2010-08-06

An Adaptive Grid-Based All Hexahedral Meshing Algorithm Based on 2-Refinement

Jared D. Edgel
Brigham Young University - Provo

An Adaptive Grid-Based All Hexahedral Meshing Algorithm Based On 2-Refinement

Jared Edgel

A thesis submitted to the faculty of
Brigham Young University
In partial fulfillment of the requirements for the degree of

Master of Science

Steven E. Benzley, Chair
Paul W. Richards
Steven J. Owen

Department of Civil and Environmental Engineering

Brigham Young University

December 2010

ABSTRACT

An Adaptive Grid-Based All Hexahedral Meshing

Algorithm Based On 2-Refinement

Jared Edgel

Department of Civil and Environmental Engineering

Master of Science

Adaptive all-hexahedral meshing algorithms have many desirable features. These algorithms provide a mesh that is efficient for analysis by providing a high element density in specific locations, such as areas of high stress gradient or high curvature and reduced mesh density in other areas of less importance. In addition, inside-out hexahedral grid based schemes, using Cartesian structured grids for the base mesh, have shown great promise in accommodating automatic all-hexahedral algorithms. In these algorithms mesh refinement is generally used to capture geometric features.

Unfortunately, most adaptive mesh generation algorithms employ a 3-refinement method. This method, although easy to employ, provides a mesh that is coarse in most areas and highly refined in other areas. Because a single refined hex is subdivided into 27 new hexes, regardless of the desired refinement, there is little control on mesh density.

This paper will present an adaptive all-hexahedral grid-based meshing algorithm that employs a 2-refinement insertion method. 2-refinement is based on dividing a hex to be refined into eight new hexes. This allows greater control on mesh density which in turn increases computational efficiency.

Keywords: meshing, refinement, adaptation, hexahedral

ACKNOWLEDGMENTS

I'd like to thank Steven E. Benzley for the opportunity he created here at BYU for me to get my masters degree and for supporting me through the entire process. I'd also like to thank him for his help and contributions throughout the technical process. I'd like to thank Steven J. Owen for his support as well as for all of his technical contributions and for the continual relationship he helps create between Sandia and BYU that provides funding and gave me this opportunity. I'd like to thank Timothy Miller for his help through the technical process. I'd like to thank Mark Dewey for his technical support throughout the programming process. I'd like to thank Bradley Mecham for his help through the technical process as well as his help in writing C++ code to program these algorithms. Lastly, I'd like to thank my wife Jennifer Edgel, whose love and support enabled me to complete this work.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

Tetrahedral and hybrid (i.e. combined tetrahedral and hexahedral) meshes are very robust, simple to implement, can automatically generate a mesh for arbitrary bodies, and are thus used in most three dimensional meshing applications. However, all-hexahedral meshes are often preferred for use in high fidelity computational structural mechanics applications. An algorithm that automatically creates a hexahedral mesh for arbitrary geometries remains a challenging problem [1]. The ability to adapt (i.e. coarsen and refine) all hexahedral meshes is an area of current research [1], [2], [3] and [4].

1.1 Grid-Based Methods

A promising solution to the adaptive hex-based meshing algorithm problem includes grid-based methods [1], [5], [6] and [7]. One of the main difficulties with these algorithms is matching the boundary of the hexes to the boundary of the geometric domain. An octree method effectively lends itself to this problem by allowing control on mesh resolution to account for high geometric curvature, thin geometric sections, and areas of high gradients. Ito, et al [1] employs an all hex grid-based method for the use in biological models. Their work proposes a simple set of templates that handle most cases of octree refinement.

1.2 3-Refinement

However, as is the case in [1] most octree methods are based on 3-refinement. This includes Schneiders initial work [2], [5] as well as Zhang [6] and others [1], [8]. 3-refinement involves splitting a selected octant within an octree three times along one edge. For three dimensions this involves uniformly splitting a single hex into 27 new hexes as shown in Figure 1-1. This 1-to-27 split often results in meshes containing highly coarse and highly refined sections. This lack of control on mesh density reduces efficiency as computation time is directly related to the number of hexes in a given mesh. The examples given in Zhang [6] clearly shows the nature of 3-refinement. 3-refinement can also introduce artificial gradients which can affect solution quality.

1.3 2-Refinement

The main contribution of this thesis is the development of efficient algorithms using 2-refinement for grid-based adaptive all-hex meshes. 2-refinement involves splitting the edge of an octant twice and, in three dimensions, results in a 1-to-8 split, also shown in Figure 1-1. This refinement provides greater control on mesh density and allows a much smoother transition from areas of higher to lower mesh densities.

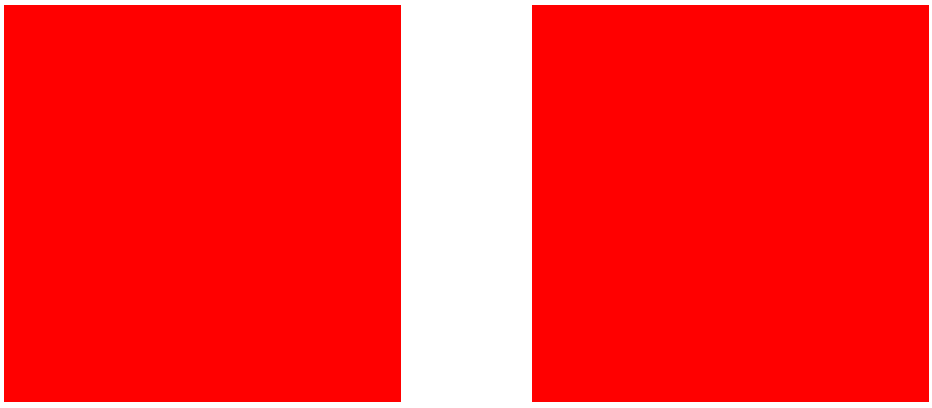


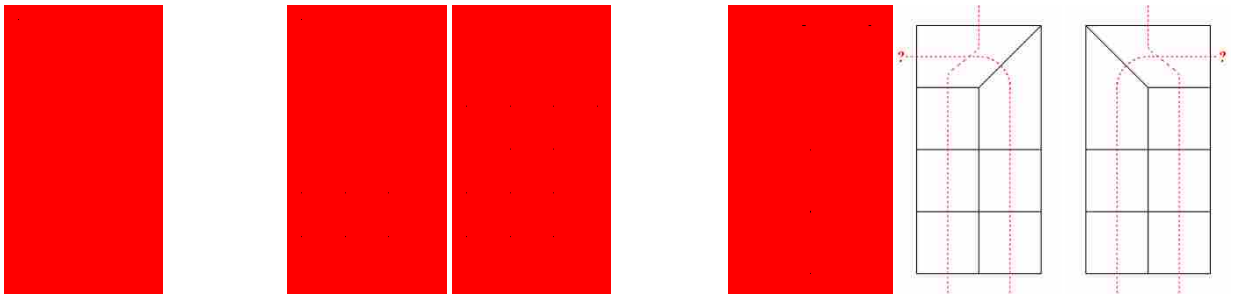
Figure 1-1: 3-Refinement and 2-Refinement

2 BACKGROUND

Hexahedral 3-refinement is often considered to be the method of choice for refinement when compared with 2-refinement [2]. This is because 3-refinement is quite simple to employ. Figure 2 (a) shows an octree with 2 octants. One octant has been chosen to be refined and is shaded grey. Figure 2-1 (b) shows the refinement being performed as 3-refinement. The dashed lines shown in this figure are the dual "chords" of the mesh [9], [10]. Consideration of the duals is a convenient way to demonstrate the basic difference between 2- and 3- refinement. The completed duals and mesh after templates have been inserted are shown in Figure 2-1 (c). Conformity is reestablished, where the middle dual is connected to the opposite dual, and the remaining two duals on either side are connected together. This solution is straight forward and maintains conformity all within the original octant.

2.1 2-Refinement Issues

The same problem is presented for 2-refinement as shown in Figure 2-1 (d). The conformable refinement solution here is not as simple as the 3-refinement case and, in fact, introduces conformability issues beyond the unrefined octant as shown in Figures 2-1 (e) and (f) (i.e. note the dangling chords still present in both options). The examples presented here are simple and are meant for illustrative purposes only. They show that a more sophisticated algorithm is required for developing 2-refinement algorithms.



(a) (b) (c) (d) (e) (f)

Figure 2-1: Comparison of 3- and 2-Refinement

Figures 2-1 (e) and (f) show that 2-refinement leaves a dangling dual. This newly created dual needs to connect to another newly created dual or must exit on a geometric boundary. In an unstructured grid there can be no guarantees that a newly created dual will be available for this connection without further altering the mesh. To ensure that this connection can be made this paper makes the tacit requirement that the initial mesh be structured. This ensures that every newly created dual will have a neighboring partner and such grids are the standard for inside-out hexahedral grid based schemes as seen in [2] and [7].

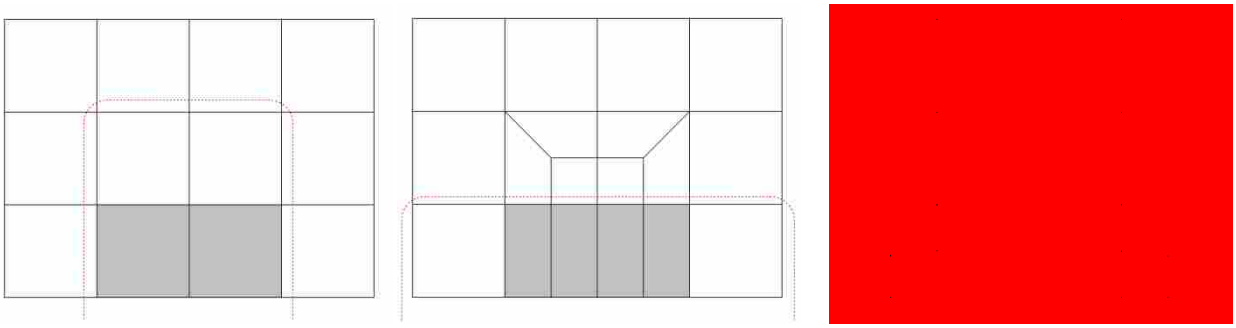
Schneiders [2] proposed a 2-refinement algorithm in two and three dimensions. His paper has useful templates as well as an efficient method for 2-refinement in two dimensions. Unfortunately he never illustrated a full three dimensional 2-refinement algorithm, nor is there evidence of implementation of such an algorithm (i.e. examples, figures, etc).

Marechal [7] has proposed a 2-refinement algorithm that is based on using templates on the dual of a mesh. His algorithm is not based on mesh adaptation but creates the mesh from scratch. It is based on an octree overlay grid which uses a four to two transition rather than a two to one transition. His algorithm is capable of refining concave regions and the examples shown look fully robust. However every template used has a hex with two 45 degree angles and a

single transition layer is used in three dimensional meshing. Both of these limitations will affect element quality of the mesh as seen in his examples. The minimum scaled jacobian value in all of his examples is 0.004.

2.2 2-Refinement by Pillowing

Pillowing is a technique proposed by Mitchell [9] to insert a connected series of conformable quadrilateral or hexahedral elements into an existing mesh. This technique can be used to introduce the necessary elements in a mesh and produce refined regions. Mesh refinement by pillowing was first investigated by Harris [3] and then revisited by Parish. The results of these two studies indicated that 2-refinement by pillowing was viable, but it was much more computationally expensive than 2-refinement by templates [3]. Figure 2-2 illustrates how pillowing is used to create the refined areas, shaded grey, as well as the necessary transition elements. Pillowing can handle all possible concave and convex situations. However, a new pillow must be introduced in order to refine in each of the X, Y and Z directions. Every time a pillow is inserted, old elements are deleted and twice as many new elements are created. This creates an exponential computation time which is impractical for large meshes.



(a) Initial unrefined mesh. First pillow is dashed line.

(b) Mesh after pillowing once. Second pillow is dashed line.

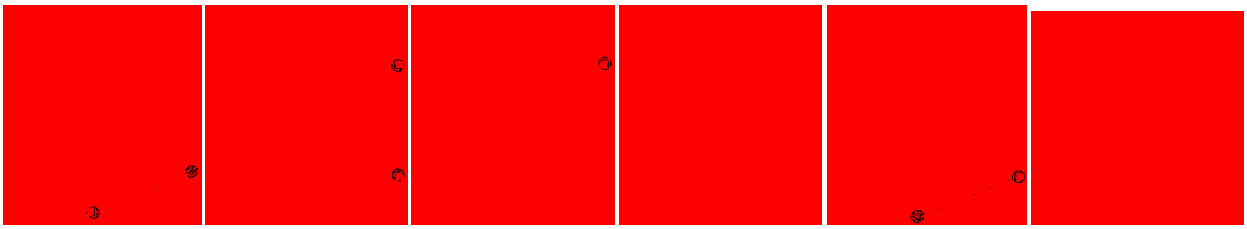
(c) Final refined result.

Figure 2-2: 2-Refinement by Pillowing

3 TEMPLATES

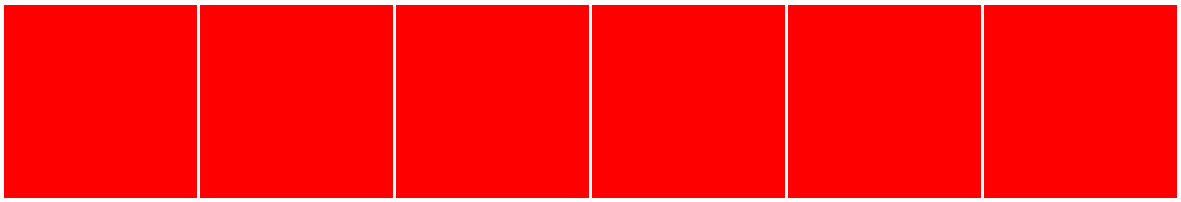
Grid-based algorithms often employ templates for the insertion of conformal elements into a transition layer between the uniform refinement zone and the original hexes. Schneiders [2] and [5] introduced the first templates to be used for 3-refinement. Zhang [8] introduced new templates for 3-refinement that created fewer elements in the transition zone but came with a penalty in element quality. They also introduced two additional templates used for diagonal vertices being refined. In general, 3-refinement templates have significantly decreased element qualities.

