

# Unconstrained Paving & Plastering: A New Idea for All Hexahedral Mesh Generation

Matthew L. Staten, Steven J. Owen, Ted D. Blacker

Sandia National Laboratories<sup>1</sup>, Albuquerque, NM, U.S.A.,  
[mlstate@sandia.gov](mailto:mlstate@sandia.gov), [sjowen@sandia.gov](mailto:sjowen@sandia.gov), [tblack@sandia.gov](mailto:tblack@sandia.gov)

**Summary:** Unconstrained Plastering is a new algorithm with the goal of generating a conformal all-hexahedral mesh on any solid geometry assembly. Paving[1] has proven reliable for quadrilateral meshing on arbitrary surfaces. However, the 3D corollary, Plastering [2][3][4][5], is unable to resolve the unmeshed center voids due to being over-constrained by a pre-existing boundary mesh. Unconstrained Plastering attempts to leverage the benefits of Paving and Plastering, without the over-constrained nature of Plastering. Unconstrained Plastering uses advancing fronts to inwardly project unconstrained hexahedral layers from an *unmeshed* boundary. Only when three layers cross, is a hex element formed. Resolving the final voids is easier since closely spaced, randomly oriented quadrilaterals do not over-constrain the problem. Implementation has begun on Unconstrained Plastering, however, proof of its reliability is still forthcoming.

**Keywords:** mesh generation, hexahedra, plastering, sweeping, paving

## 1 Introduction

The search for a reliable all-hexahedral meshing algorithm continues. Many researchers have abandoned the search, relying upon the widely available and highly robust tetrahedral meshing algorithms [6]. However, hex meshes are still preferable for many applications, and depending on the solver, still required. This paper introduces a new method for hexahedral mesh generation called Unconstrained Plastering.

### 1.1 Previous Research

For all-quadrilateral meshing, Paving [1] and its many permutations [7][8] have proven reliable. Paving starts with pre-meshed boundary edges which are classified into fronts and

---

<sup>1</sup> Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000

The submitted manuscript has been authored by a contractor of the United States Government under contract. Accordingly the United States Government retains a non-exclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for United States Government purposes

advanced inward. As fronts collide, they are seamed, smoothed, and transitioned until only a small unmeshed void remains (usually 6-sided or smaller). Then a template is inserted into this void resulting in quadrilaterals covering the entire surface.

Paving's characteristic of maintaining high quality, boundary-aligned rows of elements is what has made it a successful approach to quad meshing. In addition, because of its ability to transition in element size, Paving is able to match nearly any boundary edge mesh.

There have been many attempts to extend Paving to arbitrary 3D solid geometry. While valuable contributions to the literature, these attempts have not resulted in reliable general algorithms for hexahedral meshing. Plastering [2][3][4][5] was one of the first attempts. In Plastering, the bounding surfaces of the solid are quad meshed, fronts are determined and then advanced inward. However, once opposing fronts collide, the algorithm frequently has deficiencies. Unless the number, size, and orientation of the quadrilateral faces on opposing fronts match, Plastering is rarely able to resolve the unmeshed voids.

Many creative attempts have been made to resolve this unmeshed void left behind by plastering. Since arbitrary 3D voids can be robustly filled with tets, the idea of plastering in a few layers, followed by tet-meshing the remaining void was attempted [9][10]. Transitions between the tets and hexes were done with Pyramids [11] and multi-point constraints. The Geode-Template [12] provided a method of generating an all-hex mesh by refining both the hexes and tets. However, this required an additional refinement of the entire mesh resulting in meshes much larger than required. In addition, the Geode-Template was unable to provide reasonable element quality.

A draw-back of paving is the need for expensive intersection calculations. An alternative to Paving called Q-Morph [7] eliminated the need for intersection calculations by first triangle meshing the surface. This triangle mesh is then "transformed" into a quad mesh. Using a similar advancing front technique to paving, triangles are locally reconnected, repositioned, and combined to form quads. Q-Morph is able to form high-quality quadrilateral elements with similar characteristics to paving. Q-Morph has proven to be a robust and reliable quad meshing algorithm in common use in several commercial meshing packages.

An attempt at extending Q-Morph to a hex-dominant meshing algorithm was done with H-Morph [13]. This algorithm takes an existing tetrahedral mesh and applies local connectivity transformations to the elements. Groups of tetrahedra are then combined to form high-quality hexahedra. The advancing front approach was also used for ordering and prioritizing tetrahedral transformations. Although H-Morph had the desirable characteristics of regular layers near the boundaries, it was unable to reliably resolve the interior regions to form a completely all-hex mesh since it also attempted to honor a pre-meshed quad boundary.

Recognizing the difficulty of defining the full connectivity of a hex mesh using traditional geometry-based advancing front approaches, the Whisker-Weaving algorithm [14][15][16] attempted to address the problem from a purely topological approach. It attempts to first generate the complete dual of the mesh, from which the primal, or hex elements, are readily obtainable. Although whisker-weaving can in most cases generate a successful dual topology, resulting hex elements are often poorly shaped or inverted.

Plastering, H-Morph, Whisker Weaving and all of their permutations are classified as Outside-In-Methods. They start with a pre-defined boundary quad mesh and then attempt to use that to define the hex connectivity on the inside. Another class of Hex meshing algorithms can be classified as Inside-Out methods [17][18][19]. These algorithms fill the inside of the solid with elements first, often using an octree-based grid. This grid is then

adapted to fit the boundary. These methods place high quality elements on the interior of the volume, however, they typically generate extremely poor quality elements on the boundary. In addition, traditional Inside-Out methods are unable to mesh assemblies with conformal meshes. These inside-out methods seem particularly popular with the metal forming industry, but of less appeal in structural mechanics applications.

Sweeping based methods [20][21][22] are among the most widely used hexahedral based meshing algorithms in industry today. Sweeping, however, applies only to solids which are 2.5D, or solids which can be decomposed into 2.5D sub-regions. There has been a considerable amount of research in sweeping and many successful implementations have been published. It is typically quite simple to decompose and sweep simple to medium complexity solids. However, as more complexity is added to the solid model, the task of decomposing the solids into 2.5D sub-regions can be daunting, and in some regards, an art-form requiring significant creativity and experience.

