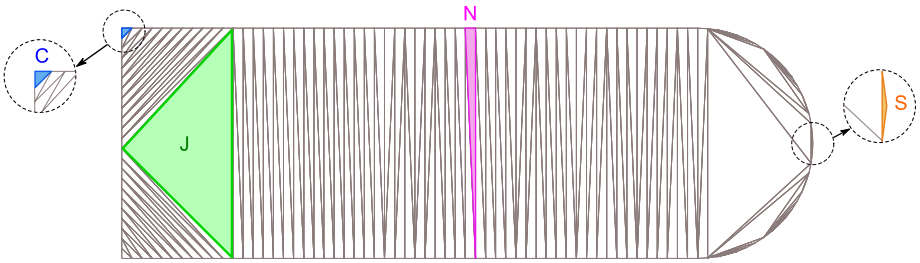


Fig. 2. 2-D Delaunay mesh with triangle types identified.

3.2 Delaunay Triangle Classification

Before the medial axis can be computed, the 3-D surface is flattened [27] and a constrained Delaunay mesh [28] is generated in parametric space. The triangles, t, of the Delaunay mesh are labelled according to the numbers of their adjacent triangles, $adj(t)$, as listed in Table 1.

$type(t)$	$adj(t).size()$	Other cond.
J	3	
N	2	
C	1	$\theta < \theta_{thresh}$
S	1	$\theta > \theta_{thresh}$

Tab. 1. Delaunay triangle types

The types C and S are distinguished from each other by the corner angle between their two free edges, θ. If it is less than a threshold, θ_{thresh} (e.g 160°), it is classified as C and the boundary is treated as a tangent discontinuity and a medial edge will connect to the vertex between the two free edges. If it is more than the threshold angle the boundary is treated as being tangent continuous.

3.3 Building the Medial Axis Topology

The medial axis topology is derived from the Delaunay triangles and their assigned types. Each medial vertex corresponds to a single triangle of type J, C or S. Each medial edge corresponds to sequences of triangles beginning and ending at J, C and S triangles and running along adjacent N triangles. Pseudo code of the algorithms for establishing the medial vertices and medial edge topologies are given in Algorithms 1 and 2. (C++ STL Containers Library terminology is used [29].)

Algorithm 1. – Establish Medial Vertices

Output: List of Medial Vertex triangles, mediaVertexTs.

```
medialVertexTs = [ ]  // a sequence container
for t in Delaunay Mesh:
   if type(t) != N:
      medialVertexTs.push_back(t)
```

A special case is where there are only N and no J triangles and therefore there are no medial vertices. The only possibility is that the face is an annulus and the medial axis forms a single closed loop.

Algorithm 2. – Establish Medial Edges

Output: List of medial edge triangle sequences, medialEdgeTs.

```
medialEdgeTs = [ ]
for mvT in medialVertexTs:
   mark(mvT)
   for adjT0 in adj(mvT):
      if hasMark(adjT0):
         continue
   meTs = [ ]
   meTs.push_back(mvT)
   mark(meTs.back())
   while type(meTs.back() == N:
      for adjT1 in adj(meTs.back()):
         if (adjT1 != *(meTs.end() -2):
            meTs.push_back(adjT1)
            mark(meTs.back())
   medialEdgeTs.push_back(meTs)
```

Due to the empty circumcircle property of the constrained Delaunay mesh, the circumcentres of the triangles tend to the medial axis as the discretisation of the boundary goes to zero. Thus, the circumcentres of the triangles are used as approximate positions for the medial axis.

3.4 Medial Edge Classification

By classifying medial edges based on the types of the associated Delaunay triangles of the end medial vertices, there are six possible types: S-S, S-C, S-J, C-C, C-J and J-J. Closed medial edges without medial vertices make a seventh type called N. These are illustrated in Fig 3.

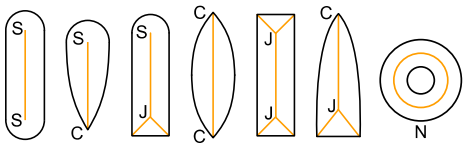


Fig. 3. Medial edge types.

3.5 Medial Axis Topology Processing

The initially generated medial axis may have unwanted artefacts as consequences of the finite discretisations of the boundaries and the subtle particulars of the Delaunay mesh. Typically, these are medial edges with lengths below the discretisation size of the boundaries and they most commonly occur in finite contact zones. A topology clean up process as outlined in Alg. 3 is used to modify the topology to remove these medial edges. Examples are shown in Fig. 4.

Algorithm 3. – Topology Clean Up

```
mvNearbyData = [ ]  /*  sequence container of tuples:
                        (medial vertices, distance, type) */
for mv in medialVertices:
   nearbyMvs, culumlativeDis = findNearbyMedialVertices(mv,
                                                        tolDis)
   mvNearbyData.push_back((nearbyMvs, culumlativeDis,
                           type(mv)))

sortedIndices = (0, 1, ... medialVertices.size() - 1)
sort(sortedIndices.begin(), sortedIndices.end(),
        lambda[mvNearbyData](i0, i1):
           return lexiographical_greater(
              mvNearbyData[i0], mvNearbyData[i1])

for i in sortedIndices: // visit all medial vertices in sorted order
   mv0 = medialVertices[i]
   if hasMark(mv0)
      continue
   mark(mv0)
   nearbyMvs0 = mvNearbyMvs[i][0]
   for mv1 in nearbyMvs0:
      mark(mv1)
      replace(mv1, mv0)
```

