Available online at www.sciencedirect.com

ScienceDirect

Procedia Engineering 00 (2014) 000–000

Procedia Engineering

www.elsevier.com/locate/procedia

23rd International Meshing Roundtable (IMR23)

An Art Gallery approach to submap meshing

Nilanjan Mukherjee

Meshing & Abstraction
Digital Factory
Siemens PLM Software
SIEMENS
2000 Eastman Dr., Milford, Ohio 45150 USA

Abstract

The Art Gallery Theorem and its lemmas have been used since the 1990s in the area of computer graphics to decompose orthogonal polygons into convex sub-regions. This paper extends and improvises Art Gallery concepts to non-orthogonal, generic polygons to perform a multiblock decomposition of faces into a set of maximal, single-loop, convex sub-faces. Concepts such as staircase, dent and notch are used to categorize face concavities and virtual vertices are inserted on smoother concave boundary bends. Multiblocking is performed without the need for Delaunay triangulation with the aid of a notch diagram. A light-weight, mesher-native topology builder that uses virtual topological elements is proposed to construct a virtual topology network from a notch diagram. Subsequently, virtual faces are processed for transfinite meshing . Results indicate the advantage of the proposed technique over existing procedures in terms of orthogonality and anisotropy.

Peer-review under responsibility of organizing committee of the 23rd International Meshing Roundtable (IMR23)

Keywords: Art Gallery; submap meshing; transfinite meshing; multiblocking; vertex-insertion; notch diagram; virtual topology network;

1. Introduction

The increased complexity of finite element analyses of structures, fluid flow and interaction problems lead to pressing quality requirements. Crash analyses limit the variation of element size in anisotropic meshes. A large range of boundary-flow and structural problems require meshes to be boundary structured or partially or completely transfinite. More and more general purpose mesh generators need to be able to cater to most classes of problems. Such meshes of high quality and constraint are often virtually impossible to generate directly on real, complex geometry. Manual partitioning of CAD geometry into well structured zones is time consuming. Submap meshing reduces that effort considerably by automatically “zoning” face sub-areas for transfinite mesh generation.

2. Art Gallery theorem and its applications

The Art Gallery Theorem (aka Watchman Theorem), first proposed by Chvatal [1], provides ground for optimizing the surveillance of art galleries. It states that for a generic, single-loop polygonal floor with n vertices, n/3 floor guards g(n) or surveillance cameras are sometimes necessary and always sufficient.

Peer-review under responsibility of organizing committee of the 23rd International Meshing Roundtable (IMR23).

$$g(n) = n/3 \quad -> \quad \text{Occasionally necessary; always sufficient} \quad (1a)$$

$$g(n) \geq n/3 \quad -> \quad \text{Sometimes necessary} \quad (1b)$$

$$g(n) \leq n/3 \quad -> \quad \text{Always sufficient} \quad (1c)$$

Fisk [2] provided a more compact proof of the theorem, as described above in eqn. 1(a-c), which is best exemplified in Fig. 1(a-b). Depending on the polygonal configuration, more specific inequalities and lemmas [3,4,5] were suggested. Figure 1a. shows for a 12-

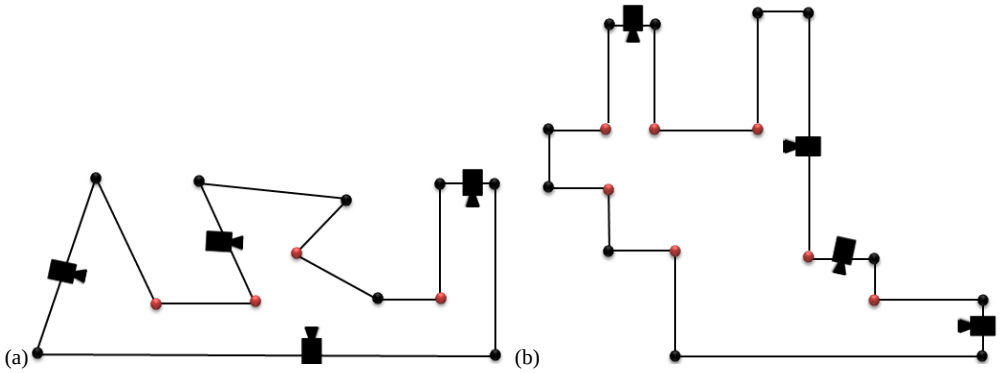

Fig. 1. Concave Art Gallery polygonal floorplan with surveillance cameras mounted on the wall. Red corners indicate concave or reflex vertices. (a) Occasionally necessary number of of guards (g(n) = n/3 = 4; n=12) (b) Sufficient condition (g(n) = 4 < n/3 = 6; n = 18).

sided concave polygonal floor, how exactly (g(n) = 4 = n/3) 4 guards or surveillance cameras can cover the entire floor; where Fig. 1b demonstrates a "sufficient condition" scenario as only 4 guards (g(n) = 4 < n/3=6) or cameras are sufficient to cover the 18-sided floor.

Aggarwal's [6] pioneering work in the application of the theorem initiated various subdivision/multiblocking algorithms for orthogonal polygons. The theorem and its lemmas explore the idea of visibility. The main goal of these algorithms is to efficiently decompose a concave orthogonal polygon to a set of maximal, convex sub-polygons. Reckhow & Culberson [7] used the concept of dents and staircases to build a dent diagram to achieve this in an optimal $O(n_2)$ algorithm. Breen [8] introduced the idea of staircase kernels as an intersection of all covering orthogonally convex polygons. Pape and Vassilev [9] extended the technique to multiloop polygons. Of the many families of lemmas resulting from Aggarwal's Theorem, two relevant and important lemmas are presented in Fig 2. Orthogonal partitioning of a concave polygon from its reflex vertices can result in convex partitioning of the face. The two lemmas referred to here, for any generic polygon with r reflex vertices, can be combined into one and written as

$$\frac{r}{2} + 1 \leq A_n \leq r + 1; \quad (2)$$

where A_n is the number of resulting convex sub-areas. Figure 2a. shows how r (4) shortest orthogonals dropped from r (4) reflex vertices can partition a polygon into r+1 (5) convex sub-areas. Figure 2b. shows the minimal condition where r (8) reflex vertices occur opposite to each other and r/2 (4) shortest orthogonals are dropped to decompose a concave polygon with r (8) vertices into r/2+1(5) convex sub-areas.