Figure 3-1 shows the 2-refinement templates with their associated average and minimum scaled jacobian values. The scaled jacobian is a quality measure often used to evaluate element quality [11]. An exhaustive search to find these templates is given in Appendix A. Common 3-refinement templates are shown in Figure 3-2 along with their minimum and average scaled jacobian values. As shown, element quality with 2-refinement templates is substantially higher than 3-refinement templates with the minimum scaled jacobian value of 0.5774 compared to 0.3077. The shaded nodes and edges in Figure 3-1 are used for orienting the template when it is inserted into the transition zone and are called directional nodes and edges.



(a)	(b)	(c)	(d)	(e)	(f)
Ave: 1.0000	Ave: 0.8047	Ave: 0.6830	Ave: 1.0000	Ave: 0.8047	Ave: 1.0000
Min: 1.0000	Min: 0.7071	Min: 0.5774	Min: 1.0000	Min: 0.7071	Min: 1.0000

Figure 3-1: 2-Refinement Templates. Directional nodes and edges are shaded black



(a)	(b)	(c)	(d)	(e)	(f)
Ave: 1.0000	Ave: 0.6264	Ave: 0.5791	Ave: 1.0000	Ave: 0.7354	Ave: 1.0000
Min: 1.0000	Min: 0.4264	Min: 0.3077	Min: 1.0000	Min: 0.5547	Min: 1.0000

Figure 3-2: 3-Refinement Templates

Most grid-based algorithms employ a transition zone that is one hex layer thick [1], [2], [6] and [7]. This paper uses a transition zone of two hex layers. This introduces extra restrictions on the refinement but produces higher quality elements and provides a smoother transition from high to low mesh density regions.

As mentioned in Figure 2-1, the difficulties of 2-refinement require a sophisticated algorithm. What follows are the three major steps used to develop a fully robust 2-refinement algorithm using template insertion and a two layer thick (i.e. staggered) transition zone. The first step develops an algorithm for one directional 2-refinement. This is followed by two directional 2-refinement and finally the culmination of a general three directional 2-refinement algorithm.

3.1 One Directional 2-Refinement

For this work we define one directional 2-refinement as refining a structured mesh in a single direction normal to a given surface through a given number of layers. Additionally in this work, two layers are used for the transition zone in which templates are inserted to make the mesh conformal. The algorithm is restricted to structured meshes only.

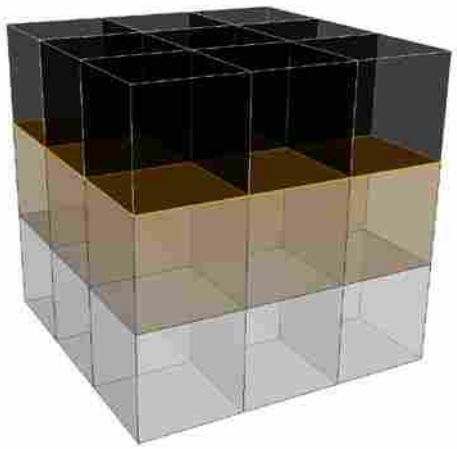
Figure 3-3 (a) shows a three by three by three mesh that is used to illustrate how the algorithm works. The topmost layer (black) is the layer selected for refinement which we will define as the Uniform Refined Zone (URZ). The number of layers in the URZ is user defined and can be any number of layers. The middle layer (tan) is the first transition zone. The bottom layer (grey) is the second transition zone. First, all of the hexes in the URZ sheets have the template from Figure 3-1 (f) inserted into them as is shown in Figure 3-3 (b). Next, the first transition layer has the template from Figure 3-1 (e) inserted into each hex is shown in Figure 3-3 (c). Finally, the second transition layer has the template from Figure 3-1(b) inserted into each hex as is shown in Figure 3-3 (d).

The method of orientating the templates for the first transition zone is described as follows. First we consider the continuous set of quadrilateral faces that are at the boundary of the URZ and the first transition zone. This is the first boundary and is seen as the faces below the shrunk hexes in Figure 3-3 (b). This boundary has a set of edges that lie in it which create a Cartesian grid. With this grid we consider only the original unrefined hexes. It can be seen that within this grid there are rows of edges that run parallel to each other with other edges running perpendicular. We select one of these rows of edges and mark it as “on” (shaded black in Figure 3-3 (b)). Then we select the next parallel row of edges and mark it as “off” (shaded tan in Figure 3-3(b)).

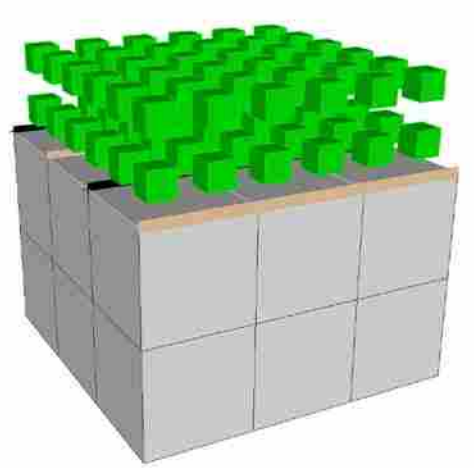
We continue this process of marking every other row of edges “on” or “off” until all parallel rows have been marked. This process is similar to the method employed by Schneiders [2] in two dimensions. This method only marks half of the rows of edges in the first boundary. The other half that runs perpendicular to these marked edges will be used in the second transition layer.

This method of marking edges “on” and “off” ensures that every hex in the first transition layer has only one edge associated with it that has been turned “on”. Once marked, the template from Figure 3-1 (e) is inserted based on its two directional nodes that form a directional edge and are shaded black. When considering a single hex the template is inserted with the directional edge aligned with the edge turned “on”. Figure 3-3 (c) shows the template from Figure 3-1 (e) being inserted into each hex properly.

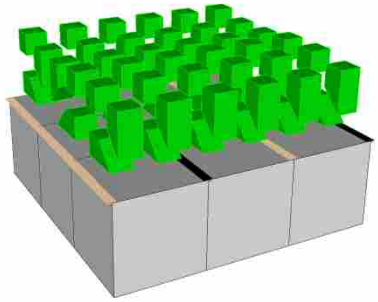
For the second transition zone we use the template from Figure 3-1 (b) using the same method as described above. The only difference is that now we consider the continuous set of faces found at the boundary of the first transition layer and the second transition layer. This is the second boundary. We use the rows of edges perpendicular to the edges used in the first boundary, but are now one level lower. We select one of these rows of edges and mark it as “on” and continue to each parallel row alternating between marking them “on” and “off”. Figure 3-3 (c) shows these rows as shaded black if marked “on” and shaded tan if marked “off”. Again, each hex in the second transition zone has only one edge turned “on”. The directional edge from Figure 3-1 (b) is matched to this “on” edge. We have now finished the refinement and template insertion and the result can be seen in Figure 3-3 (e). A flowchart of this algorithm is shown in Figure 8.



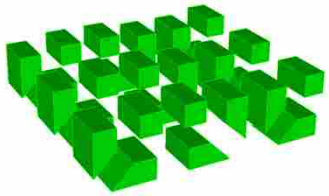
(a) 3x3x3 mesh before one directional 2-refinement.



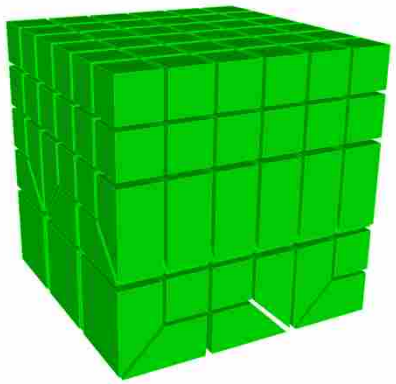
(b) Mesh after 1-to-8 template is inserted in the URZ. Rows of edges in the first boundary are marked “on” (shaded black) or “off” (shaded tan).



(c) First transition layer is templated. Rows of edges in the second boundary are marked “on” (shaded black) or “off” (shaded tan).



(d) Second transition layer is templated.



(e) Finished one directional 2 refinement with two transition zones.

Figure 3-3: 3x3x3 Mesh Example of One Directional 2-Refinement

Schneiders [2] proposed a different template that can be used for one directional 2-refinement as shown in Figure 3-4 (a). Figure 3-4 (b) shows the template proposed in this paper as applied to one column only. The average and minimum scaled jacobian values are shown as well.

(a)
Ave: 0.7039
Min: 0.4615

(b)
Ave: 0.8047
Min: 0.7071

Figure 3-4: Comparison of One Directional 2-Refinement Templates

The qualities shown would improve after smoothing the model but as shown the template proposed in this thesis has significantly higher quality. It also takes two transition layers which helps transition more smoothly from areas of higher to lower mesh density.

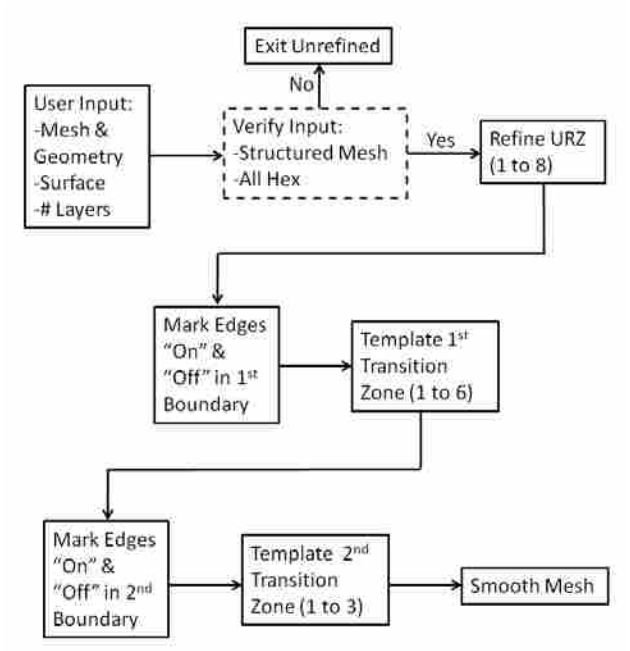


Figure 3-5: Flowchart of One Directional 2-Refinement Algorithm

3.2 Two Directional 2-Refinement

We now define two directional 2-refinement as refining adjacent columns of a structured mesh in two orthogonal directions normal to two defining surfaces. Two layers are also selected beyond the specified columns for the transition zones, in which templates will be inserted to make the mesh conformal. This algorithm is also restricted to structured meshes only.

Figure 3-6 (a) shows a three by three by three hexahedral mesh used to illustrate how the algorithm works. The column shaded black in this example is the only column to be refined and is the URZ. The hexes shaded brown are the first transition zone and hexes shaded light grey are the second transition zone. First, all of the hexes in the URZ have the template from Figure 3-1 (f) inserted into them as is shown in Figure 3-6 (b). Next, the hexes in the first transition zone that share a face with the URZ have the template from Figure 3-1 (e) inserted as is shown in Figure 3-6 (c). This leaves unrefined "corner" hexes in the first transition zone. These corner hexes share only a single edge with the URZ and have the template from Figure 3-1 (c) inserted into them as is shown in Figure 3-6 (d).

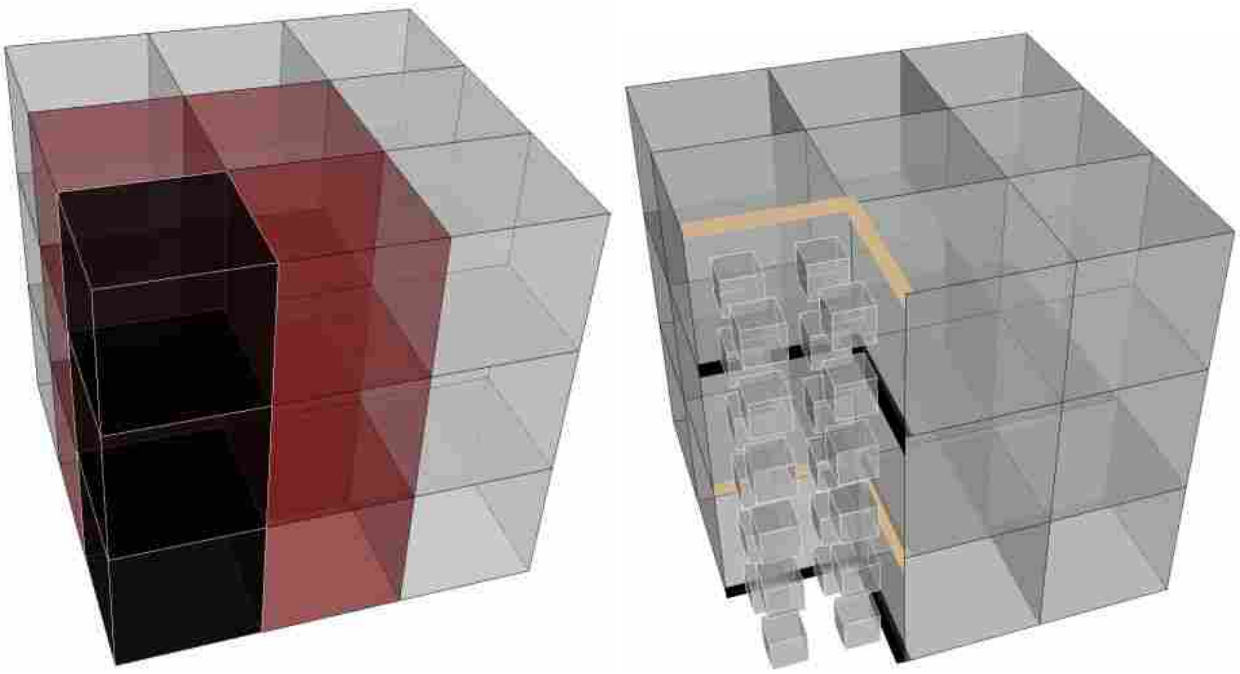
The method for determining the correct orientation for insertion of the first transition zone hexes is described as follows. Again we consider that at the interface between the URZ and the first transition zone there is a set of continuous quadrilateral faces found. This is the first boundary; however, it is no longer two-dimensional. It now wraps around the URZ and connects back on itself. This boundary does not include the top and bottom surfaces of the columns to be refined. In the first boundary we observe again rows of orthogonal edges. One set of these edges wrap around the first boundary (i.e. Figure 3-6 (b)) to connect back onto themselves. The other set runs longitudinally along the columns (i.e. Figure 3-6 (d)). We first consider the rows of edges that wrap around the first boundary. In the example of Figure 3-6, the first boundary

does wrap around but the sides where there are no hexes will be ignored. In Figure 3-6 (b) every other row is turned “on” or “off”. The edges shaded black are turned “on” and the edges shaded tan are turned “off”. As before, individual template insertion is based on matching the directional edge associated with the template to the edge that is turned “on” and the result is seen in Figure 3-6 (c). The corner hexes are treated differently as the template from Figure 3-1 (c) does not have a directional edge, but rather a directional node. Figure 3-6 (c) shows the nodes shaded black as turned “on” and shaded tan as turned “off”. It can be seen that these nodes are turned “on” or “off” based on the edge with which they are connected. Figure 3-6 (d) shows the corner hexes correctly templated.