Notes

findNearbyMedialVertices returns:
- the nearby medial vertices within inputted traversal distance tolerance and
- the cumulative traversal distances from the medial vertex to its nearby medial vertices

lexiographical_greater performs a sequence of greater_than comparisons of the elements of the tuples:

```
if mvNearbyData[i0][0].size() != mvNear-
byData[i1][0].size():
   return mvNearbyData[i0][0].size() > mvNear-
byData[i1][0].size())
else if mvNearbyData[i0][1] !=  mvNear-
byData[i1][0]:
   return mvNearbyData[i0][1] >  mvNear-
byData[i1][1]
else if mvNearbyData[i0][2] == J xor mvNear-
byData[i0][2] == J
   return mvNearbyData[i0][2] == J
else
   return false
```

replace(mv1, mv0) replaces all instances of mv1 with mv0 in the medial axis data model

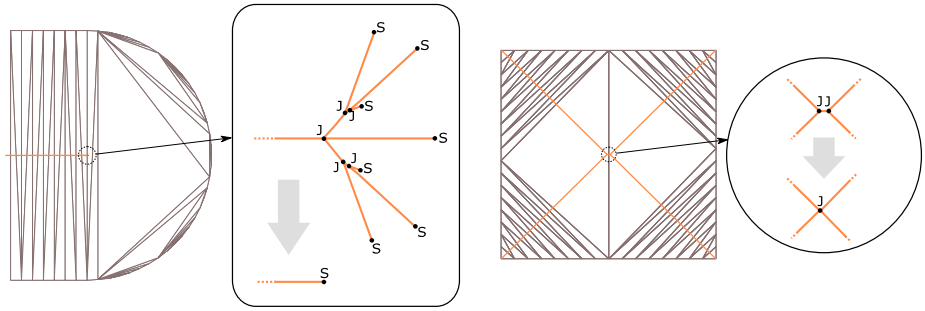


Fig. 4. Examples of medial axis topology clean up.

3.6 Medial Axis Geometry Processing

The initial geometric representations of medial edges are piecewise-linear curves through the Delaunay triangle circumcentres. To improve their smoothness two iterations of cubic smoothing filtering are applied [30]. This involves optimising the position of every vertex, v_i, in the piecewise-linear curve in turn except for the end vertices. A cubic polynomial is fitted to the surrounding vertices, $\boldsymbol{v}_{surr} = (v_{i-2}, v_{i-1}, v_{i+1}, v_{i+2})$, in a local coordinate system and then the position of v is adjusted to lie on the curve, as illustrated in Fig.5a. The local coordinate system is chosen to have its origin at the average position in $\boldsymbol{v}_{surr}$ and its x-axis aligned with the line through $\boldsymbol{v}_{surr}[0]$ and $\boldsymbol{v}_{surr}[-1]$ (i.e. the first and last elements in $\boldsymbol{v}_{surr}$). If v_i is adjacent to an end vertex then v_{i-2} or v_{i+2} is left out of $\boldsymbol{v}_{surr}$ and a least norm solution is found. An example showing the smoothed result is given in Fig.5b.

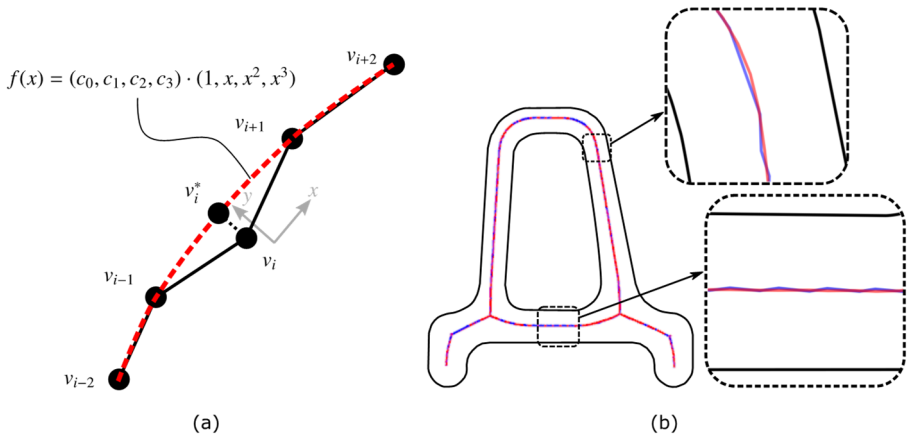
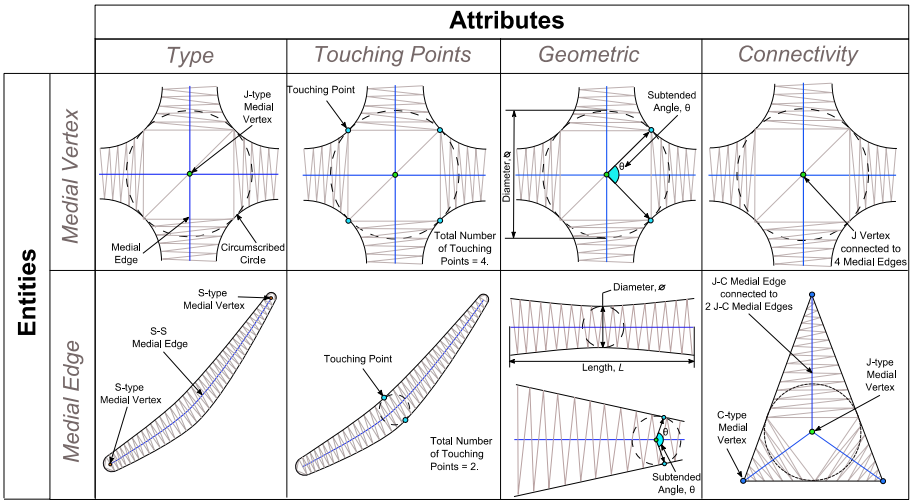


Fig. 5. (a) Cubic smoothing filtering of the vertex v_i by fitting a cubic polynomial to the surrounding vertices v_{i-2}, v_{i-1}, v_{i+1}, v_{i+2}. (b) An example showing the original (in blue) and optimised (in red) polyline medial edges after 2 iterations of cubic smoothing filtering.