Advancing front methods have proven ideal for triangle, quadrilateral and even tetrahedral meshes. They have been successful in these arenas because of the smaller number of constraints imposed by the connectivity of these simple element shapes. Hexahedral meshes, on the other hand, must maintain a connectivity of eight nodes, 12 edges, and six faces per element, with strict constraints on warping and skewness. As a result, unlike tetrahedral meshes, minor local changes to the connectivity of a hex mesh can have severe consequences to the global mesh structure. For this reason, current hexahedral advancing front methods where the boundary is prescribed apriori have rarely succeeded for general geometric configurations.

Current advancing front methods, while having the high ideal of maintaining the integrity of a prescribed boundary mesh, frequently fail because the very boundary mesh they are attempting to maintain over-constrains the problem, creating a predicament which can be intractable.

To resolve this issue, we introduce a new concept, known as Unconstrained Plastering. With this approach, we relax the constraint of prescribing a boundary apriori quad mesh. While still maintaining the desirable characteristics of advancing front meshes, Unconstrained Plastering is free to define the topology of its boundary mesh as a consequence of the interior meshing process. It is understood that not prescribing an apriori boundary quad mesh can have implications on the traditional bottom-up approach to mesh generation. These implications, however, are significantly outweighed by the prospect of automating the all-hex mesh generation process through a more top-down approach to the problem that Unconstrained Plastering offers.

## 1.2 Unconstrained Plastering, an Unproven Concept

Unconstrained Plastering is a new approach that is unique from all the others. Although it contains similarities to other existing algorithms, primarily Plastering, it should not be considered an extension of Plastering.

Finally, Unconstrained Plastering is not a finished work. Rather, Unconstrained Plastering is an idea that holds promise for hex meshing researchers and should be studied further. Implementation has begun on a prototype and there is reason to be optimistic that it has a greater potential for success than others. However, evidence of it being robust enough to handle general purpose mesh generation for industrial applications is forthcoming.

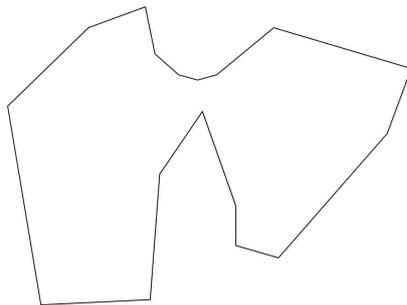
## 2 Unconstrained Paving

To best understand the general concept behind Unconstrained Plastering, we first examine the 2D corollary, Unconstrained Paving.

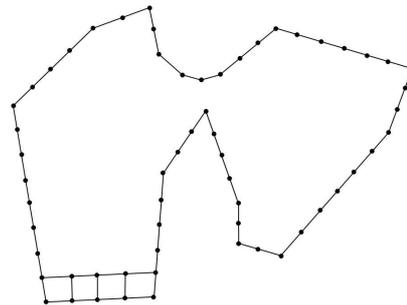
### 2.1 Advancing Unconstrained Rows

Fig. 1 shows a geometric surface ready for quad meshing. If we were to pave this surface, we would first mesh each of the surfaces boundary curves, after which we would advance a row of quads along one of the boundary curves as can be seen in Fig. 2. In this case four quads were added because the curve along which the row was paved was pre-meshed with four mesh edges.

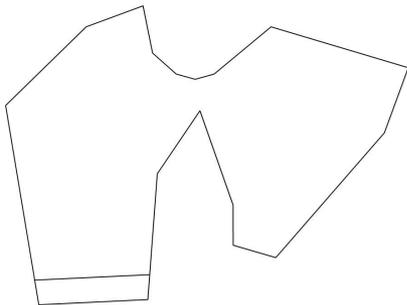
If, instead, the surface was being meshed with Unconstrained Paving, the boundary curves would not be pre-meshed with edges. Advancing, or paving, an unconstrained row would result in Fig. 3. In this case a row of quads have been inserted, however, we do not know how many quads will be in that row. The number of quads in this row is determined as adjacent rows cross it. Fig. 4 shows what the mesh looks like after a second row is advanced. Since the second row advanced crossed the first row advanced, a single quad is formed (shaded) in the corner where the two rows cross. However, both of the rows still have an undetermined number of quads in them.



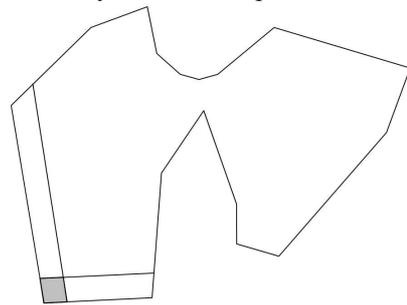
**Fig. 1** Example Surface



**Fig. 2** Example Surface with meshed boundary and one row paved

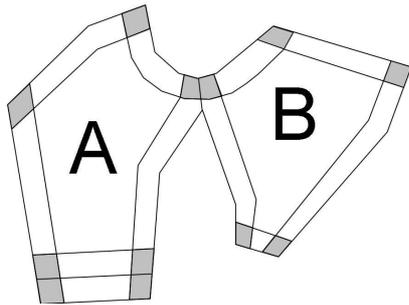


**Fig. 3** One row advanced with Unconstrained Paving

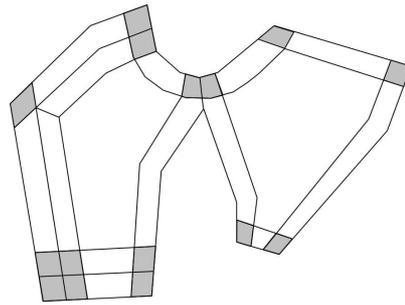


**Fig. 4** Two rows advanced with Unconstrained Paving

In Fig. 5, several additional rows have been advanced and 12 quadrilateral elements have been formed where the various unconstrained rows have crossed. At this point, the un-meshed portion of the surface has been subdivided into two sub-regions (sub-region A & B). It is important to note that both of these unmeshed regions are completely unconstrained. For example, sub-region A is bound by five edges, however, none of these edges has been meshed. Sub-region A is free to be meshed with as many divisions as needed along all of these edges.