At this point the entire first transition zone is correctly templated and the second transition zone remains. Now we consider the second boundary, which is the set of faces that lies between the first transition zone and the second transition zone. This is the set of exposed faces shown in Figure 3-6 (d). In the first boundary we used the rows of edges that wrapped around onto themselves to orient the templates correctly. Now we consider the rows of edges that run longitudinally along the second boundary. Figure 3-6 (d) shows these shaded and again we mark every other row “on” or “off”.

However, if we insert the template from Figure 3-1 (b) onto the edges that are currently turned “on”, we create hanging nodes. This problem is created because of the corners. To solve this problem we take all of the rows of edges that lie in the corners of the second boundary and we invert them “on” or “off”. In the example of Figure 3-6 this means we turn the row of edges in the corner of the second boundary “on” as seen in Figure 3-6 (e). This process now creates hexes that have two edges that are turned “on”.

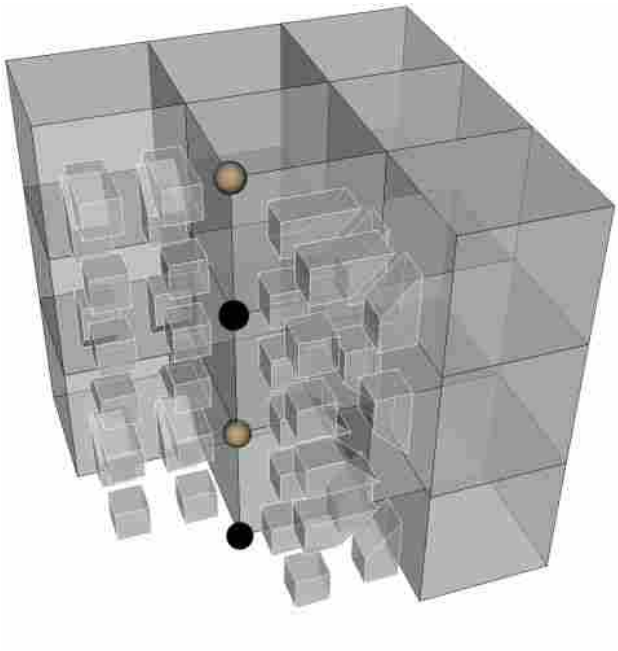
The template used on these hexes is from Figure 3-1 (a) except that we do not use the directional edge shaded in the figure. The template from Figure 3-1 (a) is oriented with the un-split face connecting to both edges turned "on". The result can be seen in Figure 3-6 (e). The rest of the hexes in the second transition zone are templated normally with the template from Figure 3-1 (b) and can be seen in Figure 3-6 (e). We have now finished templating the second transition zone and the final result of the two directional 2-refinement algorithm can be seen in Figure 3-6 (f). A flowchart of this algorithm is shown in Figure 3-9.



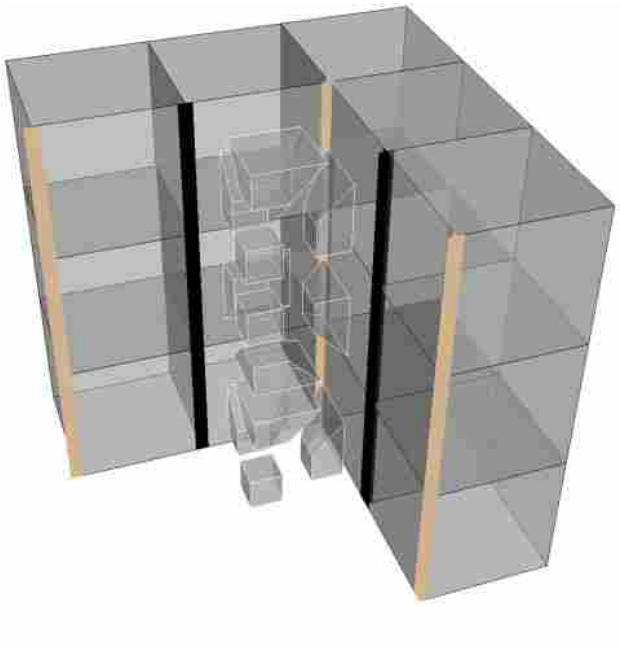
(a) 3x3x3 mesh before two directional 2-refinement.

(b) The URZ is refined. Rows of edges in the first boundary are marked "on" (shaded black) or "off" (shaded tan).

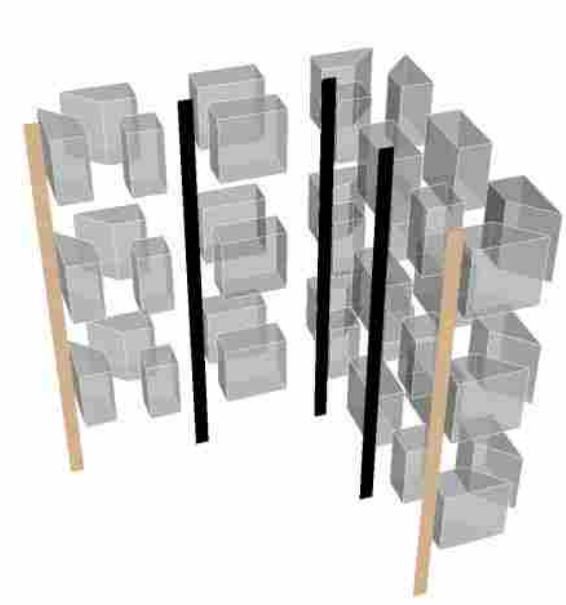
Figure 3-6: 3x3x3 Mesh Example of Two Directional 2-Refinement



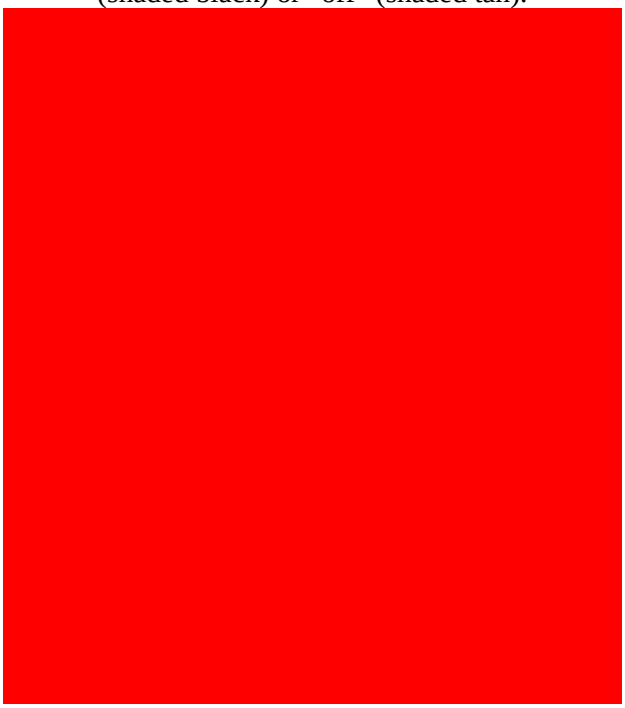
(c) 1 to 6 template is inserted. Nodes are turned “on” (shaded black) or “off” (shaded tan).



(d) First transition zone is complete. Rows of edges in second boundary are marked “on” (shaded black) or “off” (shaded tan).



(e) Edges in corner of second boundary are inverted to “on”. Second transition zone is completely templated.



(f) Two directional 2-refinement complete.

Figure 3-6 (continued): 3x3x3 Mesh Example of Two Directional 2-Refinement

Two directional refinement introduces possible concave regions with transition elements that share multiple faces with refined elements (multiply connected transition elements [4]). The only concave case considered in two directional 2-refinement is concave regions created by a doubly connected transition element whose two refined faces are adjacent to each other, or in other words two-concave regions as shown in Figure 3-7. The multiply connected transition element is shaded grey. We also require that the concave zone be at least two hexes by two hexes as shown in the figure. All other possible concave regions are simply refined using the template in Figure 3-1 (f).

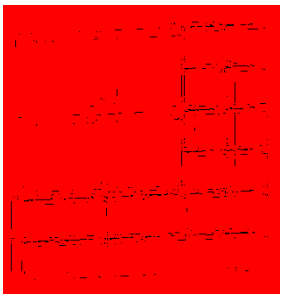
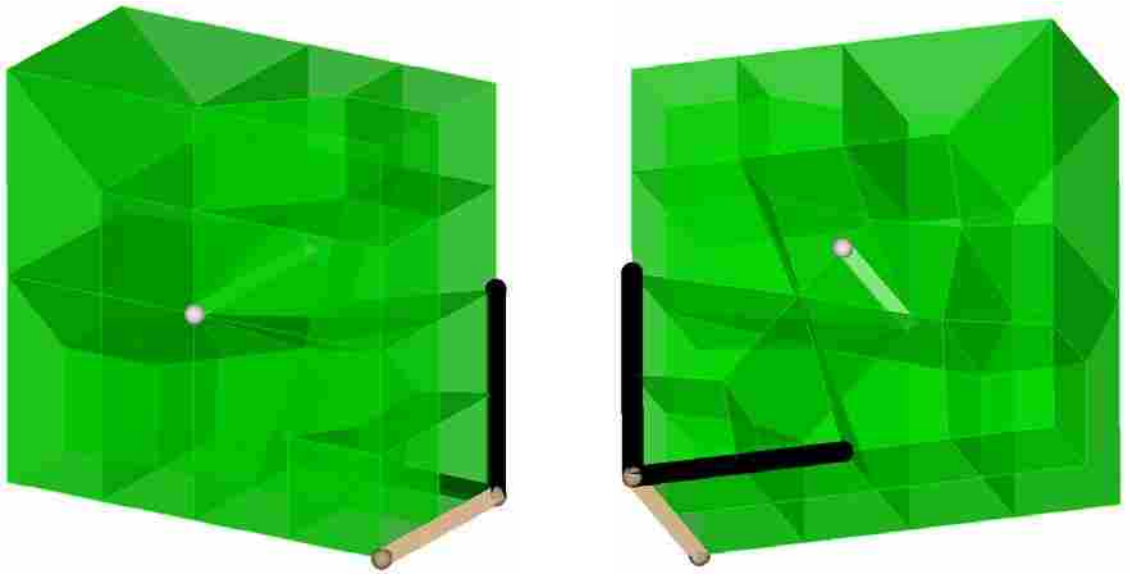


Figure 3-7: Two Concave Zone

None of the templates in Figure 3-1 can be used to mesh a two-concave region, except by completely refining this region. Thus the template shown in Figure 3-8 was created and is required to deal with concave regions. The creation of this template is provided in Appendix B. This new template uses four total hexes in a two by two by one arrangement. This template also has three directional edges. The first and second are associated with the edges turned "on" and "off" in the first boundary (shaded black in Figure 3-8). The third is the concave directional edge and is associated with the edge found at the concave region (shaded tan in Figure 3-8). There is also a central edge shaded white in Figure 3-8 that is addressed in Appendix C. A two-concave example using this template can be found in Appendix C.