3.7 Medial Object Data Model

The medial object is the data model for organising the medial axis points. In 2-D it involves medial edges that join at medial vertices. Additionally, thickness, subtended angle, touching point and length information are associated with the medial object entities, as shown in Table 2.



Tab 2. Medial Object Data Model.

3.8 Examples

The developed medial object generation strategy is generic and can be used for any face, not just beads or feature faces. Examples of computed Medial Axes on complex geometries are shown in Figs 6, 7, 8 and 9.

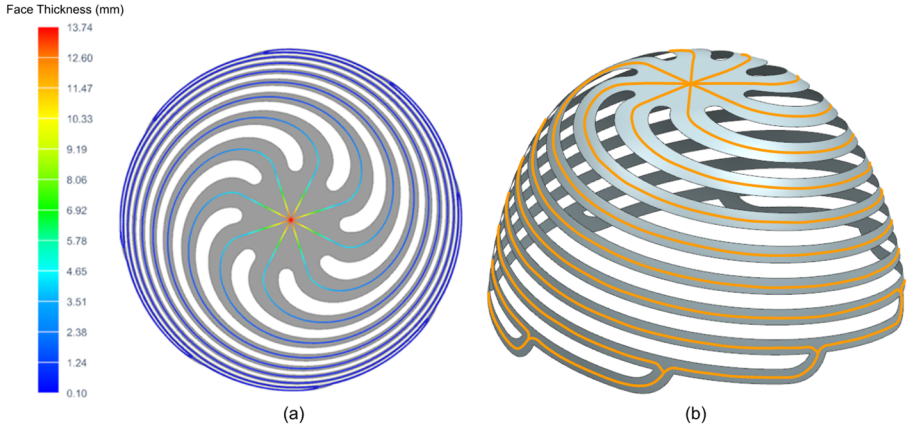


Fig. 6. A spiral bowl with its medial axis (a) in flattened 2-D space with face thickness contour plot and (b) transformed back to 3-D space.

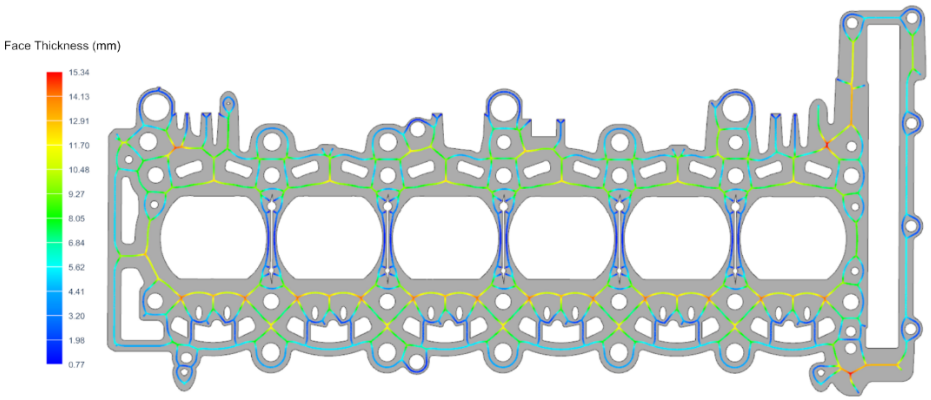


Fig. 7. Medial axis of an engine cylinder head gasket with a thickness contour plot

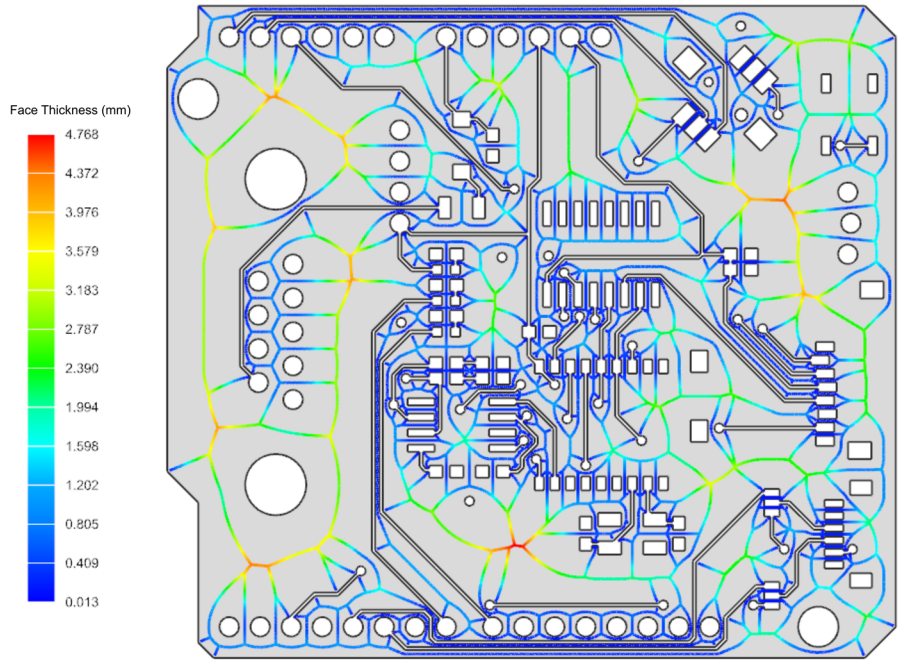


Fig. 8. Medial axis of a printed circuit board with a thickness contour plot.