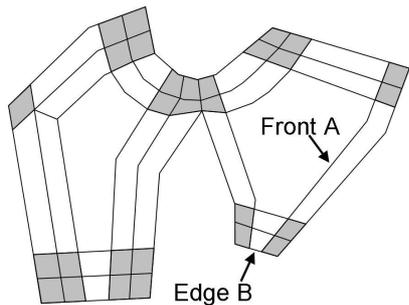
**Fig. 5** Additional rows advanced with Unconstrained Paving



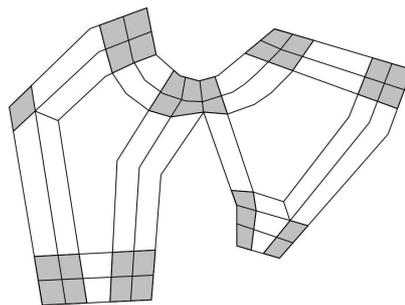
**Fig. 6** Transition row inserted based on large angle between two adjacent rows

## 2.2 Transitioning Unconstrained Rows

Like traditional Paving, Unconstrained Paving has the ability to insert irregular nodes (nodes with more or less than four adjacent quads) in order to transition and fit the shape of the surface. In traditional Paving, this is done by assigning states to the fronts based on angles with adjacent fronts. Unconstrained Plastering is no different. The start and end of an advanced unconstrained row likewise depends upon states and angles. Fig. 6 shows the advancement of an additional row, which, because of angles is the advancement of two previously advanced rows.



**Fig. 7** Front A cannot be advanced normally because Edge B is too short

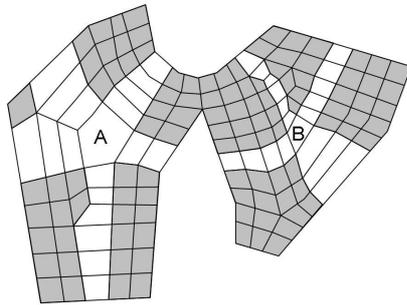


**Fig. 8** Transition row inserted based on front sizes

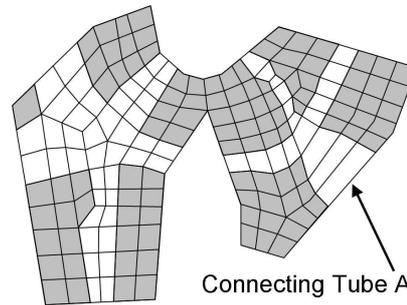
Fig. 7 shows an additional case where rows must be advanced with care. *Front A* is the next front to advance, however, *Edge B* is too short even though angles indicate that the

advancement of *Front A* should extend to *Edge B*. In this case, *Front A* can be advanced as shows in Fig. 8.

Unconstrained rows continue to advance as previously described. Rows bend through the mesh as required to maintain proper quadrilateral connectivity ensuring that all quadrilateral elements created are of proper size. In addition, Paver-like row smoothing and seaming, along with the insertion of tucks and wedges [1] are additional operations that can be performed on the unconstrained rows. Fig. 9 illustrates the example surface and how it may look after several more rows are advanced. All edges on the unmeshed sub-regions A and B are less than two times the desired element size, and so we stop advancing fronts. At this point, it is time to resolve the unmeshed voids.



**Fig. 9** Unconstrained rows advanced leaving only small unmeshed voids and connecting tubes; Quad are shaded, connecting tubes are white.



**Fig. 10** Unmeshed voids have been meshed

### 2.3 Resolving Unmeshed Voids

In general, the unmeshed sub-regions will be any general polygon, with any number of sides. It is assumed that each polygon will be convex. If it is not convex, that would suggest that an additional row needs to be advanced before resolving the unmeshed void. It is also assumed that the size of the polygon is roughly one-to-two times the desired element size. If it is larger than this, then additional unconstrained rows should be advanced until the remaining polygon is one-to-two times the desired element size.

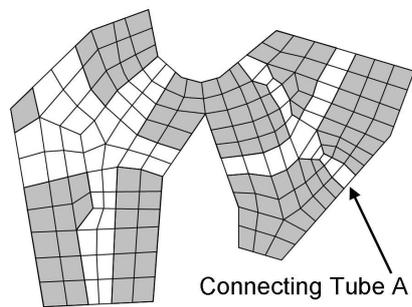
Also, note that the unmeshed region is completely unconstrained. Each of the edges on the unmeshed polygons are connected to the boundary of the mesh through “connecting tubes”. Connecting tubes are the white regions in Fig. 9 that have been crossed by only a single row. The edges of each polygon can be meshed with any number of edges, which will be propagated back to the boundary through the connecting tubes.

At this point, the polygon is meshed with a template quad mesh similar to the templates used to fill the voids during Paving [1]. The template inserted is based on the relative lengths of edges, and angles between edges. In the general case, any convex polygon can be meshed with midpoint subdivision [23]. Midpoint subdivision meshes convex polygons by adding a node at the centroid of the polygon and connecting it to nodes added at the center of each polygon boundary edge. The number of new quads formed is equal to the num-

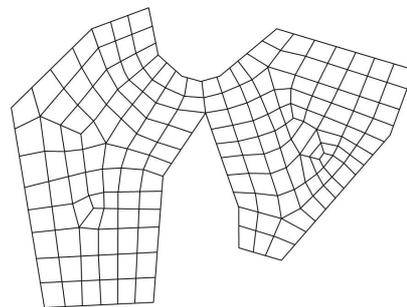
ber of points defining the polygon. Although midpoint subdivision can always be used to mesh the void, simpler templates are often possible.

In Fig. 9, since sub-region B is already four-sided and is of proper size and shape, it can be converted into a single quadrilateral element. However, sub-region A is meshed with midpoint subdivision since it has five sides. The resulting mesh is illustrated in Fig. 10.

Before Unconstrained Paving is finished, the connecting tubes must be examined for size. In Fig. 10 *Connecting Tube A* is much too wide. This can be fixed by advancing a few more rows until the proper size is obtained as shown in Fig. 11. Traditional quadrilateral cleanup operations and smoothing can then be performed to finalize the mesh connectivity and quality as shown in Fig. 12.