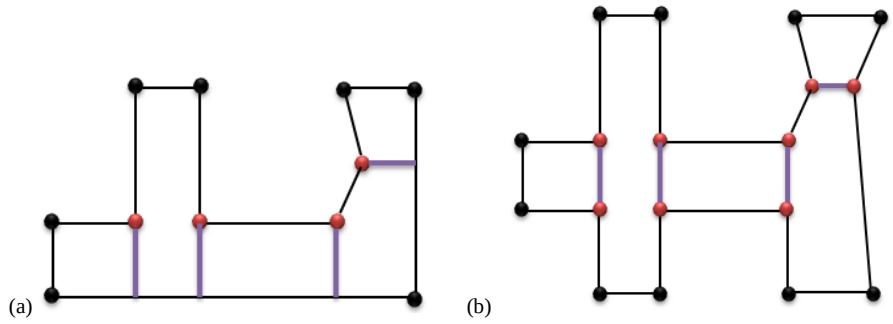

Fig. 2. Orthogonal partitioning of polygonal floors from reflex vertices. All red corners represent reflex vertices, violet solid lines denote orthogonal partition lines. (a) orthogonals dropped from r (r=4) reflex vertices lead to r+1 (5) convex sub-polygons. (b) reflex vertices mirror each other and orthogonals dropped from r (8) reflex vertices produce r/2+1 (5) convex sub-polygons.

3. Main objectives

The primary motivation behind the proposed algorithm is to subdivide concave faces that are predominantly but not necessarily orthogonal (Fig.3) into maximal convex sub-regions to facilitate transfinite meshing. This can be achieved by traditional multiblocking algorithms [10, 11]. However, such algorithms are costly as a fine constrained Delaunay mesh is required in the first place; secondly, the algorithms do not ensure that the Delaunay edges used to decompose the face are orthogonal and finally these algorithms do not guarantee that subdivision will lead to a set of maximal convex sub-regions.

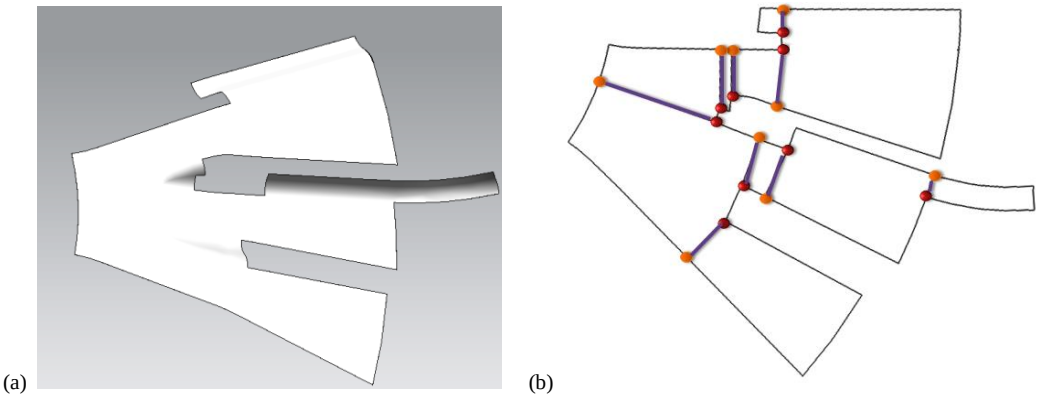

Fig. 3. Orthogonal partitioning of predominantly orthogonal concave face - a) original face (b) desired partitioning in the parameter space.

A maximal convex sub-region is one that cannot be grown any further without crossing the boundary of the original polygon. The algorithm proposed in this paper has the following advantages over conventional multiblocking and submap meshing methods [10,11,12] :

- It does not need a Delaunay mesh
- Guarantees orthogonal partitioning into maximal convex polygons if not limited by boundary constraints.
- Is not dependent on the isotropy of boundary discretization
- Uses a mesher-native virtual topology builder and is thus able to support anisotropic transfinite meshing.

Along similar lines, one might ask why traditional submap meshing methods [11,12] cannot achieve the goals listed above? These methods depend on the isotropy of the boundary seeding; thus, they do not guarantee orthogonality, and as a result don't work well on generic non-orthogonal faces; cannot guarantee face-decomposition into maximal convex sub-regions and are not designed to honor face constraints both exterior (face edges with frozen nodes or hard points) and interior (face-interior hard points), mesh size variation or anisotropy.

The main objectives of this multiblocking/submapping approach are listed as follows:-

- To help generate structured, orthogonal meshes in orthogonal sub-regions of a face (as shown in Fig. 3b).
- Is designed for submap meshing and are applied only to faces with concavity. It aims to subdivide the face into maximal convex polygonal regions.
- Since a virtual topology network is constructed for the virtual faces, it is not controlled by the nature of the initial boundary seeding .
- The virtual topology network allows for any number of face constraints (both interior and exterior) and easily permits anisotropic transfinite meshing within the face and on its boundary. Face-interior hard-points, face edges frozen with nodes from a neighbor face are some typical examples.

4. Art Gallery concept development for Submapping/Multiblocking

The Art Gallery concepts are used to multiblock concave faces for submap meshing via two major steps – first, certain Art Gallery elements are identified and second, these elements are used to create a *notch diagram*. The major lemmas used for a general face are described by eqn 2.