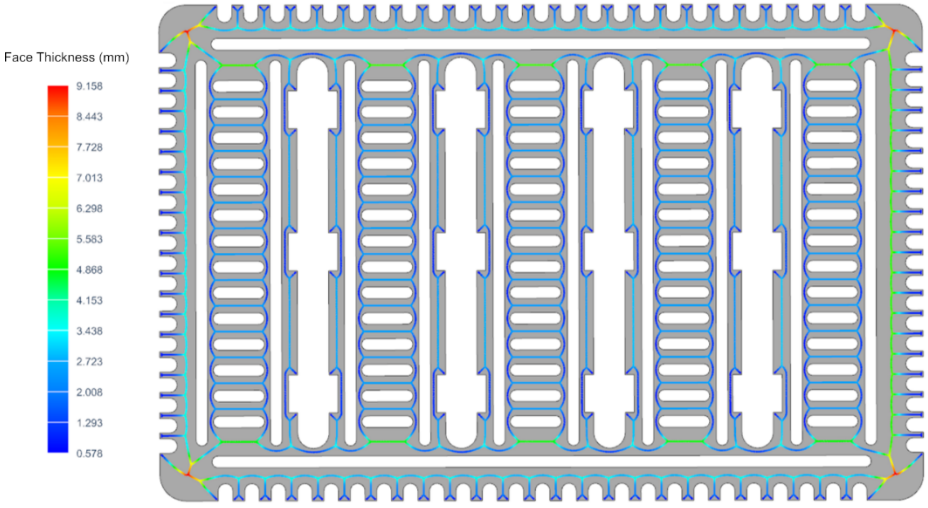


Fig. 9. Medial axis of a complex automotive component with a thickness contour plot.

4 Bead Feature Identification and Processing

This section describes how the aforementioned medial axis technology is used to identify and process a range of automotive body panel bead features, and thereby enables the creation of feature-specific quasi-structured meshes.

4.1 Bead Types

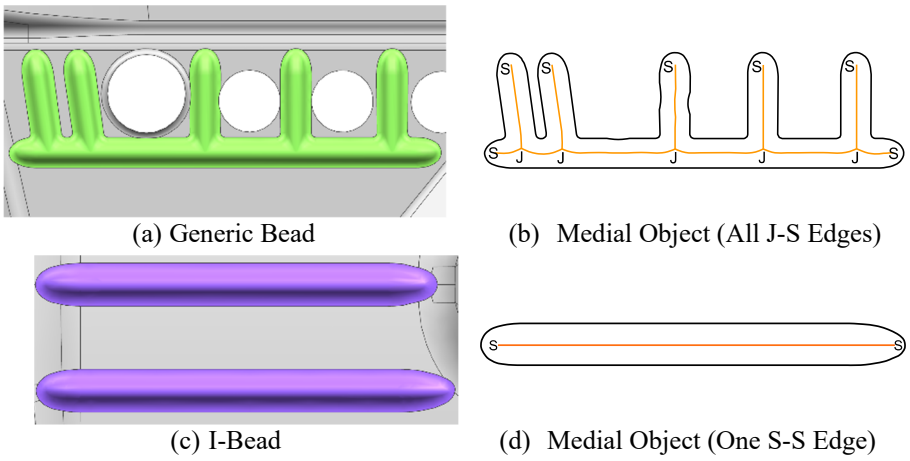
In general, a bead feature is defined as a high or medium curvature (single or two- loop) face which is long and slender with a tangent continuous boundary. Consequently, the medial object can be used to identify such features using the type, thickness and length attributes of the edge and vertex entities.

A face is deemed to be a bead feature if:

- the boundary is G^1 smooth,
- the face has a single or two loops and
- the average aspect ratio of the medial edges (computed by averaging the length to thickness ratio of its medial edges) is greater than a predefined threshold.

Beads can be of many shapes and based on the mesh preferred they can be further sub-categorised into two broad variants:

- Generic-beads which have more than 1 medial edge (Figs. 10a & 10b) and
- I-beads which have only 1 medial S-S edge (Figs. 10c & 10d)



(a) Generic Bead (b) Medial Object (All J-S Edges)

(c) I-Bead (d) Medial Object (One S-S Edge)

Fig. 10. Bead features.

The Generic-bead can be further sub-classified into X, Y, J, G, L etc. shapes as templatised meshes are required on them. Once these features have been identified a crucial metric to consider while preparing them for meshing is the Minimum Element Length (*MEL*). The *MEL* is critical for crash analyses solution stability and efficiency. It is the lowest element size below which no element edge length must fall.