**Fig. 11** Sizes in connecting tubes have been resolved



**Fig. 12** Final quad mesh after cleaning and smoothing

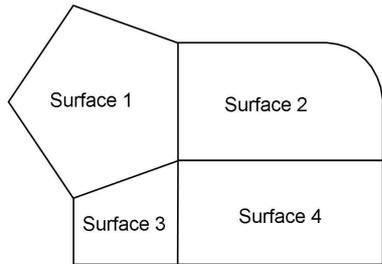
## 2.4 Unconstrained Paving with Multiple Surfaces

In real world models, rarely is the geometry confined to a single surface. For example, sheet metal parts in the auto industry representing automobile hoods often contain thousands of surfaces. Each of these surfaces must share nodes and element edges with its neighboring surfaces across its boundary edges in order to ensure a conformal mesh.

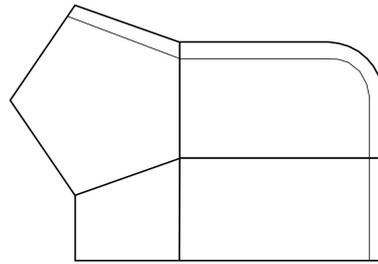
Typically, algorithms that do not pre-mesh the curves of surface before meshing have difficulty ensuring a conformal mesh [17][18][19]. However, Unconstrained Paving can be extended to ensure conformal meshes between any number of surfaces. The penalty, however, is that all of the surfaces must be meshed at the same time. For example, Fig. 13 illustrates four adjacent surfaces which require a conformal mesh. Fig. 14 shows the same model with one unconstrained row advanced. The row was advanced in three of the surfaces. Fig. 15 shows several additional rows inserted and the formation of a tuck in surface 2. Notice that curves which are shared by more than one surface are double-sided fronts advancing into both adjacent surfaces.

After additional rows are advanced, Fig. 16 shows the small unmeshed voids and the connecting tubes. It is important to note that when meshing multiple surfaces at once, the connecting tubes impose additional constraints on how the unmeshed voids can be meshed. However, these constraints can always be satisfied with midpoint subdivision since this would split each edge in every connecting tube exactly once. Fig. 17 shows what the mesh may look like before a final pass through cleanup and smoothing.

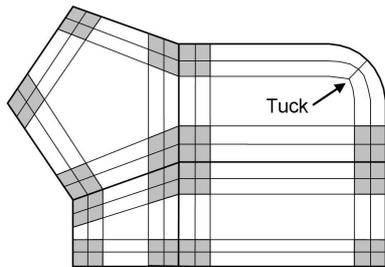
In the current research, Unconstrained Paving is used only as a thought experiment to help illustrate the concepts of Unconstrained Plastering. While implementation of Unconstrained Paving may be beneficial current unstructured quadrilateral meshing techniques satisfy FEA needs. For this reason the current research focuses implementation and prototyping efforts on the 3D Unconstrained Plastering problem.



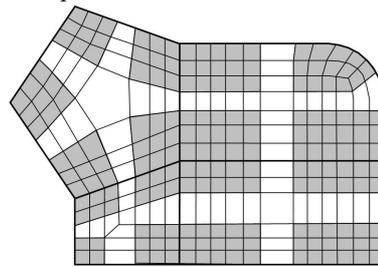
**Fig. 13** Multiple adjacent surfaces requiring a conformal mesh



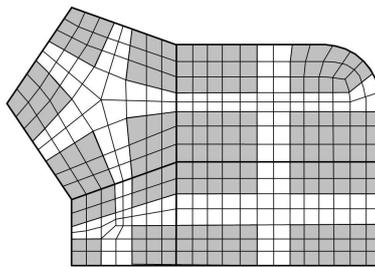
**Fig. 14** An unconstrained row has been advanced extending through multiple surfaces



**Fig. 15** Additional rows are advanced including a tuck



**Fig. 16** Only small voids and connecting tubes remain



**Fig. 17** Unmeshed voids and connecting tubes are meshed

### 3 Unconstrained Plastering

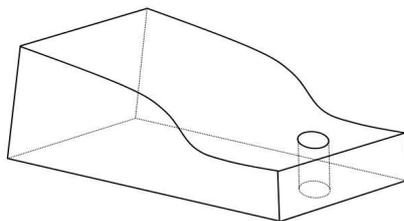
The basic principles of Unconstrained Paving extend to 3D as Unconstrained Plastering. The basic algorithm is as follows and is described in the following sections.

1. Start with a solid assembly with unmeshed volume boundaries.
2. Define fronts, which initially are the surfaces of the volumes.
3. While the unmeshed voids of the solids are larger than twice the desired element size:
  - a. Select a front to advance,
  - b. Based on sizes of fronts, and angles with adjacent fronts, determine which adjacent fronts should be advanced with the current front.
  - c. Advance the fronts
  - d. Form unconstrained columns of hexahedra where 2 layers cross.
  - e. Form actual hexahedral elements where 3 layers cross.
  - f. Perform layer smoothing and seaming.
  - g. Insert tucks and wedges as needed.
4. Identify unmeshed voids, connecting tubes, and connecting webs.
5. Define constraints between unmeshed voids through connecting tubes.
6. Mesh the interior voids with either midpoint subdivision or T-Hex.
7. Sweep the connecting tubes between voids and out to the boundary.
8. Split connecting webs as needed.
9. Smooth all nodes to improve element quality.

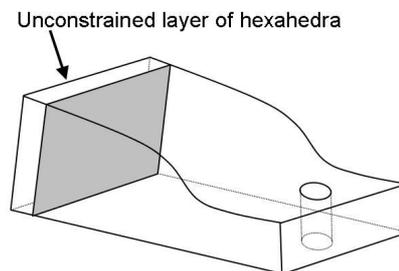
### 3.1 Advancing Unconstrained Layers

The model in Fig. 18 will be used as an example of Unconstrained Plastering. Fig. 19 shows a single unconstrained hexahedral layer advanced. The new surface displayed in Fig. 19 represents the top of the layer of hexes which will be adjacent to the advanced surface. The region between the boundary and the new surface represents an unconstrained layer of hexahedra. It is still unknown how many hexes will be in this layer, however, we do know that it will contain a single layer of hex elements.