4.1. Art Gallery elements

One of the initial tasks of Art Gallery applications is to walk along the polygon (face) boundary, to identify concavities and categorize them. Concave points or corners, for a simple orthogonal polygon face, are face vertices or edge points that make a right-reversal angle ($> 270^o - \theta_t$, where θ_t is a variable angular tolerance defined in eqn.3) with its immediate neighbors. Some Art Gallery elements this algorithm will use are described in Figure 4. The blue arrows show the walking direction and the face side of the boundary. These elements are defined with respect to the positioning of the concave corners. Two such popular Art Gallery elements as defined by Reckhow and Culberson [7] are dent (Fig.4a) and staircase (Fig. 4b). The dent is characterized by a right-reversal angle at its start and end, the segment in between is called a *dent seat*.

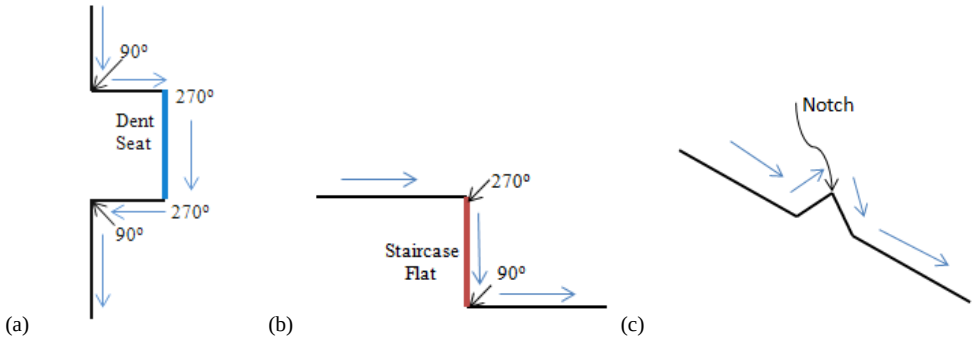

Fig. 4. Examples of Art Gallery Elements – (a) Dent, (b) Staircase and (c) Notch

The staircase is characterized by a right-reversal angle at the start and a right angle ($90^o \pm \theta_t$, θ_t = angular tolerance) at the end and the segment in between is called a staircase flat. Both the dent and the staircase are orthogonal elements. The more generic type of concavity is called a notch (Fig.4c) in this paper. It is defined as a single corner with a right-reversal angle; the leading and trailing segment angles are generic. So the staircase becomes a special case of a notch.

4.2. Dent diagram vs Notch diagram

Leveraging the Art Gallery concepts of dents and staircases, several researchers [7,8,9] have tried to decompose orthogonal polygons into maximal convex polygons. One popular method is the use of a *dent diagram* [7]. As described before, the dent seats and the staircase flats are extended to intersect with the face boundaries and each other. In the process the face gets decomposed into a set of maximal convex polygons or sub-faces (Fig.4). This paper proposes a different approach of face partitioning. It is called a *notch diagram*. Only orthogonal projections will be made from each notch to its closest edge. The resultant orthogonal projection lines (OPL) will be used to decompose the face into sub-faces or virtual faces (Fig.5). The comparison of Figure 5a. and Figure 5b. clearly shows the efficacy and economy of the latter. The *notch diagram*, for most cases leads to the least number of edge-edge intersections and the least number of virtual faces representing a set of maximal convex sub-regions when compared to a *dent diagram*. The virtual faces it produces are maximal convex polygons which cannot be grown any further. The notch diagram shown in Fig.5b proves the lemma described by eqn. 2; that a polygon **P** with r unique concave vertices can be partitioned into at most r+1 convex sub-polygons **p**. Here, with a notch diagram

r = 6; **p** = 6 + 1= 7;

and all convex sub-polygons have been maximized. With the corresponding dent diagram, however **p** = 11. The sub-polygons produced have not been maximized. That can be done as an extra step, either by stopping dent and staircase extensions at their first intersection, or by merging smaller convex regions with convex neighbors into maximal convex sub-polygons. Even when that is done, sometimes narrow sub-polygonal regions can occur.

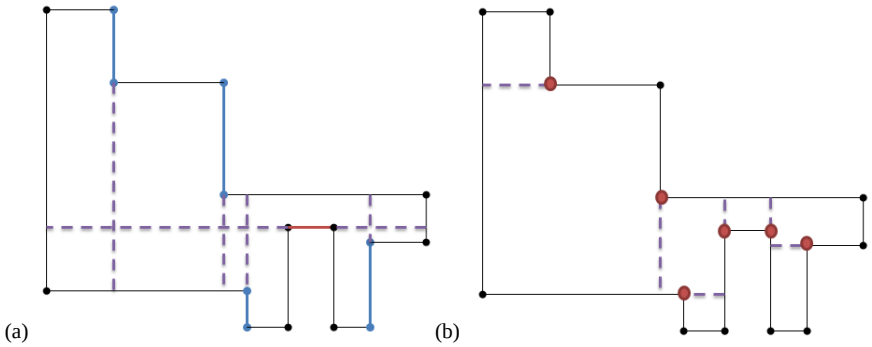

Fig. 5. In a dent diagram (a) for a simple orthogonal polygon (face), Staircase flats are in blue, dent seats are in red and all extension lines dotted violet. In a notch diagram (b) for the same face, all concave corners are in red. All orthogonal projection lines (OPL) are dotted violet.

5. Proposed Multiblocking/Submapping algorithm

The proposed multiblocking/submapping algorithm (Algorithm I) is mostly procedural. It begins with a facetted face flattened to its 2d parameter space [13] and sent to the proposed algorithm which will eventually decompose it (if the face has concavity) into sub-faces or virtual faces which are then sent to the transfinite mesher for map-meshing.