(a) Front (b) Back

Figure 3-8: Two-Concave Template

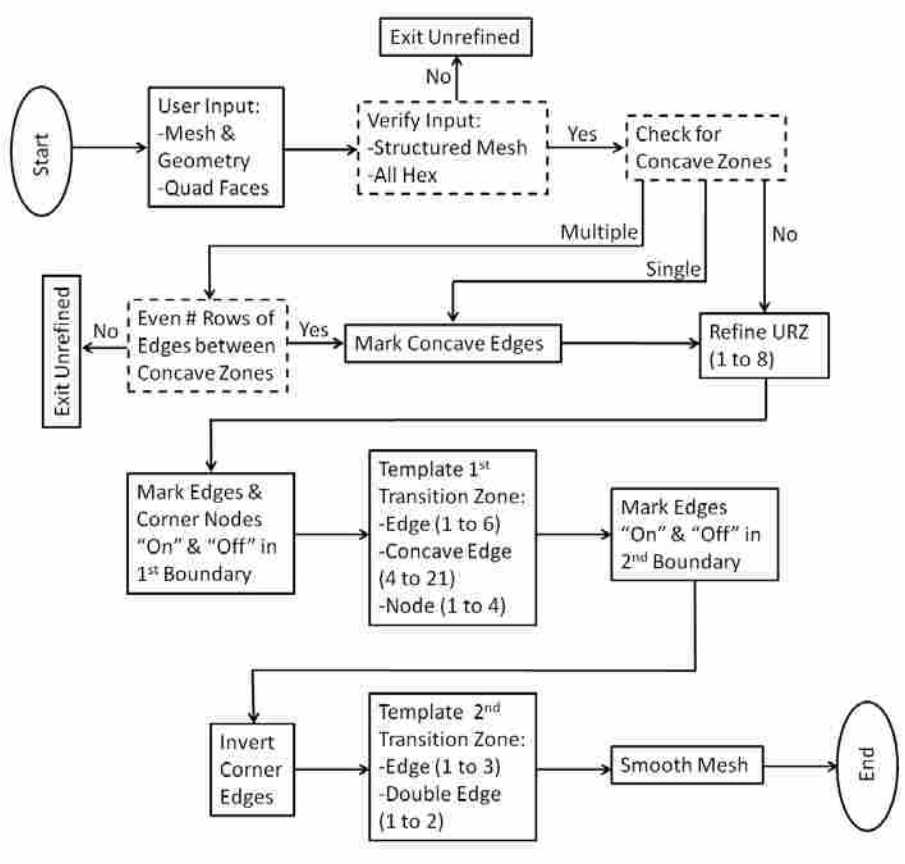


Figure 3-9: Flowchart of Two Directional 2-Refinement Algorithm

3.3 Three Directional 2-Refinement

We now define three directional 2-refinement as refining adjacent hexes of a structured mesh in three orthogonal directions normal to three defining surfaces. As shown in Figure 3 pillowing can be used for 2-refinement. Also notice that in order to pillow 2-refinement the pillow must enclose two original transition elements. Because of this quality we require refined hexes to come in pairs in the X, Y and Z directions. Marechal [7] calls this the pairing rule. Marechal makes this requirement as well but he did not utilize the fact that the pairing rule can be neglected if refined hexes lie on a geometric boundary as was the case with the one directional and two directional 2-refinement algorithms discussed earlier. As was also discussed in Figure 3, a pillow must be inserted in all three X, Y and Z directions. This forms the basis for the introduction of marking edges with priority one, two or three rather than simply "on" as was done in the previous algorithms.

Figure 3-10 (a) shows a four by four by four mesh used to illustrate how the algorithm works. In this figure the URZ is a block of two by two by two hexes and represents the smallest refine-able area given the pairing rule. These hexes are already refined using Figure 3-1 (f) and are shown as shrunken hexahedrals. The brown hexes are the first transition zone and light grey hexes are the second transition zone.

Again we consider the continuous set of faces that lie in between the URZ and the first transition zone which is the first boundary. This is the set of exposed faces in Figure 3-10 (b) that are shown after the refined hexes are removed from view. The first boundary is now a set of faces that enclose the URZ completely. In this example we will ignore the parts of the first boundary that do not have adjacent hexes. Within the first boundary again we have rows of edges that run orthogonal to each other. Because of how these edges now interact with each

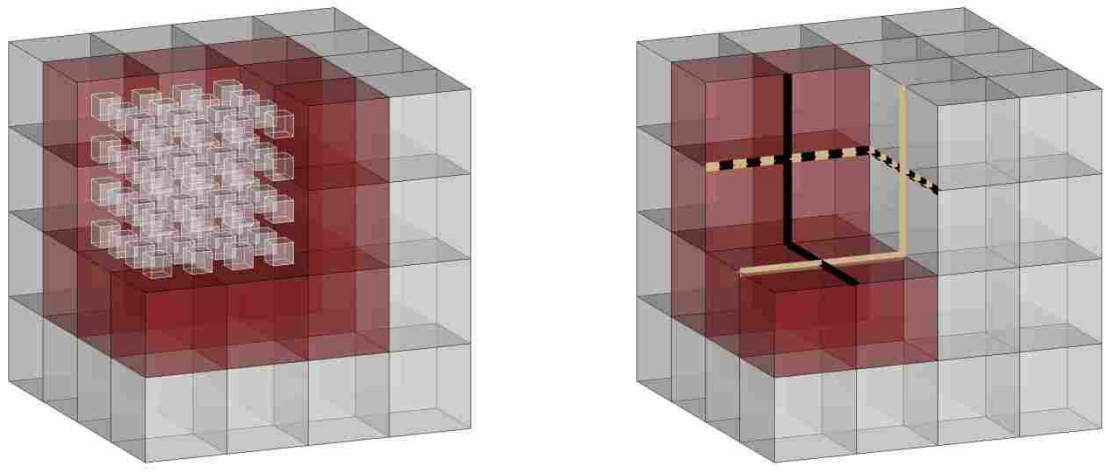
other by wrapping completely around we cannot simply turn them “on” but instead each parallel row of edges will be assigned a priority. These rows of edges lie in either the x-y plane, y-z plane or x-z plane. For this example rows of edges that lie in the y-z plane are given priority one (shaded black), rows of edges that lie in the x-z plane are given priority two (shaded dashed), rows of edges that lie in the x-y plane are given priority three (shaded tan). Only one set of edges are marked in each plane but it can be seen that each plane has a total of three sets. This is because, as in the previous algorithms, we only mark every other row of edges with a priority or “off”, ensuring that we start at the corner edges as “off”. This pattern leaves the rows marked as seen in Figure 3-10 (b).

First we consider the hexes that connect to the priority one edges which have been shaded brown in Figure 3-10 (b). We insert the template in Figure 3-1 (e) as usual with its directional edge aligning with the edge marked priority one. We also set the corner node as priority one for orienting the template from Figure 3-1 (c) with its directional node. This finishes all hexes attached to priority one edges and nodes as seen in Figure 3-10 (c). Notice that only first transition zone hexes were used for priority one templating.

Next we consider the hexes that are connected to priority two edges. Some of these hexes have already been used to template priority one. To resolve this we now use first and second transition zone hexes and to do this we re-mark edges priority two as shown in Figure 3-10 (c). The hexes to be used to template priority two have been shaded brown in this figure as well. To template hexes with edges marked priority two we will use the templates found in Figure 3-1 (b) and (e). All hexes that share a face with the URZ and have an edge marked priority two have the template from Figure 3-1 (e) inserted. All other hexes with an edge marked priority two will have the template from Figure 3-1 (b) inserted. In both cases the edge marked priority two will

be used to align the directional edge of the template when inserted. We also set the corner node as priority two for orienting the template from Figure 3-1 (c) with its directional node. Figure 3-10 (d) shows the result for priority two.

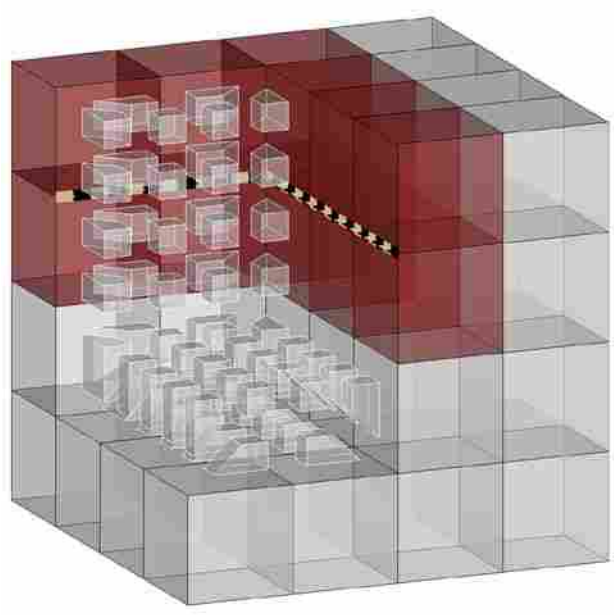
Finally we consider the hexes that are connected to priority three edges but most of them have been used for priority one and two. We then re-mark edges priority three as shown in Figure 3-10 (d) and use the hexes shaded brown in the figure. Hexes with exactly one edge marked priority three use the template from Figure 3-1 (b). Hexes with more than one edge marked priority three use the template from Figure 3-1 (a). These hexes are found in the first transition zone and the edge used to align the directional edge of the template is the edge shared in common with the URZ. Corner nodes are again marked priority three and used to align the directional node of the template in Figure 3-1 (c). All hexes used in priority three are now finished and the result is seen in Figure 3-10 (e). As illustrated, some of the hexes in the first and second transition zones require no refining and are thus left alone. Figure 3-10 (f) shows the final result.



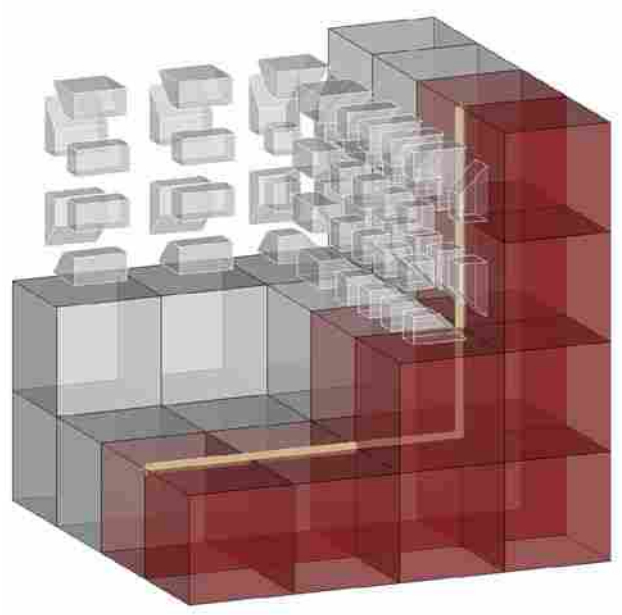
(a) 4x4x4 mesh before three directional 2-refinement. URZ zone is refined.

(b) The first boundary is exposed here with all rows of edges marked priority one, two or three.

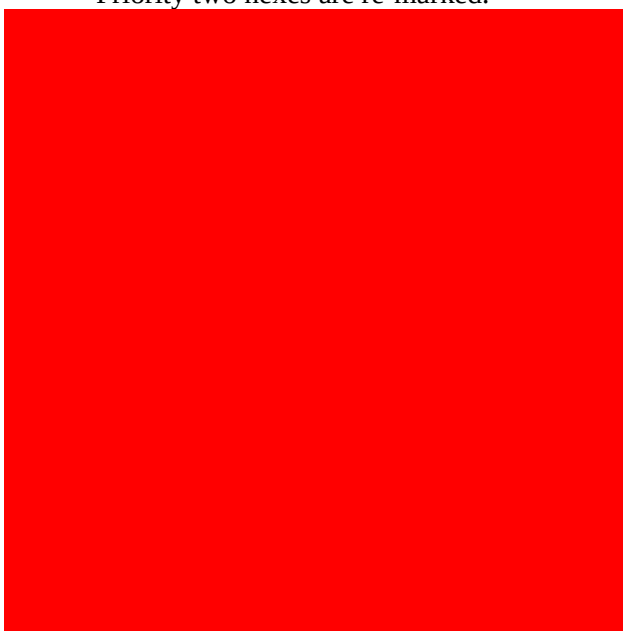
Figure 3-10: 4x4x4 Example of Three Directional 2-Refinement



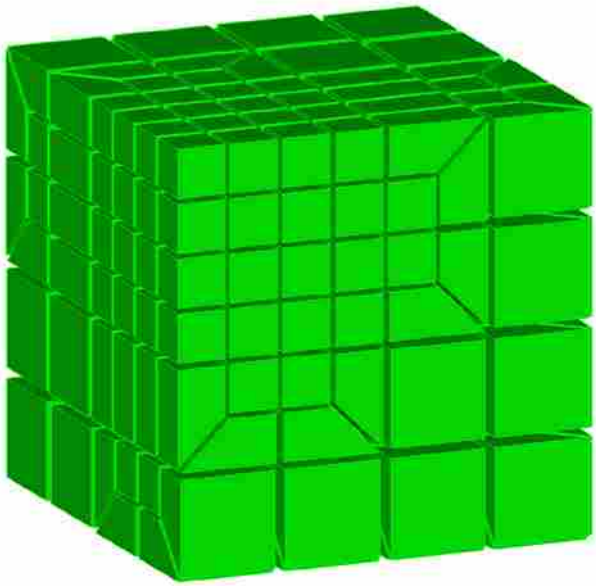
(c) Priority one hexes are templated. Priority two hexes are re-marked.



(d) Priority two hexes are templated. Priority three hexes are re-marked.



(e) Priority three hexes are templated. The remaining hexes are unchanged.