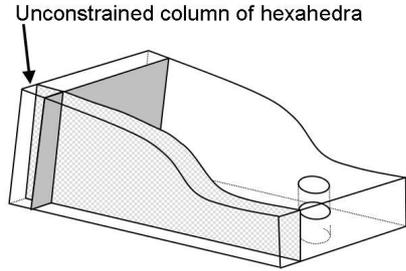
In Fig. 20, a second layer is advanced, which crosses the first layer advanced previously. When two layers cross, a column of hexahedra is formed, however, the size and number of hexahedra in this column will not be determined until additional hex layers cross this column.



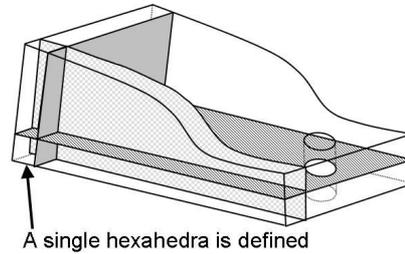
**Fig. 18** Unconstrained Plastering example model



**Fig. 19** One unconstrained hex layer has been advanced



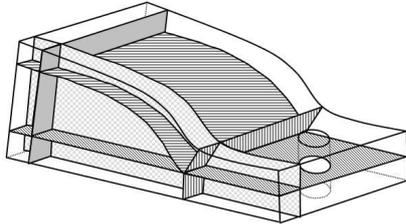
**Fig. 20** When two layers cross, an unconstrained column of hexes is defined



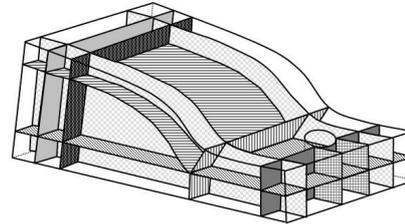
**Fig. 21** When three layers cross a hexahedral element is defined

In Fig. 21, a third layer is advanced, which crosses both of the previously defined layers. Where ever three layers cross, a hexahedral element is formed. In Fig. 21, a single hex element is defined in the lower left corner. Note that until now, no final decisions have been imposed on placement of hexahedra. It is only when three orthogonal layers intersect that hex placement becomes finalized.

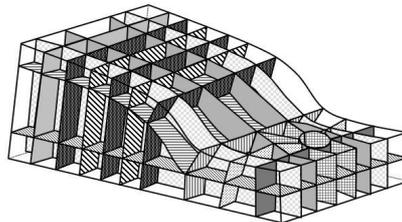
The process continues in Fig. 22, Fig. 23, and Fig. 24. Each time two layers cross a column of hexahedra is defined. Each time a third layer cross a column, a single hexahedral element is defined. During this process, there will be transition layers inserted with logic similar to that described in section 2.2. Layers are advanced until the unmeshed void is approximately twice the desired element size.



**Fig. 22** Additional layers are advanced



**Fig. 23** Additional layers are advanced



**Fig. 24** Additional layers are advanced until the unmeshed void is small

### 3.2 Front Processing Order

A front to advance is a group of one or more adjacent surfaces which are advanced together. The order that fronts are processed in Unconstrained Plastering is very important. This is an area where additional research is required. However, factors to consider when choosing the next front to advance include:

1. Number of layers away from the boundary the front is. Fronts closer to the boundary should be processed first.
2. If the front is “complete” or not. A complete front is a group of surfaces which are completely surrounded by what are referred to as “ends” in Paving and Sub-mapping[24][25]. For example, in Fig. 25, Surface 1 is complete since its boundary is adjacent to a cylindrical face which is perpendicular to Surface 1. In contrast, Surface 2 is incomplete since it is bounded on one of its loops by an “end”, but is bound on its other loop with a “corner” [24]. The best way to proceed would be advance Surface 1 several times until it becomes even with Surface 2, at which the front from Surface 1 and Surface 2 would be combined and advanced as a single front.
3. The size of the front. Smaller fronts should probably be processed first.
4. How much distance there is ahead of the front before a collision will occur. Fronts with a lot of room to advance should probably be processed first.

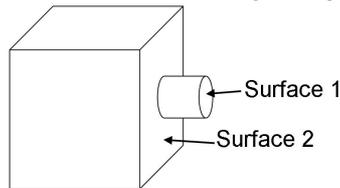


Fig. 25 Surface 1 is a “complete” front, while Surface 2 is “incomplete”

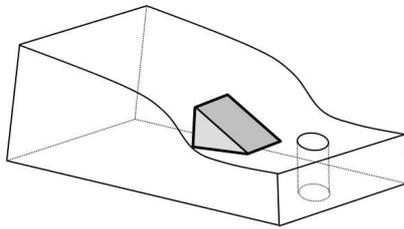
### 3.2 Resolving Unmeshed Voids

Like Unconstrained Paving, there will be unmeshed voids at the center of each volume being meshed. The unmeshed voids can be easily identified because they are the regions in space that have not been crossed by any hex layers. Fig. 26 illustrates the unmeshed void at the center of the example model. In general, these voids define general polyhedra. It is assumed that these polyhedra are convex. If they are not convex, that suggests that an additional layer should be advanced before resolving the voids. Although we have no theoretical basis to prove that advancing additional rows will always ensure a convex polyhedra, experience has shown that it does. Likewise, it is assumed that all edges and faces of the polyhedra are approximately twice the desired element size, or less. If they are larger than this, it suggests that an additional layer should be advanced.

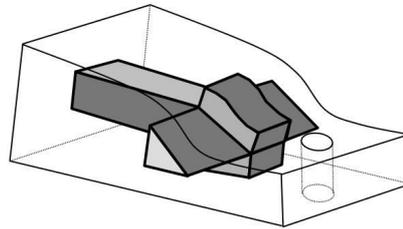
In addition to identifying the unmeshed voids, we must also identify the connecting tubes and connecting webs. Connecting tubes are those regions in space which have been crossed by only a single hex layer as illustrated in Fig. 27. In order to define a hex, three layers must cross, which gives the connecting tubes two degrees of freedom, allowing them to be

swept as a one-to-one sweep[22]. The direction of the sweep is perpendicular to the single layers which already cross the connecting tube. The number of layers in the sweep is the same as the number of hex layers the connecting tube crosses between the unmeshed polyhedra and the boundary surface.