5.1. Concavity determination

The face is first boundary discretized at a size slightly smaller than the user-driven element size which is read as intended feature size. This means if there are features on the face boundary much smaller than the element size, they are meant to be ignored. At each vertex i, the included angle φ_i is computed between the node at the vertex and its trailing and leading nodal neighbors. Based on this angle, a vertex is marked as concave or reflex following the rule

$$\varphi_i > 270^o - \theta_t \qquad (3)$$

where φ_t is the variable angular tolerance (60^o in all examples discussed in this paper).

5.2. Virtual vertex insertion in concave blends

Sometimes, faces can have smooth concave bends like fillets. There may not be a geometry vertex interior to the bend. Whether such bends should be considered "concave" or not is user-driven, by the element size. While traversing the boundary, included angles φ_i are computed at each i-th node. If a certain angle at a non-vertex node, obeys eqn. (3), that node becomes a candidate for virtual vertex insertion. If two successive non- vertex nodes get flagged as "concave", only one virtual vertex is inserted at the average location. An inserted vertex is called reflex virtual vertex. Virtual vertex insertion clearly is a function of the size of boundary discretization. With a seed either too coarse or too fine, one would miss a concave bend. Using chordal tolerance to spot concave bends is also not desirable as it is completely decoupled from the meshing size which provides for user-intention. This is why, for the purpose of vertex insertion a heuristically determined element size factor of 0.7 is used to reduce the user driven mesh size for boundary discretization.

5.3. Building a Notch Diagram for multiblocking

The notch diagram is never physically created but exists in the form of a relational data store. In order to develop the notch diagram, the face boundary is first discretized at element size. All reflex vertices are first projected orthogonally to their closest neighboring edge.

Next a relationship is established between each reflex vertex VR and its pair vertex VP. Some of the reflex vertices can be virtual (VR_v) and typically most of the pair vertices are virtual (VP_v). When all VR are linked with their VP the notch diagram is complete as shown in Fig. 6. In the particular example described in Fig. 6, all red reflex vertices are real geometry vertices whereas all orange projected pair vertices are virtual. In mathematical notations the notch diagram ND can be expressed as a collection of links between n reflex vertices and their orthogonally projected pair vertices given by eqn. 4.

$$ND = \bigcup_{i=1}^{n} \left(V_{ri} \mapsto V_{pi} \right) \qquad (4)$$

Once all virtual vertices are created and the notch diagram has been generated, all nodes on the face boundary are deleted. It should also be mentioned that in some situations when virtual edges intersect, eqn.2 may not be valid as $A_n > r + 1$.

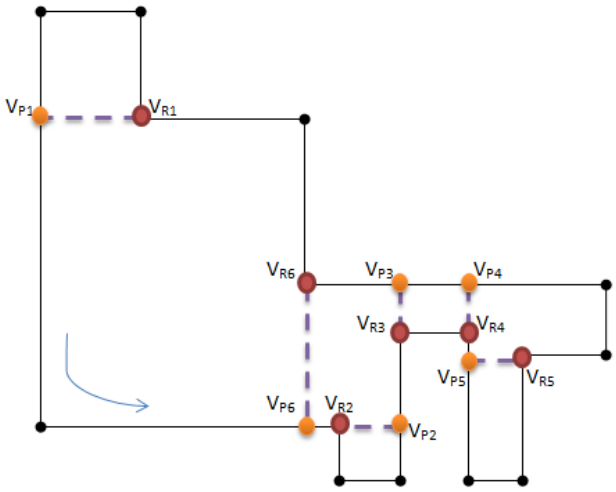

Fig. 6. Notch diagram depicting the linking between vertex pairs – reflex (in red) and their orthogonally projected pairs (in orange) walking the face boundary in a counterclockwise direction.

5.4. Determining the best split line

The notch diagram provides vertex links or maps that can now be used to determine 2D split lines that will subdivide the face into maximal convex sub-regions. However, the notch diagram may not always be the final vertex map or link used to build the virtual topology. Other split line candidates may exist very close to the projected vertices. To achieve this, the boundary is discretized a second time at user-driven element size.

Fig. 7. Use of split line criteria to determine the best split line - (a)Reflex vertex close to OPL; (b) Two closely spaced OPLs

Fig.7 describes two examples of OPLs getting close to other reflex vertices (a) or other OPLs (b). If such OPLs are honored, wafer-thin sub-faces can result leading to undesirable element size reduction in the final mesh. In such scenarios, thus, a compromise needs to be made between orthogonality and proximity. This is achieved, for a candidate split line IJ (Fig. 7b), by a split line factor described as

$$W_{IJ} = \sum_{i=1}^{4} W_{ai} \left| \cos(\alpha_i) \right| + W_l \frac{l_{IJ}}{l_c} \qquad (5)$$

where α_i are the 4 angles made by the split line IJ with its connecting boundary edges (Fig. 7b), W_{ai} are weight factors for each angle and can be made to vary for element types (0.1-0.9; 0.25 for all i, for this paper), W_l is a length factor (1.0-5.0; 2.0 for all examples used in this paper), l_{IJ} is the length of the IJ line and l_c is the characteristic length, i.e. the longest diagonal of the bounding box of the face in 2D. The split line with the smallest W_{IJ} becomes the best selection. If an orthogonally projected vertex is rejected in favor of another reflex vertex as shown in the examples in Fig. 7 (the red dotted line representing the best split line), the V_P's are rejected and it can be written