(f) Three directional 2-refinement complete

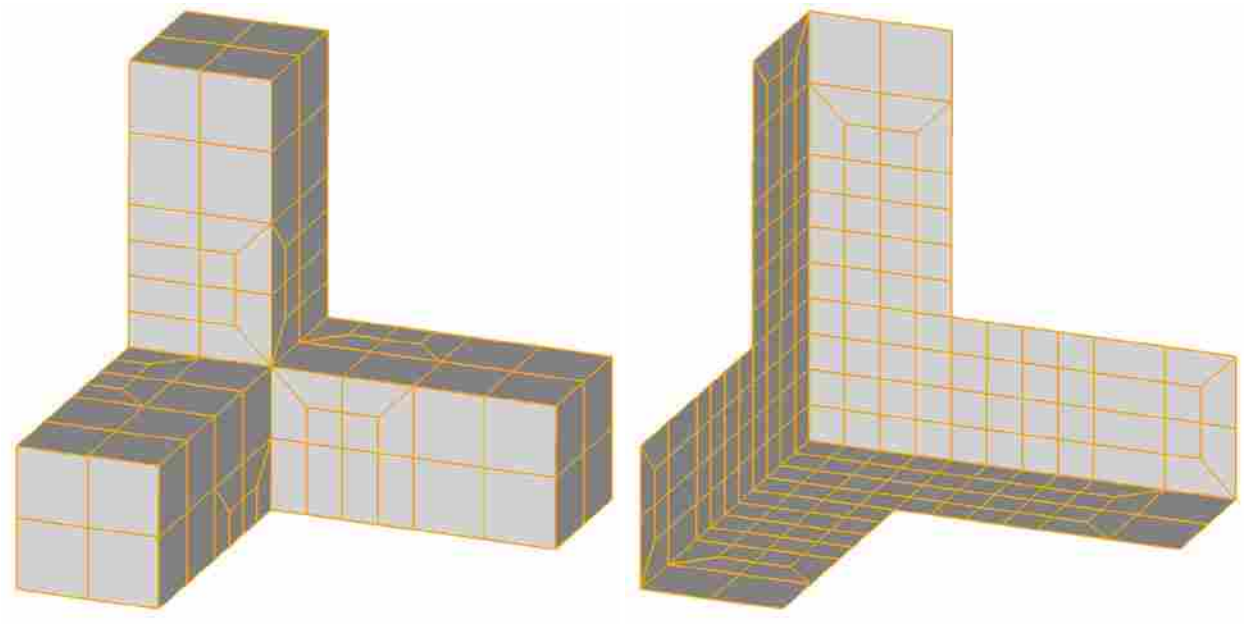
Figure 3-10 (continued): 4x4x4 Example of Three Directional 2-Refinement

Three directional refinement introduces possible concave regions with transition elements that share multiple faces with refined elements. There are two cases of concavity that we consider here. The first is the two-concave case, which was considered previously in two

directional 2-refinement. The second is when a transition element has three faces being refined, where all three faces share the same node, or in other words a three-concave region. Any three-concave regions are required to be at least two by two by two hex regions. All other possible concave regions are simply refined using the template in Figure 3-1 (f).

None of the templates introduced thus far can be used to template a three-concave region, except by completely refining the region. It requires the use of the template in Figure 3-11. The creation of this template is provided in Appendix D. A three-concave region is inherently created by three separate two-concave regions all orthogonal to each other that connect at a single point. This means that we not only need a template that uses three of its faces to handle a three-concave region but we also need it to use its remaining three faces to connect to templates that handle two-concave regions. That is the reason for the shape of this new template. It is in fact five templates in one. The three separate two-concave regions that jut out from the three-concave area are referred to as “legs”. The back side is shown as if the front side were removed from view.

The figures following Figure 3-11 show each part of this one template being separated. Each piece is a two by two by two template. The three pieces at the end of each leg of the template in Figure 3-11 are identical and are the capstones shown in Figure 3-12. They are applied beyond the refined zone each leg fits in. Thus the front side of this template shows completely unrefined faces.



(a) Front (b) Back

Figure 3-11: Three-concave template

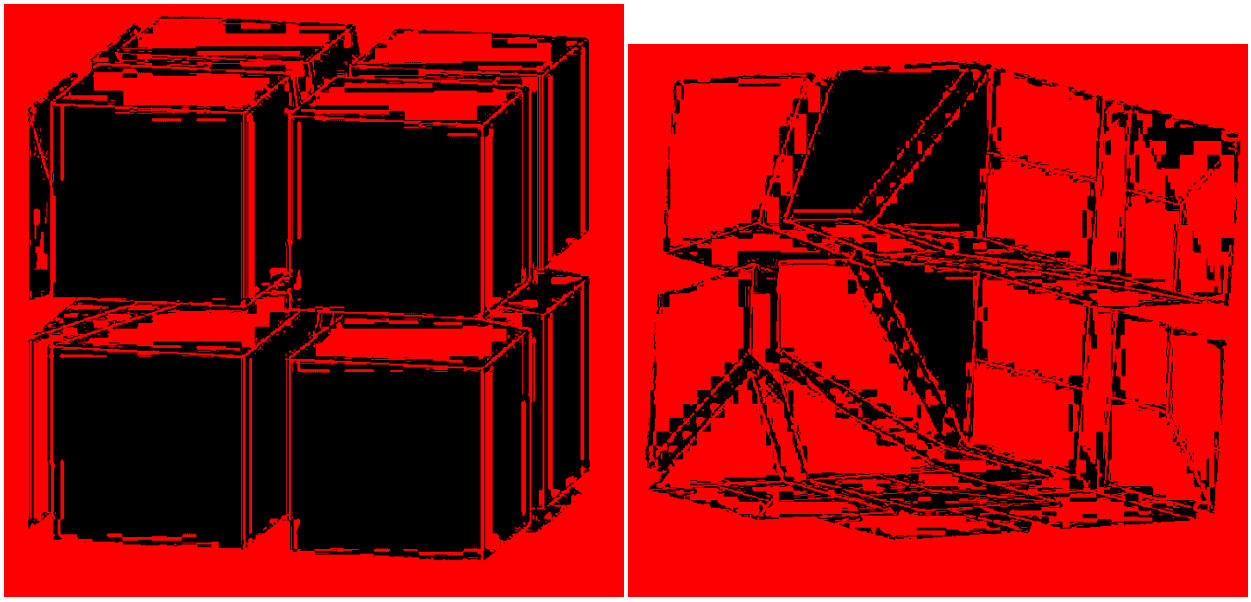


Figure 3-12: Capstone Template Front and Back Sides

Figure 3-13 is the bottom left leg in Figure 3-11 and is called two-concave leg one. Figure 3-14 is the bottom right leg in Figure 3-11 and is called 2-concave leg two. Figure 3-15 is the top leg in Figure 3-11 and is called two-concave leg three. The lengths of the two-concave

regions these templates fit into will differ from mesh to mesh, depending on the geometry and the refinement requirements of the model. Thus these templates may be used multiple times to fit the lengths of each two-concave region in any given three-concave situation.

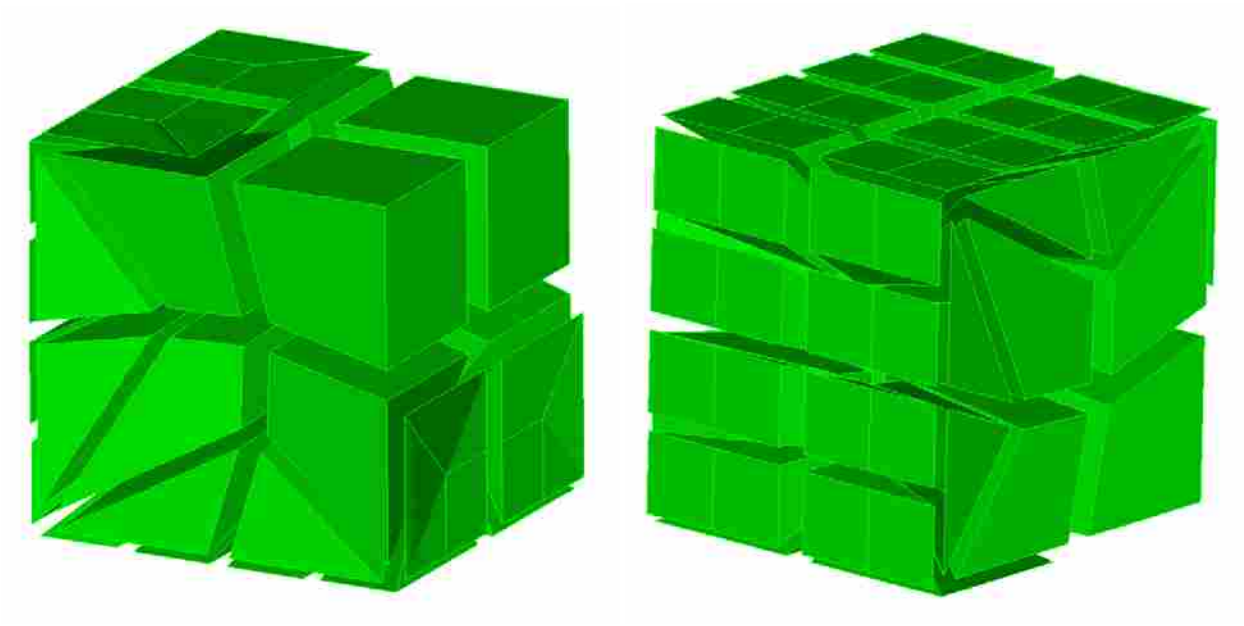


Figure 3-13: Two-Concave Leg One Template Front and Back Sides

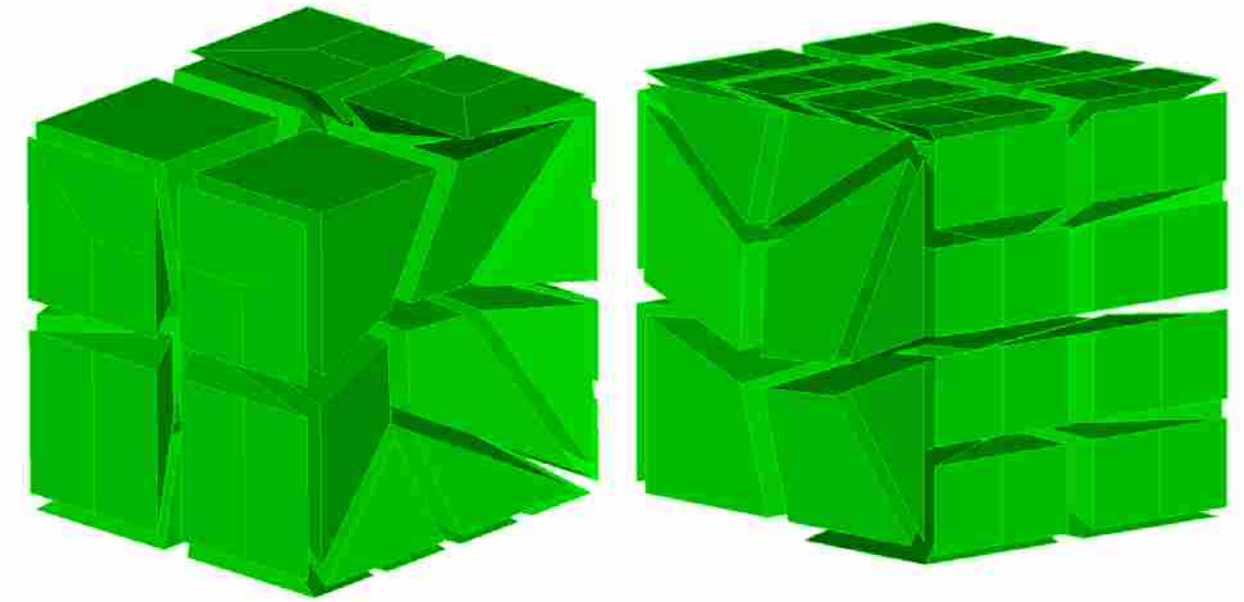


Figure 3-14: Two-Concave Leg Two Template Front and Back Sides

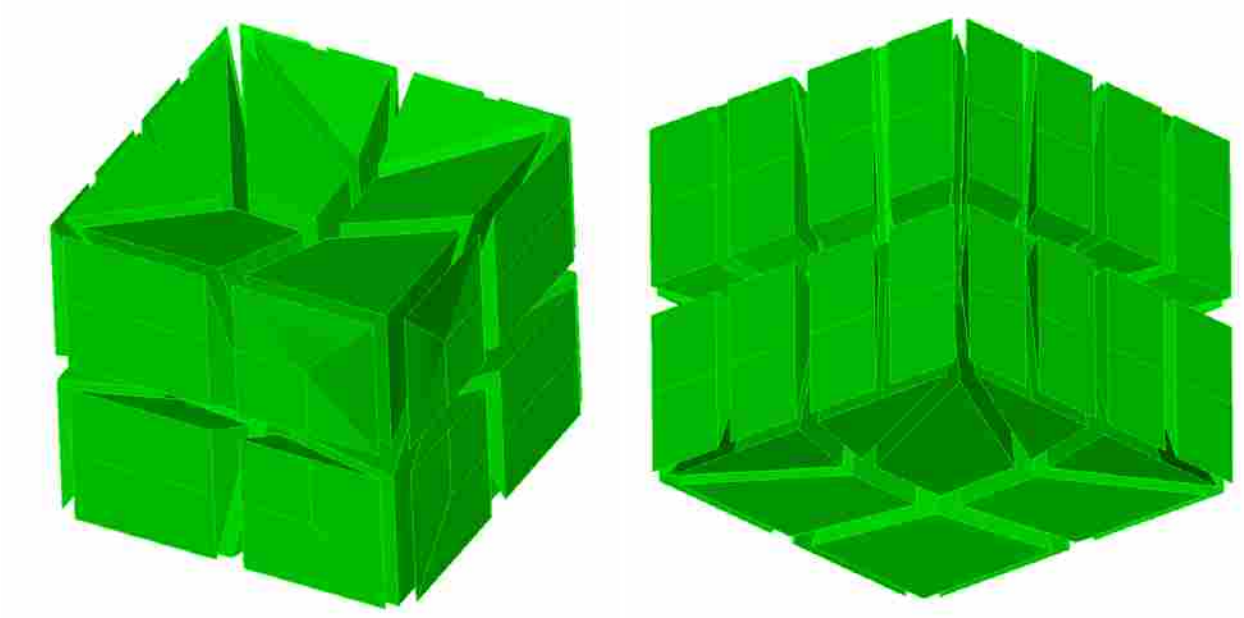


Figure 3-15: Two-Concave Leg Three Template Front and Back Sides

Figure 3-16 is the central template that fills the three-concave region. The back side is completely refined and is the side that connects to the URZ. The front three sides are those that connect directly to each two-concave leg.