4.2 I-Beads

To facilitate the generation of a structured mesh the I-bead is partitioned or multi-blocked into three regions. A central rectangular portion and a tip at each end. Two pairs of medial axis touching points are used to define virtual vertices and virtual edges that establishes the multi-block topology, as shown in Fig. 11. These touching points are located at positions along the S-S medial edge where

- the face thickness (diameter of the inscribed circle) is greater than $2 \times MEL$ and
- $r + d > 4 \times MEL$, where r is the radius of the inscribed circle at the end S-type medial vertex and d is distance along the medial axis from the end medial vertex.

Once the virtual edges have been established the multi-block topology is formed using an approach by Mukherjee and Makem [31].

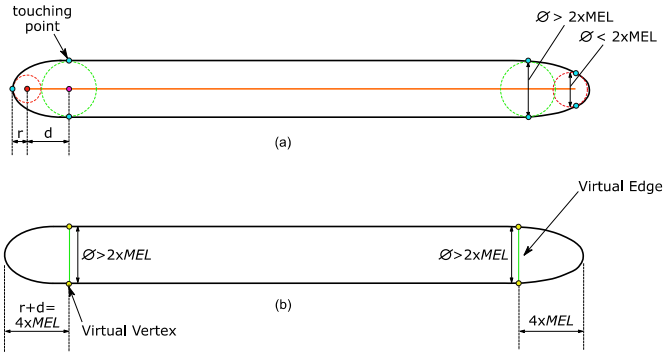


Fig. 11. Multi-blocking of I-bead feature (a) determining the split lines (b) forming the block topology.

4.3 Generic Beads

Generic beads are processed for meshing in manner different than for I-beads. Instead of multi-blocking the face, the medial axis is used as a ridge curve which the mesh is forced to respect. Typically, near medial vertices, medial edges possess a degree of curvature which is not conducive for ridge curve use. Thus, a T-junction squaring process is performed as illustrated in Fig. 12.

Medial vertices with three medial edges are candidates for T-junction squaring. First, positions on the medial edges at a distance of the inscribed circle radius from the medial vertex are approximated. If the medial edge ends before that distance the position of the medial vertex on the other end is used. Next, local tangents of the medial edges are evaluated using a modified version of the cubic filtering smoothing method. Two conditions are required to carry out the squaring. These are (referring to Fig. 12):

1. $\angle(\boldsymbol{t}_i, \boldsymbol{t}_j) > \theta_{thresh}$ where $i, j \in \{1,2,3\}, i \neq j$ … (165° is used for θ_{thresh}),
2. $\angle\big((\boldsymbol{p}_i - \boldsymbol{p}_k), \boldsymbol{t}_k\big) < \angle\Big((\boldsymbol{p}_i - \boldsymbol{p}_k), \big(\boldsymbol{p}_j - \boldsymbol{p}_k\big)\Big)$

The medial vertex location $\boldsymbol{p}_{mv}$ is moved to $\boldsymbol{p}^*_{mv}$, the intersection of the line through $\boldsymbol{p}_k$ in the direction of $\boldsymbol{t}_k$ and the line through $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$. The medial edges are trimmed from their respective points and extended to the new medial vertex.

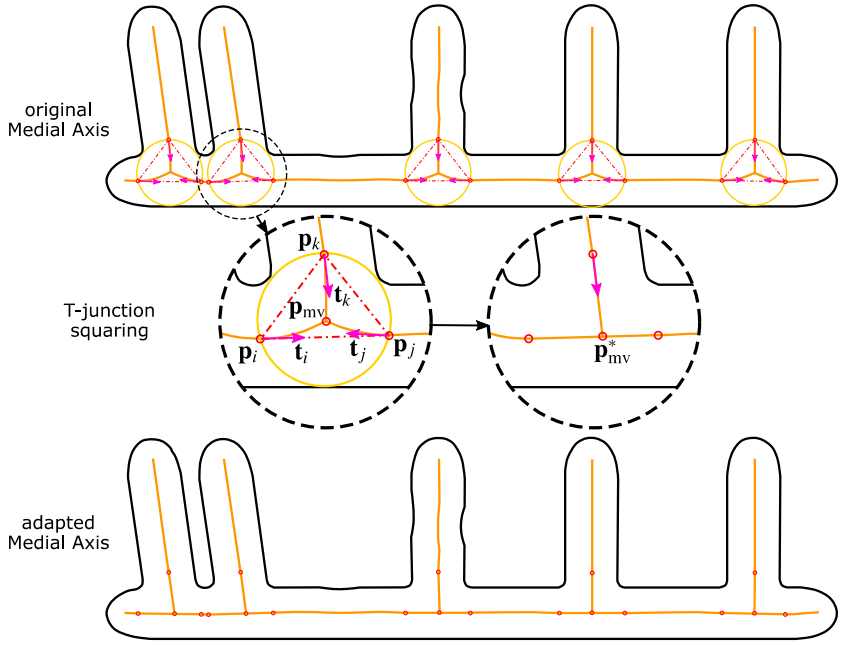


Fig. 12. T-junction squaring of the medial axis on a generic bead.

4.4 Annular Beads

Annular beads are another common type of feature specific to body panels where a structured mesh must be applied. Unlike the I and Generic beads, these beads are two-loop faces with G^1 boundaries and one loop is a distance offset of the other. The medial axis can identify any face as an annulus if it has a single *N*-type medial edge, as shown in Fig. 13.