ALGORITHM I

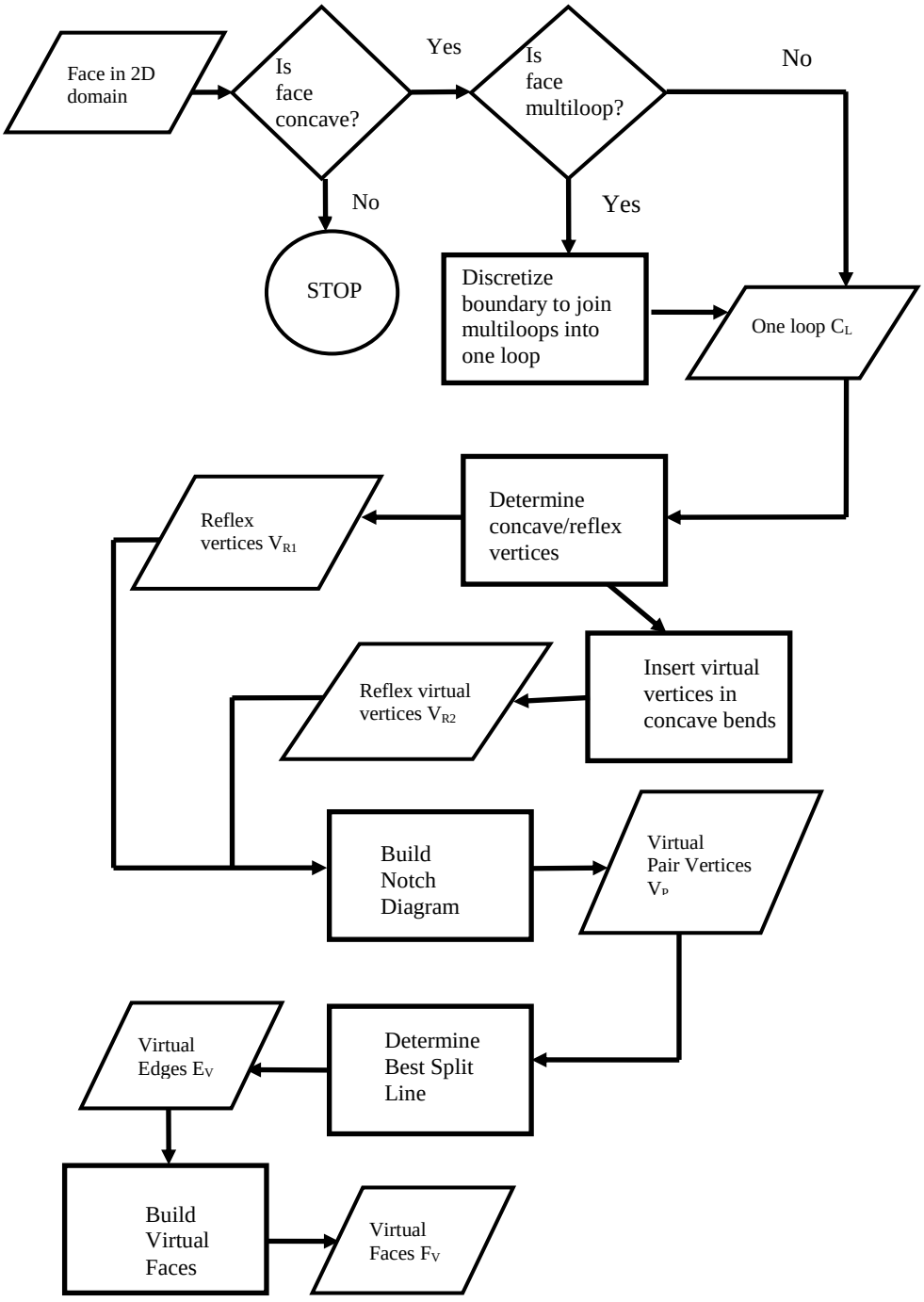

Fig. 8. Flowchart explaining the flow of data and command showing how a single face is processed using the proposed multiblock decomposition technique for submap meshing.

$$V_{ri} \equiv V_{pj}; \quad V_{rj} \equiv V_{pi} \tag{6}$$

The *notch diagram* is updated in the end with new vertex links and all unused virtual vertices are deleted along with the boundary nodes. The overall algorithm (Algorithm I) is described in a nutshell by the flowchart in Fig. 8.

5.5. Multiblocking for multiloop faces

For multiloop faces, the face boundary is first discretized as usual at element size. Next, the parameter space is tiled with a 2D voxel model or Cartesian grid. All boundary nodes are 'grid-marked' and a standard and previously described spatial search algorithm [14] is used to locate proximate node-pairs between loops. The best node-pair is selected based on the split line factor described in eqn. 5. These nodes are now connected with a straight line in the parametric space and discretized further based on element size. As the first inner loop joins the outer, it becomes part of a gradually lengthening outer loop. Every time an inner loop is joined/connected this way with the growing outer loop, the number of unconnected inner loops reduce by one as the outer loop grows. Finally, there is only one long loop with some repeated nodes as shown in Fig. 9. All nodes on the join lines repeat in the final node loop as one has to stop at these nodes twice while walking the loop.

Circular inner loops are always split in four parts and reflex vertices are inserted as discussed in 5.2.

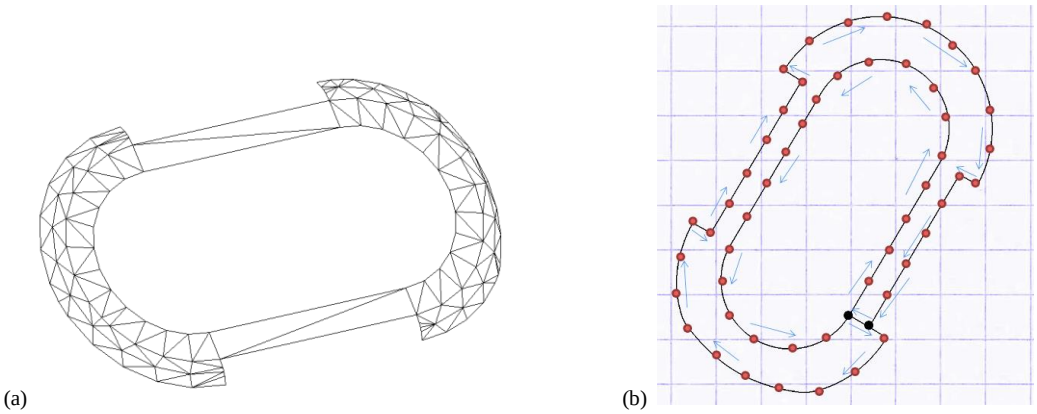

Fig. 9. A multiloop 3D face (a) is boundary discretized and placed in its 2D parametric space against a Cartesian grid which is used to locate 2 nearest nodes (in black) that are connected to form a single node loop where only those 2 nodes repeat.