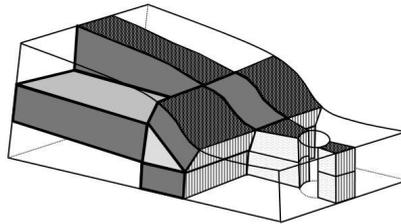
Connecting webs are those regions in space which have been crossed by only two hex layers as illustrated in Fig. 28. Connecting webs only have a single degree of freedom. Essentially, connecting webs represents a layer of hexahedra which will be split the same number of times that the adjacent connecting tubes are split. In this example model, there is one small connecting web section which is not attached to any connecting tubes or unmeshed void. This will happen in locations where seaming has been performed, since seaming will often eliminate or split the unmeshed void. In the example model, the front that was advanced from the front-right surface was seamed with the fronts extending from the hole.



**Fig. 26** The unmeshed void

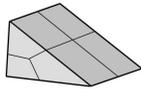


**Fig. 27** The connecting tubes

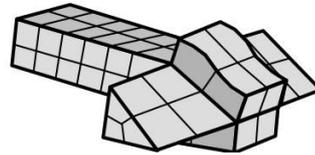


**Fig. 28** The connecting webs

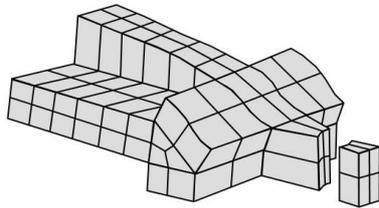
After the unmeshed voids, connecting tubes, and connecting webs have been identified, the unmeshed void is meshed using either midpoint subdivision[23], or T-Hex. Midpoint subdivision is the preferable method since it generates higher quality elements. To determine if midpoint subdivision is possible, a simple count of the number of curves connected to each vertex on the unmeshed polyhedra is done. If there are any vertices which have four or more connected curves, then midpoint subdivision is not possible. The unmeshed void in Fig. 26 can be meshed with midpoint subdivision as illustrated in Fig. 29.



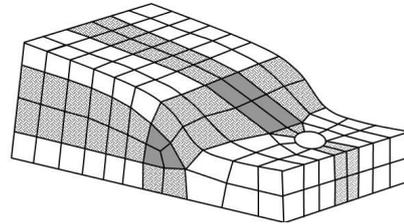
**Fig. 29** Midpoint subdivision of the unmeshed void



**Fig. 30** The connecting tubes are swept



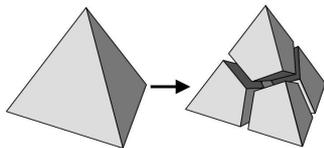
**Fig. 31** The connecting webs are split



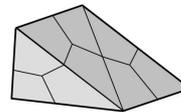
**Fig. 32** Final mesh using midpoint subdivision of voids; connecting tubes are shaded; connecting webs are cross-hatched

After the unmeshed void is meshed, the connecting tubes are swept as shown in Fig. 30 using the mesh from the unmeshed void as the source. Finally, the connecting webs can be split as illustrated in Fig. 31. The final mesh on the example model, after some global smoothing, is shown in Fig. 32. The connecting tubes exposed to the boundary of the mesh are shaded dark and the exposed connecting webs are cross-hatched.

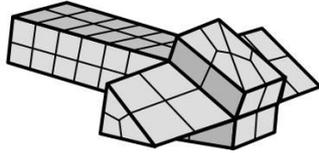
If the polyhedra cannot be meshed with midpoint subdivision, it is meshed with the T-Hex template instead. To do this, we first take each non-triangular polygon on each unmeshed polyhedra and split it into triangles. If the polygon being split is connected to other unmeshed polyhedra through connecting tubes, we must be careful that the face is split the same on both polyhedra so the sweeper can match them up through the connecting tubes. To ensure that they are split the same, a node can be added at the center of the face and the newly created center node. After each face is split into triangles, the polyhedra are meshed with tets. Since we are assuming that the unmeshed void is 1-2 times the desired element size, we would like to mesh these polyhedra without introducing any nodes interior to the polyhedra. Not putting any new nodes in the polyhedra will also help with element quality since T-Hex meshes are worst when T-Hexing around a node surrounded completely by tet elements. After the polyhedra are tet meshed, each tet is split into four hexahedral elements using the T-Hex template shown in Fig. 33. The T-Hex mesh for the polyhedra in the example problem is shown in Fig. 34 and the mesh on the connecting tubes in Fig. 35. Since, in this example, we were meshing only a single solid, there were no constraints between multiple unmeshed voids. Therefore, the quadrilateral face on the top of the polyhedra was split into only two triangles before tetrahedralization. Fig. 36 shows the final mesh after some global smoothing. The connecting tubes exposed to the boundary of the mesh are shaded dark and the exposed connecting webs are cross-hatched.



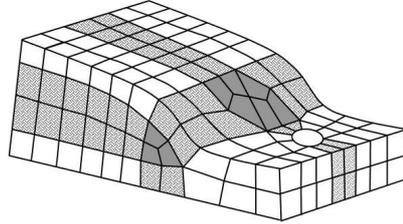
**Fig. 33** T-Hex template



**Fig. 34** T-Hex on unmeshed void



**Fig. 35** The connecting tubes are swept with T-Hex mesh



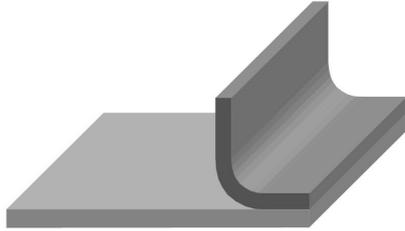
**Fig. 36** Final mesh using T-Hex on voids; connecting tubes are shaded; connecting webs are cross-hatched