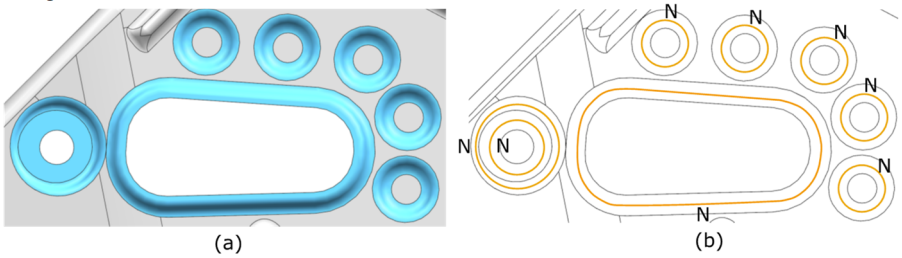


Fig. 13. A typical body panel displaying (a) annular beads with (b) associated medial objects (single N-type medial edges per face).

After the annuli have been identified the touching points of the inscribed circle of the medial axis are used to define virtual vertices at a parametric position of $t = 0.0, 0.25, 0.5$ and 0.75 along the medial edge. In a similar fashion to the I-bead, these vertices are

linked to form virtual edges which are used to decompose the face into a series of 4-sided, map-meshable blocks [31], as shown in Fig. 14.

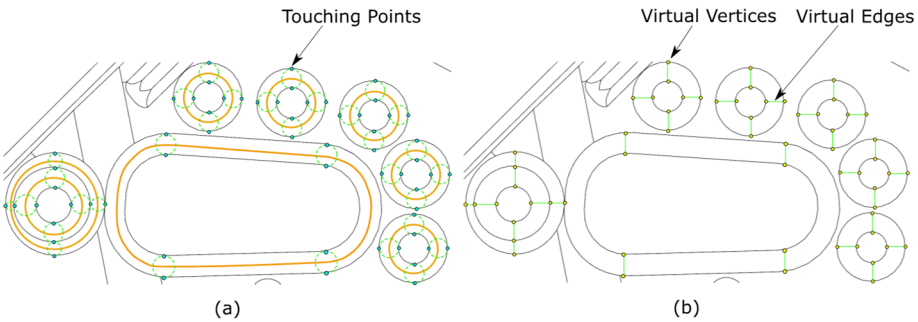


Fig. 14. Multi-blocking of annular beads (a) determining the split lines (b) forming the block topology.

5 Meshing

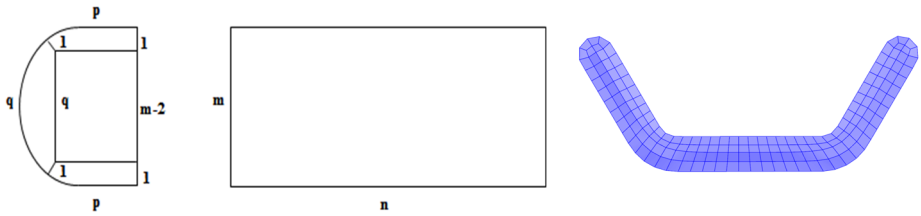
Being special features, the beads, along with other features are meshed before other faces of the body panel in order to allow for lesser constraints and higher degrees of freedom. The meshing methods employed for beads are a function of their type.

5.1 I-Beads

The I-Bead is multi-blocked using a procedure developed recently [32] into a rectangular mid-section and two semi-oval end caps as shown in Fig. 11. The mid-section is mapped meshed with an even number of elements along the thickness, so as to capture the crest of the feature. The end caps are meshed using a templatised clamping procedure as described in Fig. 15a. The sub-face is decomposed into 4 sub-areas, each of which is map-meshed. The element count scheme used is described by letters m,n,p and q obeying the following relations

$$m \,\%\, 2 = 0,$$
$$q = m - 2.$$

The final mesh is the result of variational [33] and optimisation smoothing [34] of the mesh assembly from the three sub-sections, as shown in Fig. 15(b).

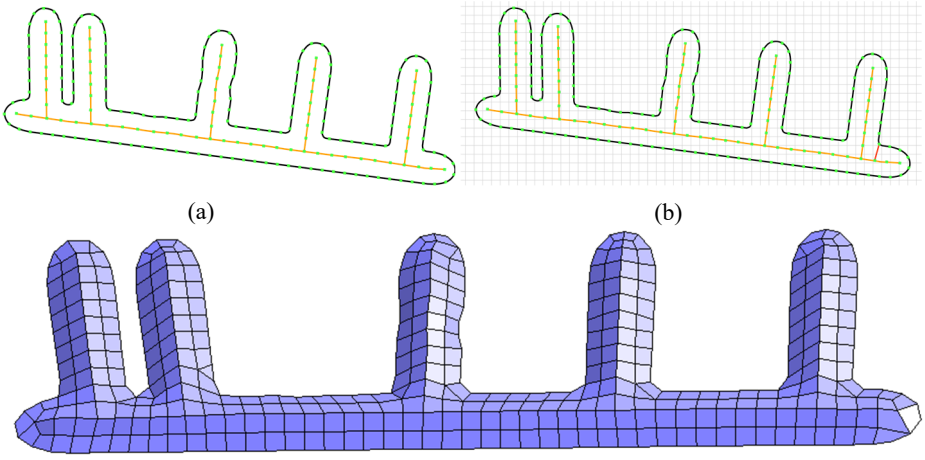


(a) (b)

Fig. 15. Meshing I-beads showing element count distribution (a) and the final mesh (b).