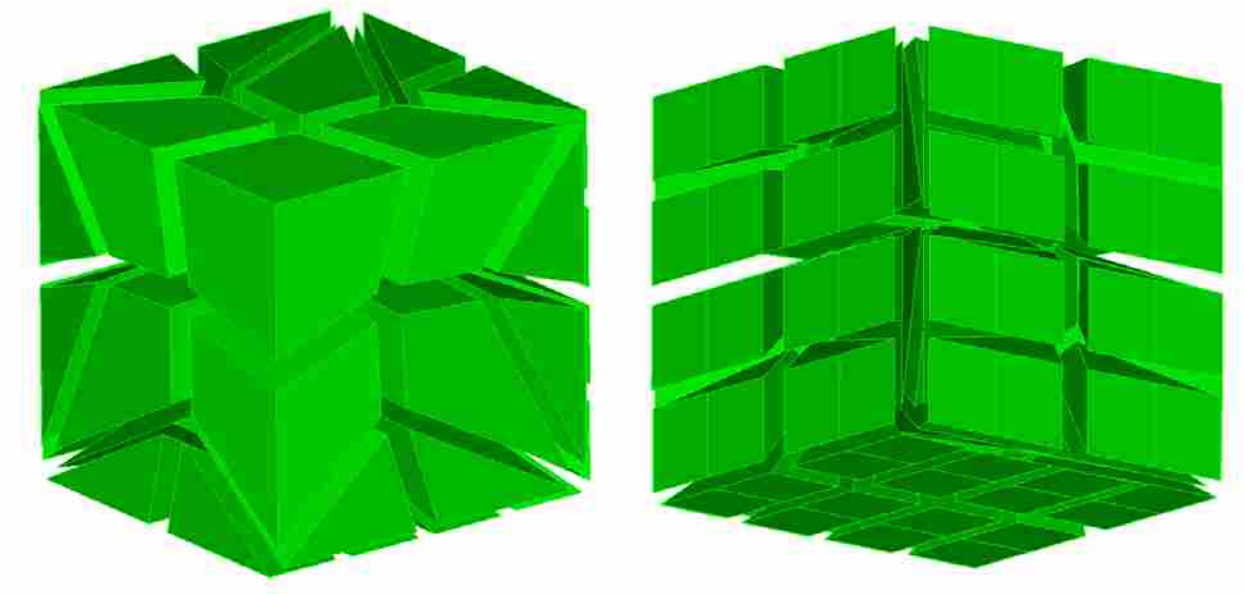


Figure 3-16: 3-Concave Center Template Front and Back Sides

It can be seen that all of these templates from Figure 3-11 have hexes with faces being forced into the same plane. Thus the element quality is extremely low upon initial insertion. To improve element quality smoothing must be performed after the refinement is completed. In this case, all of the hexes with faces on the same plane are connected to regular shaped hexes, thus smoothing is available. An example of a three-concave solution and the resulting element quality after smoothing is found in Appendix E. A flowchart showing the proposed algorithm is shown in Figure 3-17.

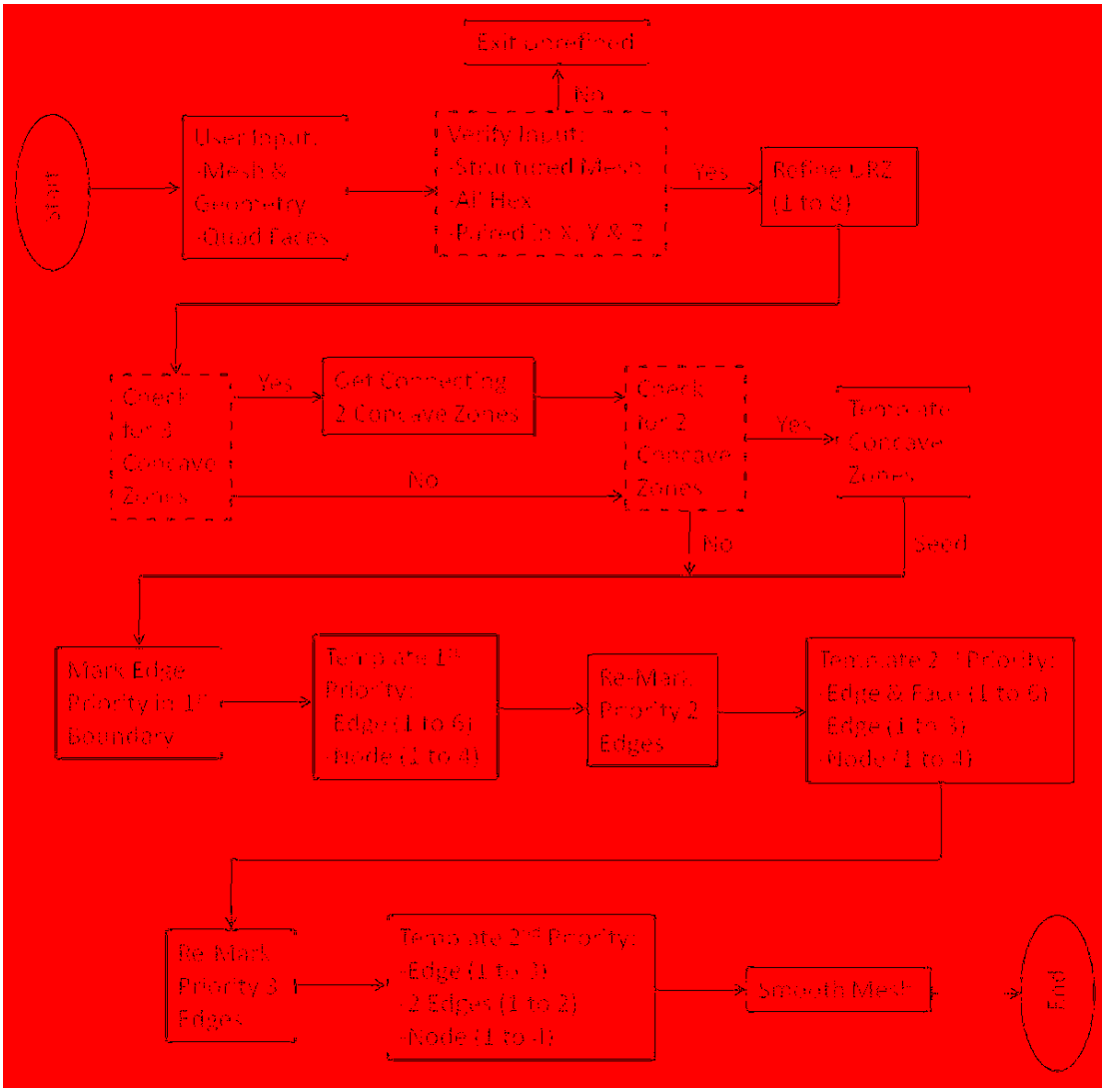
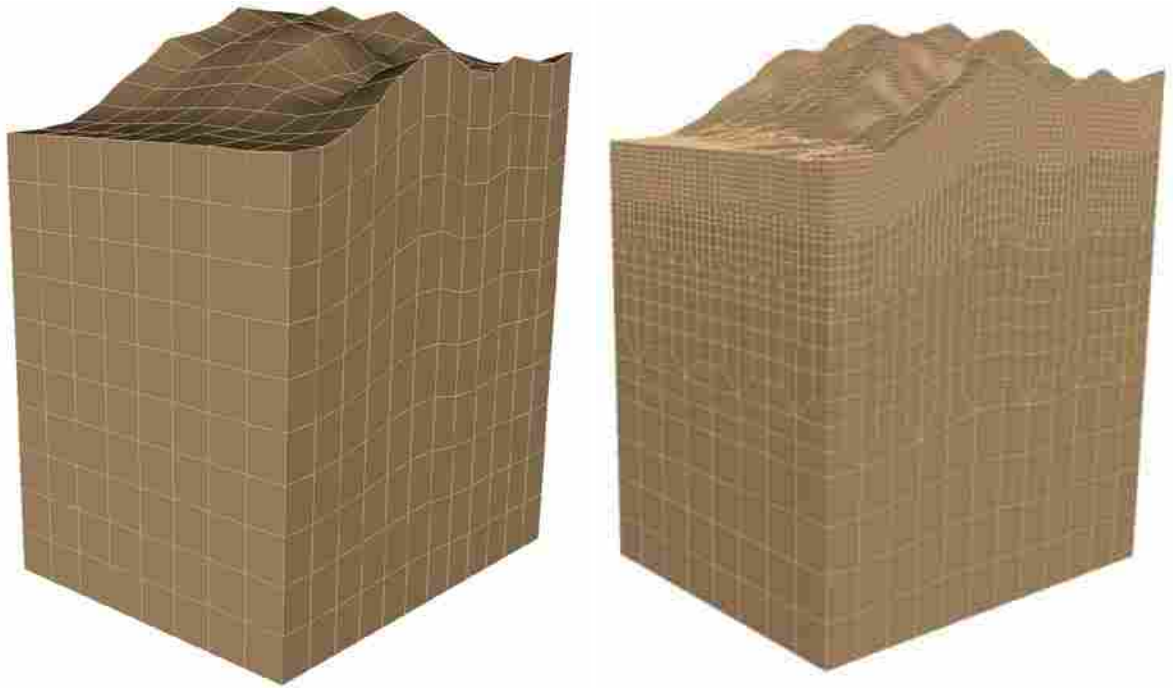


Figure 3-17: Flowchart of Three Directional 2-Refinement Algorithm

4 EXAMPLES

4.1 One Directional Example

The example here is a geology model of the earth beneath Y-mount in Provo, UT with surface data from ARCGIS. The model is 3.03 miles by 2.46 miles by 2.65 miles. The mesh before refinement is shown in Figure 4-1 (a). The model was refined to provide higher resolution at the surface for analysis which is common in geologic modeling. The result can be seen in Figure 4-1 (b). The element quality results are shown in Table 4-1. The model was refined three times and the iteration column in Table 4-1 refers to each refinement.



(a) Before refinement. (b) After refinement.

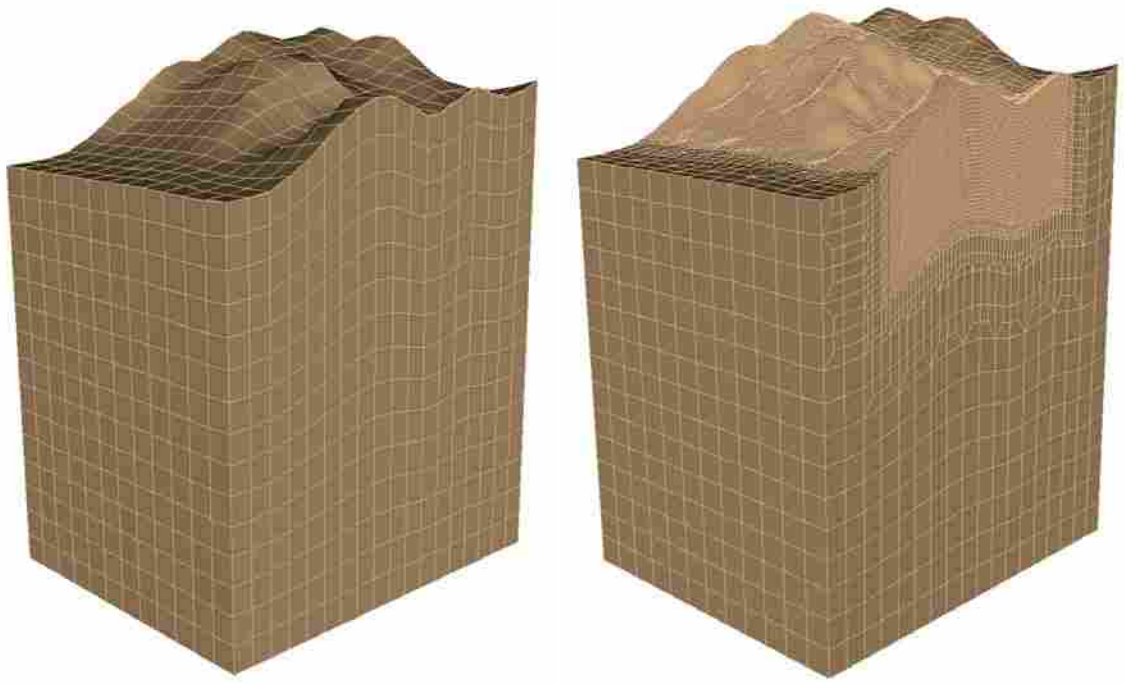
Figure 4-1: Y-Mount Model Illustrating One Directional 2-Refinement

Table 4-1: Results of One Directional 2-Refinement Example

Iteration	# Elements	Ave	Std Dev	Min	Max
0	960	0.9705	4.95E-02	0.4863	1
1	4,320	0.9233	9.77E-02	0.3796	0.9998
2	17,760	0.9124	1.04E-01	0.3552	0.9998
3	71,520	0.9022	1.11E-01	0.2846	0.9998

4.2 Two Directional Example

The geologic Y mount model is used again for two directional 2-refinement. The mesh before refinement is shown in Figure 4-2 (a). The model was refined to capture additional features in the center of the model and the result can be seen in Figure 4-2 (b). The element quality results are shown in Table 4-2. The model was refined three times and the iteration column in Table 4-2 refers to each refinement.



(a) Before refinement. (b) After refinement.