T-Hex has long been known as a guaranteed way to get an all hexahedral mesh on nearly any solid geometry. However, the quality of the elements that result is rarely sufficient for most solver codes. Critics of Unconstrained Plastering will point to the use of T-Hex on interior voids as a major downfall of Unconstrained Plastering. However, before that judgment can be made, the following should be considered:

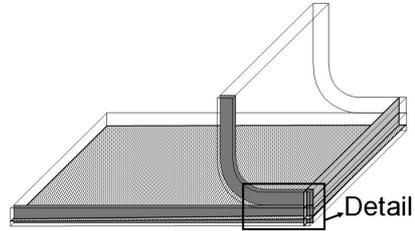
1. T-Hex is only used when interior voids have a vertex with a valence of four or more. In most cases, the interior voids can be meshed with midpoint subdivision.
2. The worst quality hexahedra in T-Hex meshes are found adjacent to nodes which were completely surrounded by tets in the initial tet mesh. This is because a tet mesh can have nodes with a valence of 15 or more, which results in the same number of hexahedra when the T-Hex template is applied. This case should not appear during Unconstrained Plastering, since we assume that enough unconstrained layers have been advanced to make the interior voids small enough to be tet meshed with no interior nodes.
3. The T-Hex looking elements that are swept to the boundary through connecting tubes are not poor in quality since a swept T-Quad mesh is much higher quality than a traditional T-Hex mesh.
4. Any poor quality hexahedra that are formed by Unconstrained Plastering will be in the interior voids which should be on the deep interior of the volumes, with the exception of thin parts which require only one or two layers of hexahedra through the thickness.

### 3.4 Unconstrained Assembly Meshing

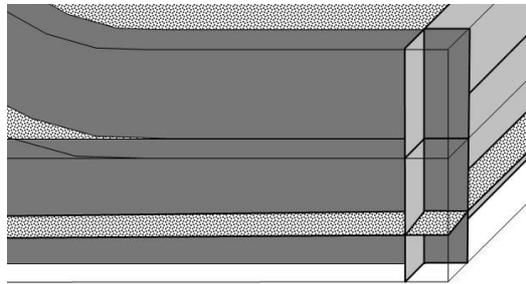
Even though Unconstrained Plastering does not seem capable of honoring existing boundary quad meshes, it can be used to mesh assemblies of solids and still get a conformal mesh. Like Unconstrained Paving, however, all of the volumes in the assembly must be meshed at once. Fig. 37 illustrates a simple assembly model to be meshed. Fig. 38 and Fig. 39 show the same model after 3 unconstrained layers have been advanced. Fig. 39 shows that the unconstrained layers have been extended and advanced through both of the solids in the assembly. Like Unconstrained Paving, surfaces which are shared by both volumes will behave as double-sided fronts advancing into both volumes.



**Fig. 37** Assembly model



**Fig. 38** Assembly model with three advanced unconstrained plastering layers



**Fig. 39** Detail of Fig. 38

### 3.5 Element Quality with Unconstrained Plastering

Implementation of Unconstrained Plastering has not progressed far enough to make any claims on element quality. However, like other advancing front algorithms, Unconstrained Plastering will have the tendency to put the highest quality element near the boundary. Subsequent publications on Unconstrained Plastering will report element quality findings as the research matures.

One limitation that Unconstrained Plastering will have compared to Unconstrained Paving is the lack of hexahedral cleanup operations. Unlike quadrilateral cleanup, hexahedral cleanup operations are limited due to the highly constrained nature of hexahedra [26][27]. As a result, Unconstrained Plastering will be required to create hexahedral topology that will permit good element quality rather than relying on a post-processing cleanup step to fix poor elements.

## 4 Implementation Details

As stated earlier, implementation has begun on a full 3D implementation of Unconstrained Plastering. The authors have chosen to use a faceted surface based approach. The basic algorithm that is being followed is:

1. Triangle mesh all of the boundary surfaces using an element size approximately equal to the desired hexahedral element size.
2. Traverse through this triangle mesh to eliminate any unnecessary CAD artifacts (i.e. small angles, slivers, etc).
3. Divide the triangles up into groups which form Surfaces or Fronts. They are grouped together considering the original CAD topology, but also dihedral angles between the original CAD surfaces.
4. For each volume in the assembly being meshed, form a "cell". Each cell has three associated layer ids. These initial cells get (UNDEFINED, UNDEFINED, UNDEFINED) as their initial layer ids.
5. While any cell is larger than twice the desired element size:
  - a. Choose a set of Surfaces to advance to form a new layer.
  - b. Advance the triangle mesh on these surfaces into the volume. A faceted surface is created offset by the desired element size to the advancing surfaces.
  - c. Form a new cell between each newly created surface its corresponding front surface. This new Cell inherits the layer ids from the cell being advanced into. It is also assigned a new layer id which represents the layer just created.
  - d. Smooth and seam the newly advanced faceted surface with its neighboring surfaces.
6. Create constraints between the unmeshed voids through the connecting tubes.
7. Mesh each unmeshed void with either midpoint subdivision or T-Hex.
8. Sweep the connecting tubes.
9. Split the connecting webs as needed.
10. Send the entire mesh to a smoother for global smoothing. Smoothing is needed on curves and surfaces, in addition to the nodes on the interior of the volumes.

## 5 Conclusions

The concept of advancing unconstrained rows of quads and layers of hexahedra has been introduced through the algorithms of Unconstrained Paving and Unconstrained Plastering. The concept is most relevant with Unconstrained Plastering since it eliminates the problems of resolving highly constrained unmeshed voids which is common with most other advancing front hexahedral meshing algorithms.

The algorithms presented are able to mesh assembly models with conformal meshes with the penalty that all of the volumes/surfaces in the model must be meshed at the same time. Meshing all of the volumes in an assembly at once increases memory requirements since the mesh on the entire assembly will need to be in the mesher's internal datastructures at once, which are typically larger than mesh storage datastructures.

Implementation of Unconstrained Plastering has begun and the authors are optimistic that it will be more successful than other free hex meshing algorithms. However, additional research is required before any claims will be made.