5.2 Generic Beads

For the generic bead, ridge curves computed from the medial object data are handed on to the mesher. The mesher constructs a scar-loop (self-retracing) from the discrete 3D point data and inserts it into the face as an artificial face-loop (Fig.16a). The mesher modifies the face topology with this injected, retraceable inner loop.



(a) (b)

(c)

Fig. 16. Meshing a generic comb-shaped bead with ridge curves, where (a) the ridge curve is face-inserted, (b) discretised face node loops are joined and meshed (c).

The face loops are discretised and the inner loop is connected (Fig. 16b) with the outer at locations (in red) based on proximity determined from the underlying voxel model the 2d flattened geometry is overlaid on. The face area, therefore, is reduced to a single self-touching node loop which is meshed using a subdivision meshing procedure [27]. This approach ensures that the generic bead feature line is embedded on the face as a meshing constraint and is accordingly honoured by the mesh (Fig. 16c). The mesh on the comb-shaped bead (Fig. 10a) is unstructured, but the crest line (ridge curve) of the bead feature is honoured by the mesh. Further examples are shown in Fig. 17.

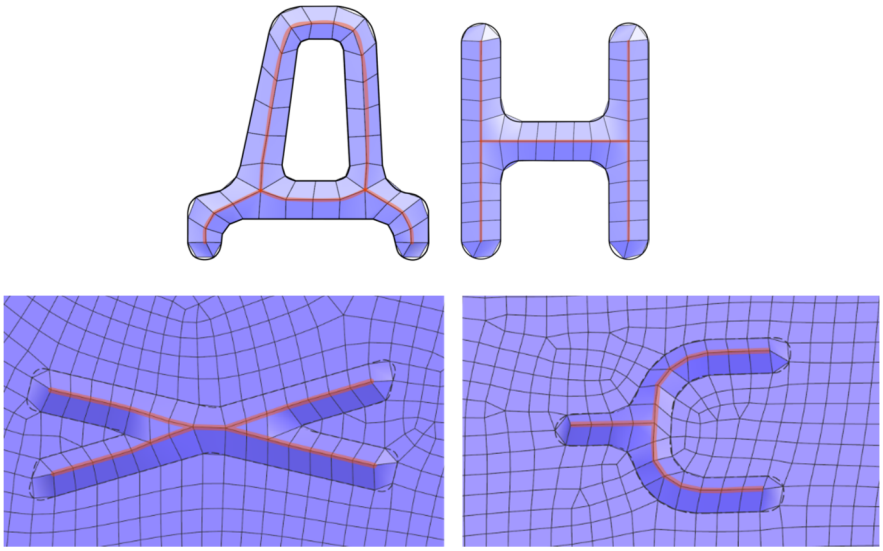


Fig. 17. Example meshes on generic beads using ridge curves.

5.3 Annular Beads

Perfectly structured meshes are desired on annular beads. To ensure this, annular bead faces are multi-blocked into 4-6 segments and each annular segment or virtual face is then mapped meshed. Fig. 18 shows such a cluster of concentric annular beads (from the examples in Fig.13 & 14) split into 4-6 sub-faces (mesher-native virtual edges representing the "splits" are indicated by blue lines) and mapped meshed with usually an even number of elements around the circumference.

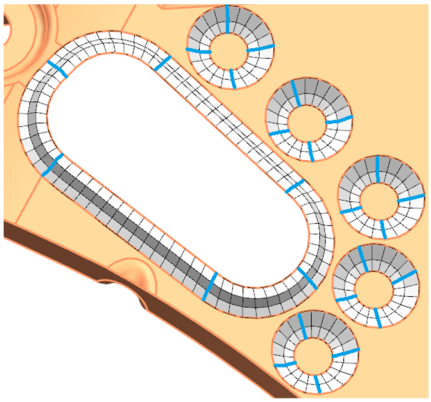


Fig. 18. Examples of meshes on annular beads.

6 Conclusions

This paper presents an assembly of unique strategies for identifying, processing and subsequently meshing bead features in automotive body panels. State-of-the-art medial axis technology is improvised for appropriate boundary discretisation, annulus identification by zero medial vertex instances and topology modifications to eliminate unwanted artefacts. Medial edge geometry is improved with cubic smoothing and T-junction squaring employing careful vertex repositioning. Three broad types of bead features, namely I-shaped, generic and annular beads are identified and processed. While multi-blocking strategies are used for meshing the I-shaped and annular beads, a ridge curve, representing the feature centre or crest line, is extracted for generic beads. Feature-specific surface meshing methods involving clamping templates and face node loop insertion followed by boundary connection are used to mesh the beads. The results show high quality structured meshes honouring crest lines.