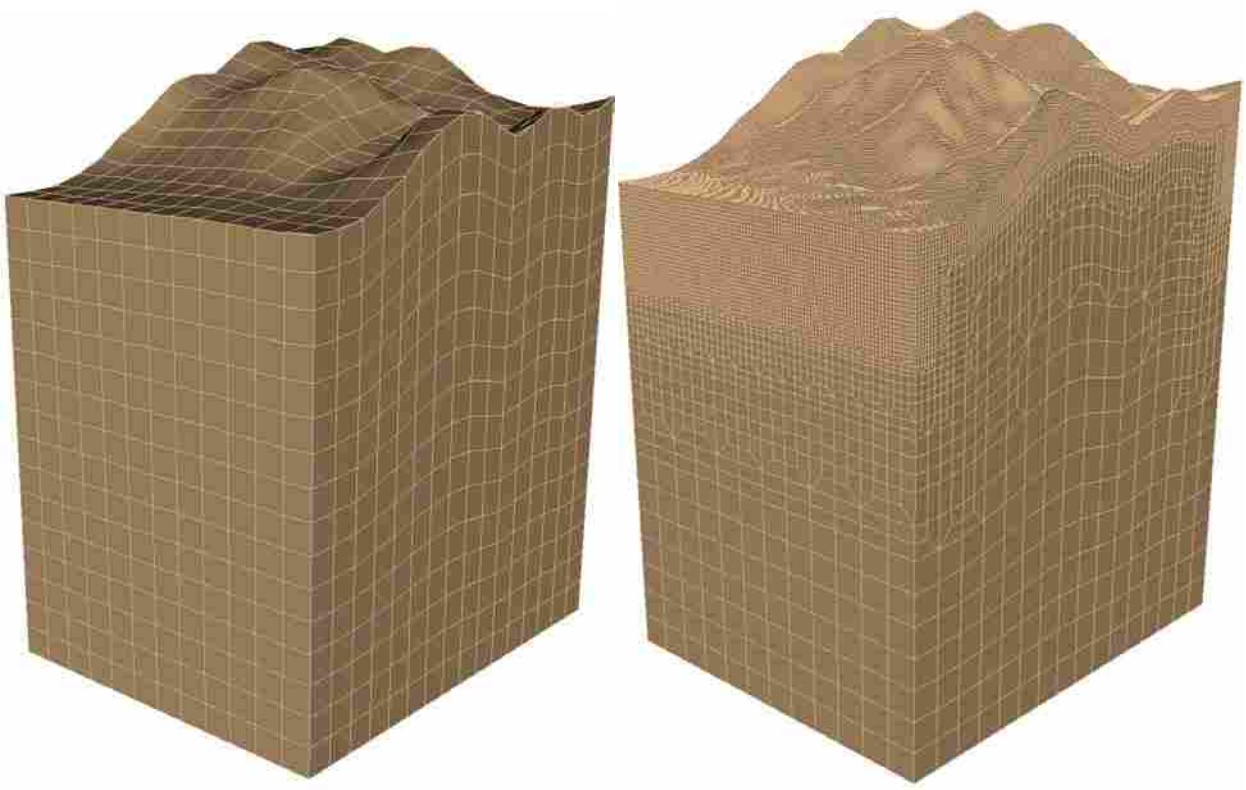
Figure 4-2: Y-Mount Model Illustrating Two Directional 2-Refinement

Table 4-2: Results of Two Directional 2-Refinement Example

Iteration	# Elements	Ave	Std Dev	Min	Max
0	3,240	0.9733	4.76E-02	0.5146	1
1	11,208	0.9105	1.21E-01	0.4041	.9998
2	50,520	0.8978	1.36E-01	0.3462	.9978
3	282,360	0.9104	1.14E-01	0.2686	.9978

4.3 Two Concave Example

The geologic Y mount model is used again for a two concave example. The mesh before refinement is shown in Figure 4-3 (a). The model was refined to capture additional features at the top and sides of the model and the result can be seen in Figure 4-3 (b). The element quality results are shown in Table 4-3. The iteration column in Table 3 refers to each refinement.



(a) Before refinement. (b) After refinement.

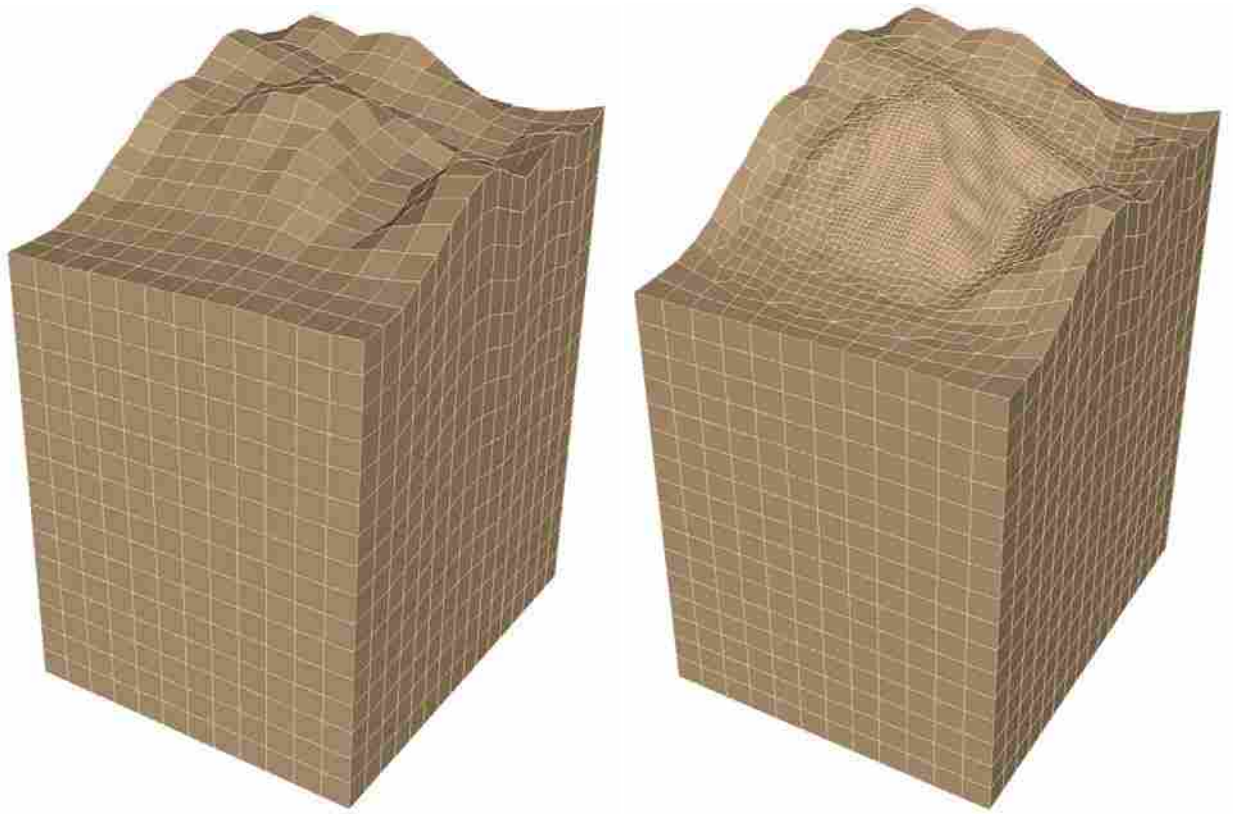
Figure 4-3: Y-Mount Model Illustrating Two Concave Example

Table 4-3: Results of Two Concave Example

Iteration	# Elements	Ave	Std Dev	Min	Max
0	3,240	0.9733	4.76E-02	0.5146	1
1	11,208	0.9105	1.21E-01	0.4041	.9998
2	50,520	0.8978	1.36E-01	0.3462	.9978
3	282,360	0.9104	1.14E-01	0.2686	.9978

4.4 Three Directional Example

The geologic Y mount model is used again for a three directional 2-refinement example. The mesh before refinement is shown in Figure 4-4 (a). The model was refined to capture additional features of the central peak in the model and the result can be seen in Figure 4-4 (b). The element quality results are shown in Table 4-4. Figure 4-5 is a break away view. The model was refined three times and the iteration column in Table 4-4 refers to each refinement.



(a) Before refinement. (b) After refinement.

Figure 4-4: Y-Mount Model Illustrating Three Directional 2-Refinement

Table 4-4: Results of Three Directional 2-Refinement Example

Iteration	# Elements	Ave	Std Dev	Min	Max
0	3,072	0.9597	5.47E-02	0.4626	1.0000
1	7,900	0.9035	1.24E-01	0.4200	1.0000
2	14,456	0.8640	1.25E-01	0.3717	1.0000
3	21,156	0.8585	1.31E-01	0.3049	1.0000

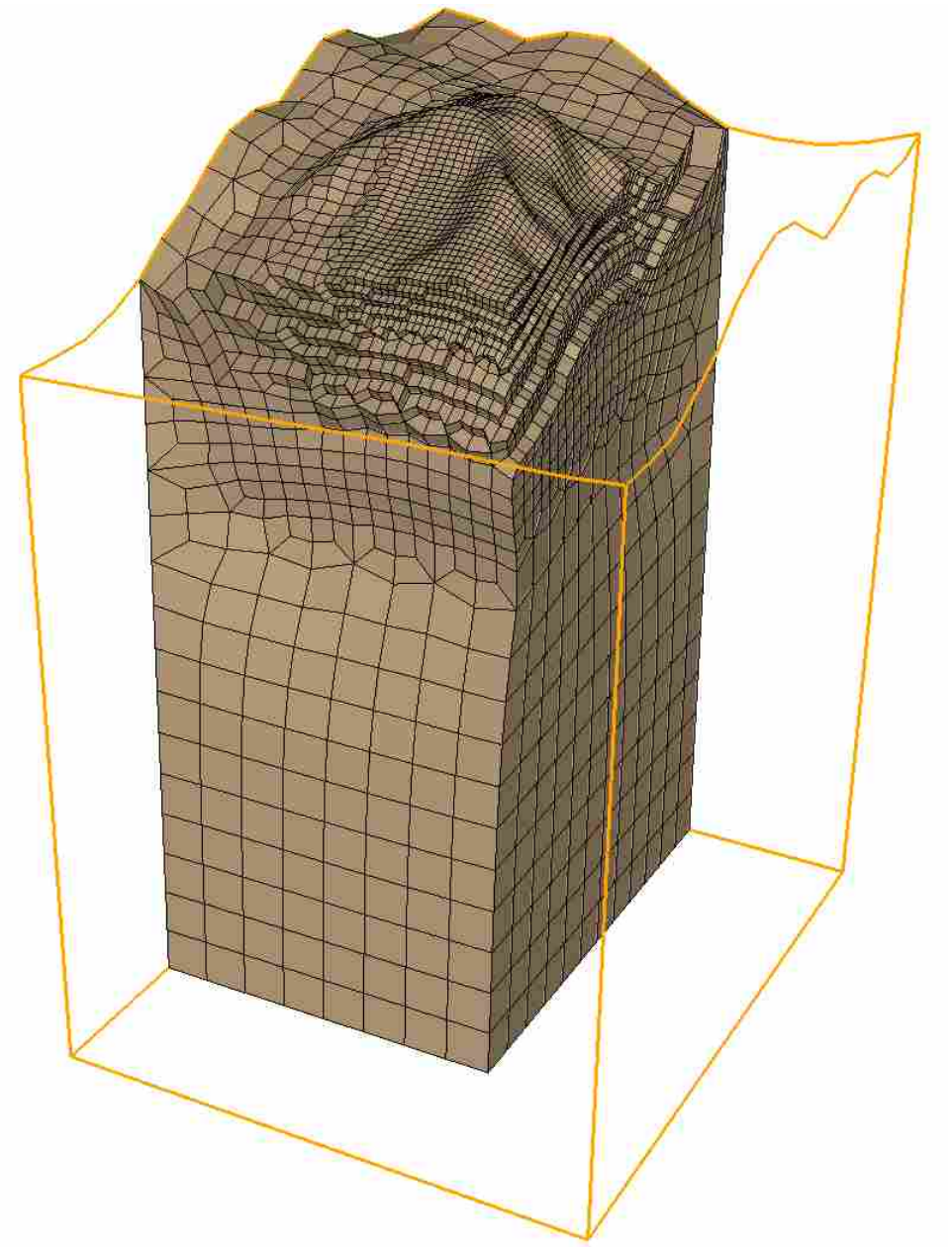


Figure 4-5: Break Away View of Three Directional Y-mount Example

5 CONCLUSION

An Adaptive all-hexahedral meshing algorithm has been presented here. This algorithm is grid-based and employs a 2-refinement insertion method. 2-refinement is based on dividing a hex to be refined into eight new hexes. This provides a mesh that is efficient for analysis by providing a high element density in specific locations, such as areas of high stress gradient or high curvature and reduced mesh density in other areas of less importance. This algorithm used a two layer transition zone to increase element quality and keep transitions from lower to higher mesh densities smooth. Templates were introduced to allow both convex and concave refinement.

REFERENCES

1. Ito Y, Shih A, Soni B. Octree-based reasonable-quality hexahedral mesh generation using a new set of refinement templates. *International Journal for Numerical Methods in Engineering* 2009; 77:1809-1833. DOI: 10.1002/ nme.2470

2. Schneiders R. Refining quadrilateral and hexahedral element meshes. *Proceedings of the 5th International Conference,* Mississippi State University, 1996; 679-688.

3. Harris N, Benzley S, Owen S. Conformal refinement of all-hexahedral element meshes based on multiple twist plane insertion. *Proceedings of the 13th International Meshing Roundtable,* Sandia National Laboratories, Williamsburg, VA, 2004; 157-168. SAND #2004-3765C.

4. Parrish M, Borden M, Staten M, Benzley S. A selective approach to conformal refinement of unstructured hexahedral finite element meshes. *Proceedings of the 16th International Meshing Roundtable,* Seattle, WA, 2007; 251-268. DOI: 10.1007/978-3-540-75103-8_15.

5. Schneiders R, Schindler R, Weiler F. Octree-based generation of hexahedral element meshes, *Proceedings of the 5th International Meshing Roundtable,* Sandia National Laboratories, 1996; 205-216.

6. Zhang Y, Bajaj C. Adaptive and quality quadrilateral/hexahedral meshing from volumetric data. *Computer Methods in Applied Mechanics and Engineering* 2006; 195:942-960. DOI: 10.1016/j.cma.2005.02.016.

7. Maréchal L. Advances in octree-based all-hexahedral mesh generation: handling sharp features. *Proceedings of the 18th International Meshing Roundtable,* Salt Lake City, UT, 2009; 65-84.

8. Zhang H, Zhao G. Adaptive hexahedral mesh generation based on local domain curvature and thickness using a modified grid-based method. *Finite Elements in Analysis and Design* 2007; 43:691-704. DOI: 10.1016/ j.finel.2007.03.001.