Unconstrained Paving is also presented which is a potential improvement upon traditional advancing front quadrilateral meshing algorithms. However, since the quadrilateral meshing problem already has several solutions, the priority of researching and implementing Unconstrained Paving is lower than that of Unconstrained Plastering.

## References

---

- 1 T. D. Blacker, M. B. Stephenson. "Paving: A New Approach to Automated Quadrilateral Mesh Generation", *International Journal for Numerical Methods in Engineering*, 32, 811-847 (1991).
- 2 S. A. Canann, "Plastering: A New Approach to Automated 3-D Hexahedral Mesh Generation", *American Institute of Aeronautics and Astronautics* (1992).
- 3 J. Hipp, R. Lober, "Plastering: All-Hexahedral Mesh Generation Through Connectivity Resolution", *Proc. 3<sup>rd</sup> International Meshing Roundtable* (1994).
- 4 S. A. Canann, "Plastering and Optismoothing: New Approaches to Automated 3D Hexahedral Mesh Generation and Mesh Smoothing", Ph.D. Dissertation, Brigham Young University, Provo, Utah, USA (1991).
- 5 T. D. Blacker, R. J. Meyers, "Seams and Wedges in Plastering: A 3D Hexahedral Mesh Generation Algorithm", *Engineering With Computers*, 2, 83-93 (1993).
- 6 P.-L. George, H. Borouchaki, "Delaunay Triangulation and Meshing: Application to Finite Elements", © Editions HERMES, Paris, 1998.
- 7 S. J. Owen, M. L. Staten, S. A. Canann, S. Siagal, "Q-Morph: An Indirect Approach to Advancing Fron Quad Meshing", *International Journal for Numerical Methods in Engineering*, 44, 1317-1340 (1999).
- 8 D. R. White, P. Kinney, "Redesign of the Paving Algorithm: Robustness Enhancements through Element by Element Meshing", *Proc. 6<sup>th</sup> Int. Meshing Roundtable*, 323-335 (1997).
- 9 D. Dewhirst, S. Vangavolu, H. Wattrick, "The Combination of Hexahedral and Tetrahedral Meshing Algorithms", *Proc. 4<sup>th</sup> International Meshing Roundtable*, 291-304 (1995).
- 10 R. Meyers, T. Tautges, P. Tuchinsky, "The 'Hex-Tet' Hex-Dominant Meshing Algorithm as Implemented in CUBIT", *Proc. 7<sup>th</sup> International Meshing Roundtable*, 151-158 (1998).
- 11 S. J. Owen, S. Canann, S. Siagal, "Pyramid Elements for Maintaining Tetrahedra to Hexahedra Conformability", *Trends in Unstructured Mesh Generation*, AMD Vol 220, 123-129, ASME (1997).
- 12 R. W. Leland, D. Melander, R. Meyers, S. Mitchell, T. Tautges, "The Geode Algorithm: Combining Hex/Tet Plastering, Dicing and Transition Elements for Automatic, All-Hex Mesh Generation", *Proc 7<sup>th</sup> International Meshing Roundtable*, 515-521 (1998).

- 13 S. J. Owen, "Non-Simplicial Unstructured Mesh Generation", Ph.D. Dissertation, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA (1999).
- 14 T. J. Tautges, T. Blacker, S. Mitchell, "The Whisker-Weaving Algorithm: A Connectivity Based Method for Constructing All-Hexahedral Finite Element Meshes", *International Journal for Numerical Methods in Engineering*, 39, 3327-3349 (1996).
- 15 P. Murdoch, S. Benzley, "The Spatial Twist Continuum", *Proc. 4<sup>th</sup> International Meshing Roundtable*, 243-251 (1995).
- 16 N. T. Folwell, S. A. Mitchell, "Reliable Whisker Weaving via Curve Contraction," *Proc. 7<sup>th</sup> International Meshing Roundtable*, 365-378 (1998).
- 17 R. Schneiders, R. Schindler, R. Weiler, "Octree-Based Generation of Hexahedral Element Meshes", *Proc. 5<sup>th</sup> International Meshing Roundtable*, 205-217 (1996).
- 18 P. Kraft, "Automatic Remeshing with Hexahedral Elements: Problems, Solutions and Applications", *Proc. 8<sup>th</sup> International Meshing Roundtable*, 357-368 (1999).
- 19 G. D. Dhondt, "Unstructured 20-Node Brick Element Meshing", *Proc. 8<sup>th</sup> International Meshing Roundtable*, 369-376 (1999).
- 20 T. D. Blacker, "The Cooper Tool", *Proc. 5<sup>th</sup> International Meshing Roundtable*, 13-29 (1996).
- 21 Mingwu Lai, "Automatic Hexahedral Mesh Generation by Generalized Multiple Source to Multiple Target Sweeping", Ph.D. Dissertation, Brigham Young University, Provo, Utah, USA (1998).
- 22 M. L. Staten, S. Canann, S. Owen, "BMSweep: Locating Interior Nodes During Sweeping", *Proc. 7<sup>th</sup> International Meshing Roundtable*, 7-18 (1998).
- 23 T. S. Li, R. M. McKeag, C. G. Armstrong, "Hexahedral Meshing Using Midpoint Subdivision and Integer Programming", *Computer Methods in Applied Mechanics and Engineering*, Vol 124, Issue 1-2, 171-193 (1995).
- 24 D. R. White, L. Mingwu, S. Benzley, "Automated Hexahedral Mesh Generation by Virtual Decomposition", *Proc. 4<sup>th</sup> International Meshing Roundtable*, 165-176 (1995).
- 25 D. R. White, "Automatic, Quadrilateral and Hexahedral Meshing of Pseudo-Cartesian Geometries using Virtual Subdivision", Master's Thesis, Brigham Young University, Provo, Utah, USA (1996).
- 26 M Bern, D. Eppstein, "Flipping Cubical Meshes," *Proc. 10<sup>th</sup> International Meshing Roundtable*, 19-29 (2001).
- 27 P. Knupp and S. A. Mitchell, "Integration of mesh optimization with 3D all-hex mesh generation," *Tech. Rep. SAND99-2852*, Sandia National Laboratories, 1999, <http://citeseer.ist.psu.edu/knupp99integration.html>.