References

1. http://www.opel.com . (2018). Opel international - Product & Company Information, News, Experience, Excitement. [online] Available at: https://www.opel.com/ [Accessed 13 Jun. 2018].
2. Amenta, N., Choi, S., Kolluri, R.K. The power crust. In Proc. ACM Solid Modeling. 2001. 249-260.
3. Dey, T.K., Zhao, W. Approximating the medial axis from the Voronoi diagram with a convergence guarantee. Algorithmica 2004. 38:179–200.
4. Li, P., Wang, B., Sun, F., Guo, X, Zhang, C., Wenping, W. Q-Mat: Computing Medial Axis Transform by Quadratic Error Minimization. ACM Transactions on Graphics. 2015. 35. 1-16.
5. Sun, F., Choi, Y.K., Yu.,Y., Wang, W. Medial meshes – a compact and accurate representation of medial axis transform. IEEE Transactions on Visualization and Computer Graphics.2016. 22. 1278-1290.
6. Foskey, M., Lin, M.C., Manocha, D. Efficient computation of a simplified medial axis. Journal of computing and information science in Engineering. 2003. 4. 274-284.
7. Chaussard, J., Couprie, M., Talbot, H. A discrete λ-medial axis. Proc. 15th IAPR Int. Conf. Discrete Geometry Comput. Imagery. 2009. 421 – 433.
8. B. Miklos, J. Giesen, and M. Pauly, “Discrete scale axis representations for 3D Geometry”. ACM Trans Graph. 2010. 29. 1 – 10.
9. Tagliasacchi, A., Delame, T., Spagnuolo, M., Amenta, N., Telea, A. 3D Skeletons: A State-of-the-Art Report. Eurographics 2016. 35. 1-24.
10. Saha, P.K., Borgefors, G., Di Baja, G.S.: A survey on skeletilization algorithms and their applications. Pattern Recognition Letters. 76. 2016. 3-12.
11. Arcelli, C., Sanniti G., Serino, L. Distance driven skeletalization in voxel images. IEEE TPAMI 33. 2011. 4. 709-720.
12. Xia, H., Tucker, P.G. Fast equal and biased distance fields for medial axis transform with meshing in mind. Journal of applied mathematical modelling. 2011. 35. 5804-5819.
13. Attali, D., Boissonmat, J.D., Edelsbrunner, H. Stability and computation of medial axes – a state-of-the-art-review. Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration. 2009. 109-125.

14. Aichholzer, O., Aigner, W., Aurenhammer, F., Hackl, T., Juttler, B., Rabl, M. Medial axis computation for planar free-form shapes. Journal of Computer Aided Design. 2009.
15. Buchegger, F., Juttler, B., Kapl, M. Total curvature variation fairing for medial axis regularization. Graphical models. 2014. 76. 633-647.
16. Peng, J., Wang, H., Li, J., Song, C. Generation method and application of product-oriented medial axis. 6th International Conference on Logistic, Informatics and Service Science. 2016. 16. 160-174.
17. Cao, L., Liu, L. Computation of the medial axis and offset curves of curved boundaries in the planar domain. Computer Aided Design. 2008. 40. 465-475.
18. Ding, D., Pan, Z., Cuiuri, D., Li, H., Larkin, N. Adaptive path planning for wire-feed additive manufacturing using medial axis transformation. Journal of Cleaner Production. 2016. 133. 942-952.
19. Van Toll, W, Cook, A., Van Kreveld, M.J., Geraerts, R. The explicit corridor map: using the medial axis for real-time path planning and crowd simulation. Proceedings of the 32nd International Symposium on Computational Geometry. 2016. 51. 72-75.
20. Durix, B., Morin, G., Chambon, S. Skeleton-based multiview reconstruction. IEEE International Conference on Image Processing. 2016. 4947-4051.
21. Yuan, J., Cheriyadat, A.M. Image feature based GPS trace filtering for road network generation and road segmentation. Machine and Vision Applications. 2016. 27. 1-12.
22. Tagliasacchi, A., Delame, T., Spagnuolo, M., Amenta, N., Telea, A. 3D Skeletons: A state-of-the-art report. Eurographics. 2016. 35. 1-26
23. Yasseen, Z., Verroust-Blonder, A., Nasri, A. Shape matching by part alignment using extended chordal axis transform. Pattern Recognition. 2016. 57. 115-135.
24. Tam, T, Armstrong, C. 2D finite element mesh generation by medial axis subdivision. Advances in Engineering Software. 13. 1991. 313-324.
25. Fogg, H.J., Armstrong, C.G., Robinson, T.T. Enhanced medial-axis-based block-structured meshing in 2D. Journal of Computer Aided Design. 2016. 72. 87-101.
26. Makem, J.E., Armstrong C.G., Robinson, T.T. Automatic decomposition and efficient semi-structured meshing of complex solids. Engineering with Computers. 30. 2014. 345-361. https://doi.org/10.1007/s00366-012-0302-x.
27. Beatty, K. and Mukherjee, N., Flattening 3D Triangulations for Quality Surface Mesh Generation, Proc. 17th Int. Meshing Roundtable, 2008, pp.125-139.
28. Shewchuck, J.R. Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator. Applied computational geometry towards geometric engineering. 1996. 203-222.
29. C++ STL Containers Library, [online] Available at: https://en.cppreference.com/w/cpp/container
30. Bonneau, G-P, Hahmann, S. Smooth Polylines on Polygon Meshes. Geometric Modeling for Scientific Visualization, 2004, pp 69-84.
31. Mukherjee, N., Makem, J.E., A Cartesian Slab Based Multiblocking Strategy for Irregular Cylindrical Surfaces. Proc. 26th Int. Meshing Roundtable, 2017.
32. Makem, J.E. and Mukherjee, N., Mesh generation system and method, Patent Application WO2017040006A1, Siemens PLM Software Inc., 2015-09-01. URL: https://patents.google.com/patent/WO2017040006A1/en.
33. Mukherjee, N., A hybrid, Variational 3D smoother for orphaned shell meshes, Proc. 11th Int. Meshing Roundtable, 2002, pp. 379-390.
34. Mukherjee, N., Makem, J.E., Fogg, H.J. A 3D Constrained Optimisation Smoother to Post-process Quadrilateral Meshes for Body-in-white. Proc. 25th Int. Meshing Roundtable, 2016, pp. 262-275.