As soon as the multiloop face is reduced to a single connected nodeloop, a notch diagram is constructed as explained in section 5.2, 5.3 & 5.4 and the problem of multiblocking the face thereafter is exactly as same as that of a single-loop face.

6. Mesher-native topology builder

Once the notch diagram has been finally updated the virtual edges (dotted violet lines in Fig. 5, 6 & 7) are first generated between linked vertex pairs. If virtual edges intersect with each other, they are split at those locations and face interior virtual vertices are created at the intersection points. Given the notch diagram, the virtual edges and vertices and the original face geometry, the mesher-native topology builder constructs the virtual faces. All temporarily generated nodes on the face boundary are deleted at this point. The topology engine is extremely light-weight, contains very little data – mostly pointers/references to nodes/vertices/ edges – and runs in the parameter space. When the final mesh is generated on the virtual faces, the nodes/elements of all virtual faces are associated with the original parent face/edges and the virtual topology is discarded.

6.1. Topology elements

The main topology elements of this multiblock model are listed here –

1. Virtual Vertex – dummy vertices created in three ways – a) by insertion at concave bends on the boundary, b) by orthogonally projecting reflex vertices on edges and c) at the intersection of virtual edges. These are nodes in disguise that have a data-free topological signature assigned to them for identification as vertex.
2. Sub-Edge – when a virtual vertex is introduced on an edge, the edge is topologically split up into two sub-edges, which are references to parts of a geometry edge.
3. Virtual Edge – dummy edges representing split lines (straight lines in 2D) running between reflex vertices and their

pairs. Its data comprises its end points.

4. Virtual Face – subdivision or multiblocked faces into which the virtual edges break the original face.

6.2. Virtual topology network

Once all the virtual edges are created, the virtual face topology is constructed via an edge walking algorithm –

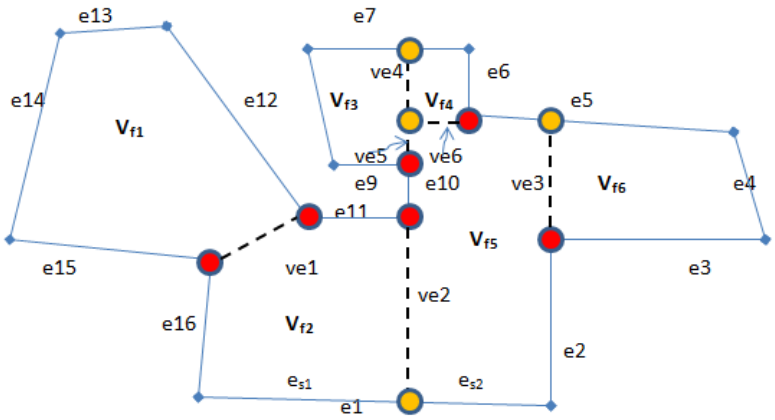

Fig. 10. Topological elements of a multiblocked face with real reflex vertices shown in red, virtual vertices in orange and virtual edges as black dotted lines.

ALGORITHM II

The input to this algorithm is a notch diagram (as shown in Fig. 10) and the virtual topology elements – edges, vertices, v-vertices, v-edges and sub-edges. All face-edges are oriented in a counterclockwise direction. First an STL (Standard Template Library) MAP is formed of all face-edges ($\mathbf{E_{List}}$) and an empty vertex queue, $\mathbf{V_Q}$

While $\mathbf{E_{List}} > 0$
{

1. Starting with the face-interior virtual edges an attempt is made to build virtual faces. When all virtual edges (v-edge) are connected to 2 virtual faces (v-face) virtual topology is established.
2. Starting with a v-edge ($\mathbf{vE_x}$) as a walking edge $\mathbf{vE_w}$(= $\mathbf{vE_x}$) going in any direction or edge use $\mathbf{EU_i}$, a start vertex $\mathbf{V_s}$ and an end vertex $\mathbf{V_e}$ are fixed. The walk starts with $\mathbf{V_e}$ in the loop-direction (CCW) & returns to $\mathbf{V_s}$ using the shortest path.
3. $\mathbf{V_e}$ is added to the v-queue $\mathbf{V_Q}$ if not already added.
4. To determine the shortest path, first, get all adjacent edges connected to $\mathbf{V_e}$.
5. Going over all adjacent edges $\sum\mathbf{E_a}$ we select edge $\mathbf{vE_f}$
    a. If only one of those edges is a v-edge, its adjacent vertex $\mathbf{V_a}$ is selected.
    b. If all edges are real edges and exist in $\mathbf{E_{List}}$, get the one that is farthest along the loop direction. Get its end vertex (going in the original face-loop direction) $\mathbf{V_a}$.
    c. For the very first vertex $\mathbf{V_e}$ of all the adjacent edges, only the ones need to be considered that make a face normal parallel to that of the original face. A cross product of the tangents at the meeting point of the two edges is used to compare with the face normal.
    d. If more than one of the edges are v-edges, the face normal comparison is used to rule out the incorrect ones. If more than one v-edge is correctly oriented with respect to $\mathbf{vE_x}$, the edge that makes the smallest included angle with $\mathbf{vE_x}$, is selected.
6. When the adjacent vertex $\mathbf{V_a}$ is found, it is added to the vertex tree $\mathbf{V_Q}$. This becomes the new $\mathbf{V_e}$. The walking edge is $\mathbf{vE_w} = \mathbf{vE_f}$. We go back to step 3.

7. The walk stop when $\mathbf{V_s}$ is found. The vertex tree $\mathbf{V_Q}$ has a chain of vertices (and thus edges) that will form the v-face loop. The v-face is now created. The vertex Tree $\mathbf{V_Q}$ is cleared.
8. As the v-faces are formed, the real-edges used in the face-loop are removed from the Edge STL MAP $\mathbf{E_{List}}$. When this Edge Map $\mathbf{E_{List}}$ is emptied, the process ends. This means all the boundary edges are part of the newly created v-faces.
9. If the v-edge is still not connected to 2 v-faces, flip its direction, get the other edge-use $\mathbf{EU_j}$ and go back to step 2.

}

7. Transfinite meshing

To set up the transfinite meshing problem pseudo-edges are constructed along the sides of a four/three sided virtual face; a series of pseudo-edge clans [15] are built next and element count conflict is resolved by means of a Corrector-Constrainer solution [15]. Face interior hard-points are honored [16] and anisotropic Transfinite Interpolation or TFIs are commonly used.

8. Results and discussion

Transfinite mesh on the face displayed in Fig.3 is presented in Fig.11. Following the proposed Art Gallery based multiblocking algorithm, the face with 9 reflex vertices decomposes in its parametric space into 10 virtual faces which are map-meshed.

Two other examples are presented to illustrate the efficacy of the proposed algorithm. Fig. 12 (a-d) presents the decomposition of the multiloop face described in Fig.9. Fig. 12a shows the concave vertices (in red) identified from the joined nodeloop and their orthogonally projected virtual vertex pairs (in orange). Accordingly the 2-loop face is decomposed into four virtual faces (Fig. 12b) and eventually map-meshed (Fig. 12c-d).

The element count propagation algorithm gives maximum weight to the most curved split line (in 3D) in the radial direction. This example illustrates the advantages of the proposed algorithm over traditional submap meshing methods [11,12]. In the latter, initial isotropic boundary discretization determines the splits and often the final mesh and thus cannot submap mesh such fillet surfaces with intended anisotropy arising from radial curvature.

Fig. 13 shows a concave face with 4 reflex vertices multiblocked into 5 virtual faces and submap meshed with a constant global element size.

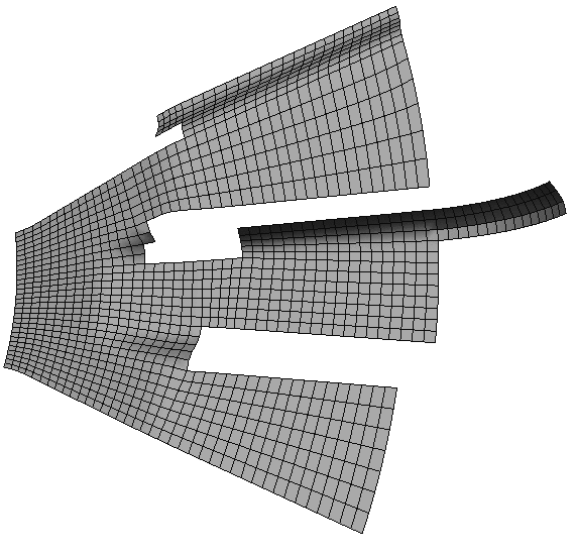

Fig. 11. Submap mesh created by Art Gallery decomposition of the face shown in Fig 3.

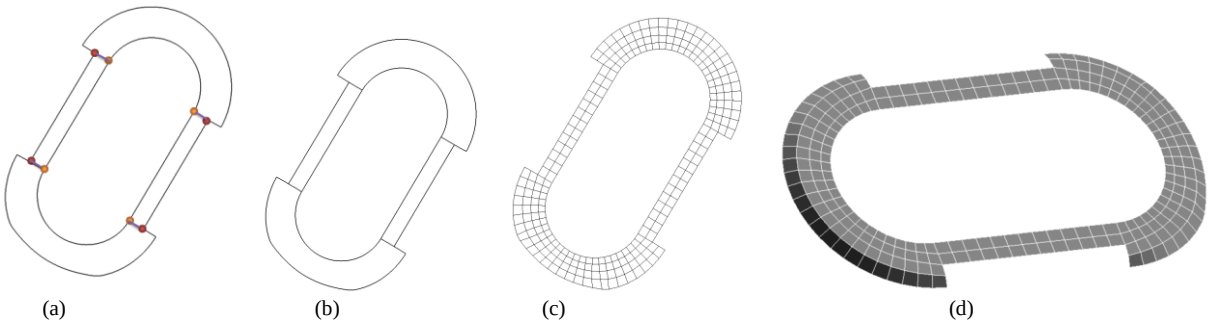

Fig. 12. Steps showing the multiblocking and mesh generation of the multiloop face described in Fig 9.

Two of the OPLs (shown in dotted red) in Fig. 13b do not project to the closest edge, rather they are projected to the next best choices (solid black lines). This is because of constraints on those closest edges. The candidate edge for OPL generation can have constraints – edges frozen with nodes of a neighboring mesh, hard-points, boundary conditions or edge-end-vertices coming too close to the virtual vertex etc. In this case, the virtual vertices due to the OPLs come too close to the edge-end-vertices (less than the global element size), hence the closest edge is rejected and next closest edge is selected for orthogonal projection.

The Art Gallery based multiblocking technique leading to mesher-native virtual topology also makes it possible to create an anisotropic submap mesh. This is a clear advantage over traditional submap meshing methods. Fig.14 illustrates the anisotropic transfinite meshing capability of the proposed multiblocking algorithm on the same face. The concave face is decomposed the exact same way into 5 virtual faces but 5 different sizes, s_1,s_2,s_3,s_4 and s_g are applied to different edges of the face resulting in an anisotropic transfinite mesh. Mainly because of the virtual topology network, this algorithm becomes a great enabler of such anisotropic TFIs.

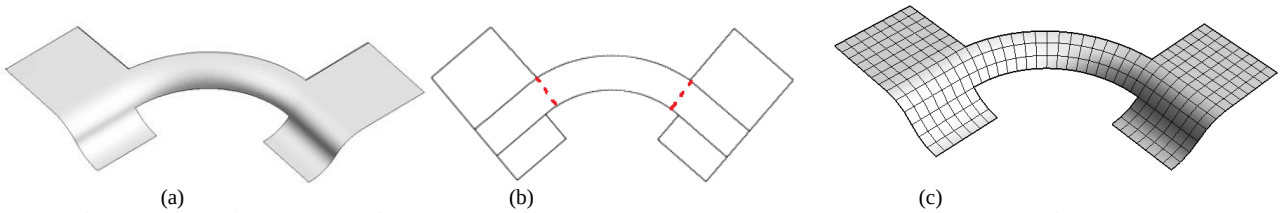

Fig. 13. A facetted concave face (a) with 4 reflex vertices is submap meshed (c) via Art Gallery multiblocking into 5 virtual faces shown in parametric space (b).

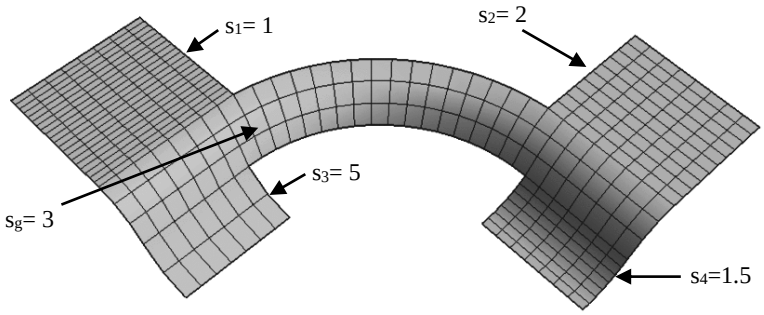

Fig. 14. An example of anisotropic transfinite mesh on the art gallery multiblocked face that uses 4 different sizes on certain edges and a global size (s_g) for the rest of the boundary and interior.

9. Conclusion

In this paper Art Gallery lemmas and concepts are used to decompose concave faces into a set of maximal, single-loop, convex virtual faces. The face is discretized at a size slightly lower than the user driven size and virtual vertices are inserted on smoother concave regions. A notch diagram is next built by orthogonally projecting all reflex vertices to their nearest edges. Best split lines are selected based on a split line criteria that conjointly weighs its length and orientation. Virtual faces are constructed by a lightweight, mesher-native topology builder; these faces are subsequently processed for transfinite meshing. Results indicate four clear advantages over previously reported submapping techniques – firstly, this method does not need a Delaunay mesh; secondly, it does not depend on the isotropy of the boundary seed - it is only used as a first pass to determine the reflex vertices;

thirdly, it has the capability of introducing virtual vertices at points of concavity devoid of real CAD vertices; and finally, it guarantees orthogonal decomposition into maximal convex polygons and allows for a virtual topology network that can provide for several flavors of anisotropy in the transfinite meshes generated.

References

[1] V. Chvatal, A combinatorial theorem in plane geometry, J. Combinatorial Theory. Series B, 18 (1975) 39-41.
[2] S. Fisk, A short proof of Chvatal's watchman theorem, J. Combinatorial Theory. Series B, 24 (1978) 374.
[3] J. Kahn, M. Klawe and D. Kleitman, Traditional galleries require fewer watch-men, SIAM J. Alg.Discrete Meth. 4 (1983) 194-206.
[4] J. O'Rourke, Art gallery theorems and algorithms, Oxford University Press, London, 1987.
[5] F. Hoffmann, M. Kaufmann and K. Kriegel (1991), The art gallery theorem for polygons with holes. Proc. 32nd Symp. Found. Comp. Sc. (1991) 39-48.
[6] A. Aggarwal, The Art Gallery Theorem: Its Variations, Applications and Algorithmic Aspects, PhD Thesis. John Hopkins Univ., Dept. Elec.Engg.Comp.Sc., 1984
[7] R. Reckhow and J. Culberson, Covering a Simple Orthogonal Polygon with a Minimum Number of Orthogonally Convex Polygons., Proc. 3rd Ann. Symp. Comp. Geom., 1987, pp.268-277.
[8] M. Breen, Staircase kernels in orthogonal polygons, Archiv.Mathemtk., 59 (1992) 588-594.
[9] S.Pape and T.Vassilev, Visibilty: Finding the Staircase Kernel in Orthogonal Polygons, A.J.Comp.App.Math. 2(2) (2012) 17-24.
[10] T.Tam and C.G.Armstrong, 2D Finite Element Mesh Generation By Medial Axis Subdivision, Advances in Engineering Software, 13 (1991) 312-324.
[11] E. Ruiz-Giron, Structured and Semi-Structured Algorithms for Hexahedral Mesh Generation, PhD Thesis, Univ. Politecnica de Catalunya, Dept. Math., 2009.
[12] M. Whiteley, D.White, S. Benzeley and T. Blacker, Two and three quarter dimensional meshing facilitators, Engg. Comp. 12 (1996) 144-154.
[13] K. Beatty and N. Mukherjee, Flattening 3D Triangulations for Quality Surface Mesh Generation, Proc. 17th Int. Meshing Roundtable, 2008, pp.125-139.
[14] N. Mukherjee, A Combined Subdivision and Advancing Loop-Front Surface Mesher (Triangular) for Automotive Structures, Int. J. Vehicle Structures & Systems, 2(1) (2010) 28-37.
[15] K. Beatty and N. Mukherjee, A Transfinite Meshing Approach for Body-In-White Analyses, Proc. 19th Int. Meshing Roundtable, 2010, pp.49-65.
[16] N. Mukherjee, High Quality Bi-Linear Transfinite Meshing with Interior Point Constraints, Proc. 15th Int. Meshing Roundtable, 2006, pp. 309-323.