9. Murdoch P, Benzley S, Blacker T, Mitchell S. The spatial twist continuum: a connectivity based method for representing all-hexahedral finite element meshes. *Finite Elements in Analysis and Design* 1997; 28:137-149

10. Murdoch P, Benzley S. The spacial twist continuum. *Proceedings of the 4th International Meshing Roundtable*, Sandia National Laboratories, 1995; 243-251.

11. Knupp PM. Achieving finite element mesh quality via optimization of the Jacobian matrix norm and associated quantities. Part II-A framework for volume mesh optimization and the condition number of the Jacobian matrix. International Journal for Numerical Methods in Engineering 200; 48:1165-1185. DOI: 10.1002/(SICI) 1097-0207 (20000720) 48:8<1165::AID-NME940>3.0.CO;2-Y.

APPENDIX A: CREATION OF BASIC 2-REFINEMETN TEMPLATES

The templates used in two dimensional (2-D) 2-refinement will be used to create the templates to be used in three dimensional (3-D) 2-refinement. Figure A-1 illustrates the 2-D templates typically used for 2-refinement. These templates can first be applied to the six faces of a single hexahedral to create a surface mesh for a possible single hexahedral. An exhaustive search of all arrangements of the 2-D templates on these six faces yields 10 acceptable surface meshes as shown in Figure A-2. The last four of these prove impossible to mesh on the interior of the hexahedral. The remaining six with their valid volume mesh are shown in Figure 3-1. These represent all possible templates that can be used to replace a single hexahedral. The trivial non-refined template has not been shown in these figures.

Figure A-1: 2-D 2-Refinement Templates

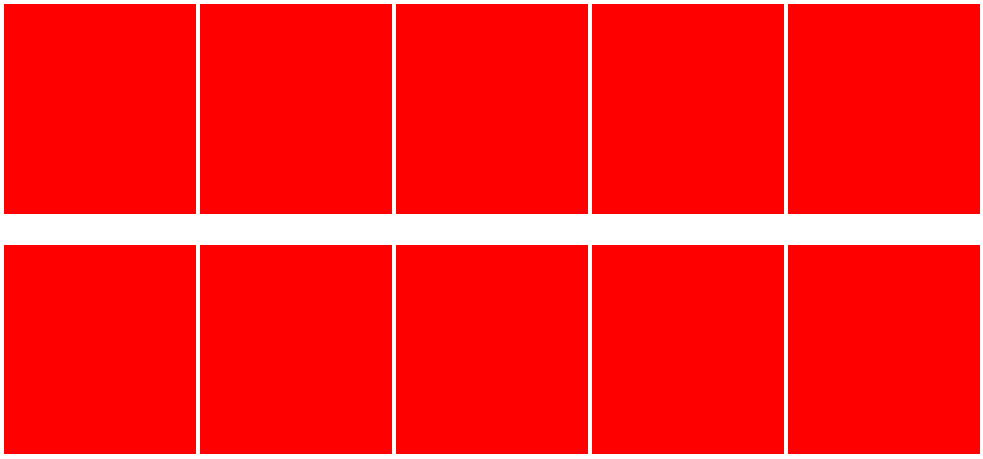


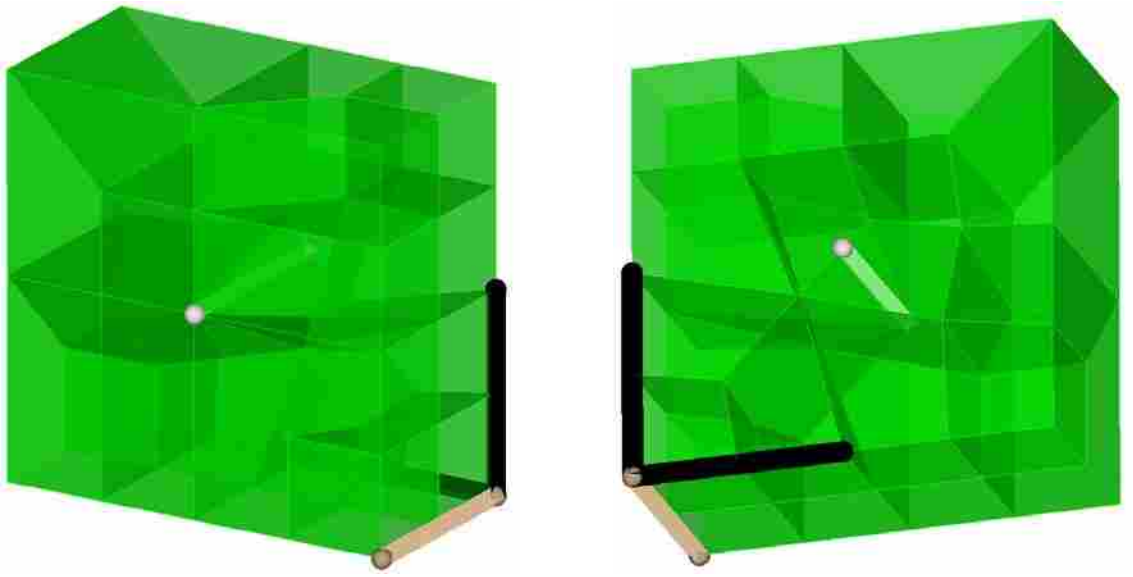
Figure A-2: Acceptable 2-Refinement Surface Meshes

APPENDIX B: TWO CONCAVE TEMPLATE CREATION

This template was created with a careful understanding of how the duals interact in a concavely refined region. First I set a basis for the size of a possible two concave template. With a two layer transition zone I knew the template could be a two by two by two. Once the size was established the faces on the template were created. The majority of the faces are already set by the neighboring hexes. The remaining faces were chosen based on how to bend the duals in a two concave region. With the size and faces set, a surface model was then created in CUBIT. These surfaces were used to create a volume and the surface was meshed. The interior of the volume was then meshed using the whisker weave meshing capability of CUBIT. Upon examination I found that the interior mesh was symmetric about a middle plane. This allowed the two by two by two to be split into the two by two by one as shown in Figure B-1. After smoothing the final scaled jacobian quality of the template is shown in Table B-1. A full list of nodes and their respective ids and X, Y and Z coordinates is provided in Table B-2. A full list of hexes with their respective node ids are provided in Table B-3.

Table B-1: Two Concave Template Element Quality

# Elements	Average	Std Dev	Min	Max
21	0.7402	1.36E-1	0.5177	0.9694



(a) Front (b) Back

Figure B-1: Two-Concave Template

Table B-2: Two Concave Template Node IDs and X, Y and Z Coordinates

Node ID	X	Y	Z
1	3.00	0.00	1.00
2	4.00	0.00	1.00
3	4.00	0.00	0.00
4	3.00	0.00	0.00
5	3.17	0.96	0.83
6	4.00	1.00	1.00
7	4.00	1.00	0.00
8	3.00	1.00	0.00
9	2.54	1.01	1.46
10	4.00	1.00	2.00
11	4.00	0.00	2.00
12	2.00	0.00	2.00
13	2.00	0.00	0.00
14	2.00	1.00	0.00
15	3.15	2.00	0.85
16	4.00	2.00	1.00
17	4.00	2.00	0.00
18	3.00	2.00	0.00
19	2.41	2.00	1.59
20	4.00	2.00	2.00
21	2.00	2.00	0.00

Node ID	X	Y	Z
22	0.87	2.00	2.06
23	0.00	2.00	2.00
24	0.00	0.00	2.00
25	1.00	0.00	2.00
26	0.95	2.00	3.05
27	0.00	2.00	4.00
28	0.00	0.00	4.00
29	1.00	0.00	3.00
30	1.00	2.00	0.00
31	1.05	2.00	1.17
32	1.07	1.04	1.04
33	1.00	1.00	0.00
34	2.83	2.00	2.95
35	4.00	2.00	3.00
36	4.00	2.00	4.00
37	3.00	2.00	4.00
38	2.96	1.04	2.93
39	4.00	1.00	3.00
40	4.00	1.00	4.00
41	3.00	1.00	4.00
42	0.00	1.00	0.00

Table B-2 (continued): Two Concave Template Node IDs and X, Y and Z Coordinates

Node ID	X	Y	Z
43	0.00	0.00	0.00
44	1.00	0.00	0.00
45	0.00	1.00	1.00
46	1.75	2.00	2.25
47	2.00	0.00	3.00
48	1.94	2.00	3.13
49	0.00	2.00	0.00
50	0.00	2.00	1.00
51	2.00	2.00	4.00
52	2.00	0.00	4.00
53	4.00	0.00	3.00
54	4.00	0.00	4.00

Table B-3: Two Concave Template Hex IDs and Respective Nodes

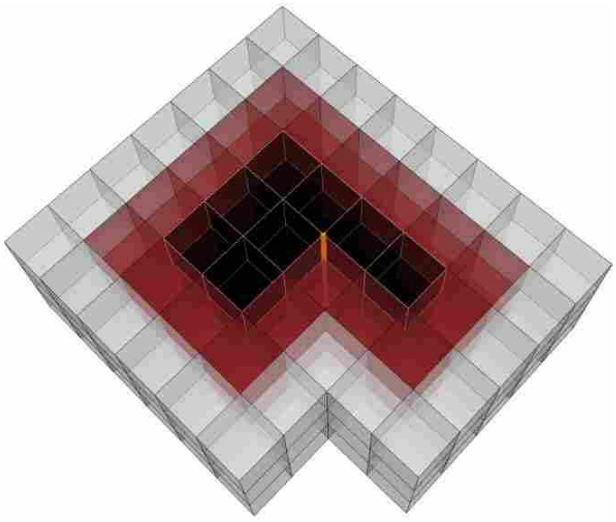
Hex ID	Node 1	Node 2	Node 3	Node 4	Node 5	Node 6	Node 7	Node 8
1	1	2	3	4	5	6	7	8
2	9	10	11	12	5	6	2	1
3	4	1	5	8	13	12	9	14
4	5	6	7	8	15	16	17	18
5	9	10	6	5	19	20	16	15
6	8	5	15	18	14	9	19	21
7	22	23	24	25	26	27	28	29
8	30	31	32	33	21	19	9	14
9	34	35	36	37	38	39	40	41
10	33	42	43	44	32	45	24	25
11	46	12	47	48	22	25	29	26
12	30	49	42	33	31	50	45	32
13	26	27	28	29	48	51	52	47
14	44	25	12	13	33	32	9	14
15	31	50	45	32	22	23	24	25
16	48	51	52	47	34	37	41	38
17	19	20	35	34	9	10	39	38
18	38	39	40	41	47	53	54	52
19	46	12	9	19	48	47	38	34
20	38	39	53	47	9	10	11	12
21	22	25	32	31	46	12	9	19

APPENDIX C: TWO CONCAVE EXAMPLE

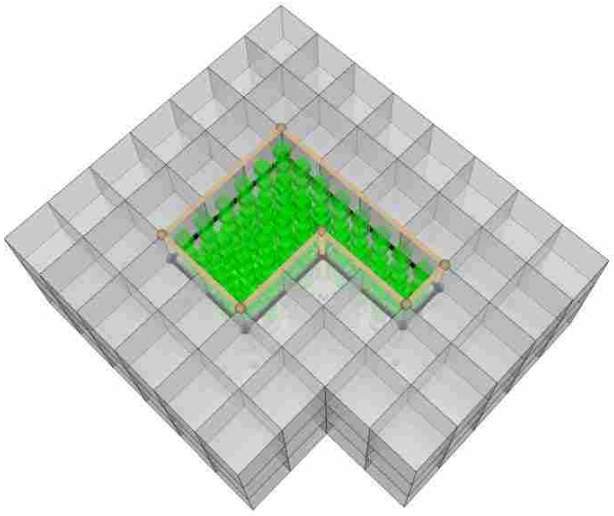
Figure C-1 shows how this template is used on a refined zone with a two-concave region. Figure C-1 (a) shows the URZ as black, the first transition zone as brown and the second transition zone as light grey. First, the hexes in the URZ are refined with Figure 3-1 (f). Next the rows of edges that wrap around the first boundary are alternately turned “on” (shaded black) or “off” (shaded tan) as shown in Figure C-1 (b).

Next, we insert the template from Figure B-1 into the concave zone. To do this we need to orient the template. The concave directional edge (shaded tan in Figure B-1) is matched to the edge found at the concave region in Figure C-1 (highlighted in Figure C-1(a)). The other two directional edges (shaded black in Figure B-1) are then matched to the first boundary edges marked “on” and that connect to the concave directional edge (shaded black in Figure B-1 (b)). This method of orienting the concave template is continued through the entire volume along the two-concave region. Figure C-1 (c) shows the concave template along with the rest of the templates used to finish the first transition zone.

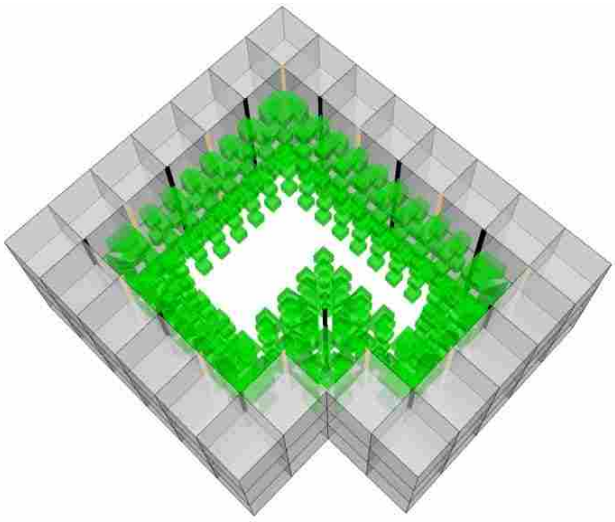
Next, the rows of edges that run longitudinally along the second boundary are alternately turned “on” or “off” as seen in Figure C-1 (c). Finally, the corner edges in the second boundary are inverted from “on” to “off” or “off” to “on”. All the edges that are turned “on” in the second boundary are then used to completely template the second transition zone. Notice that not all of the hexes in the second transition zone require a template and are thus left unchanged.



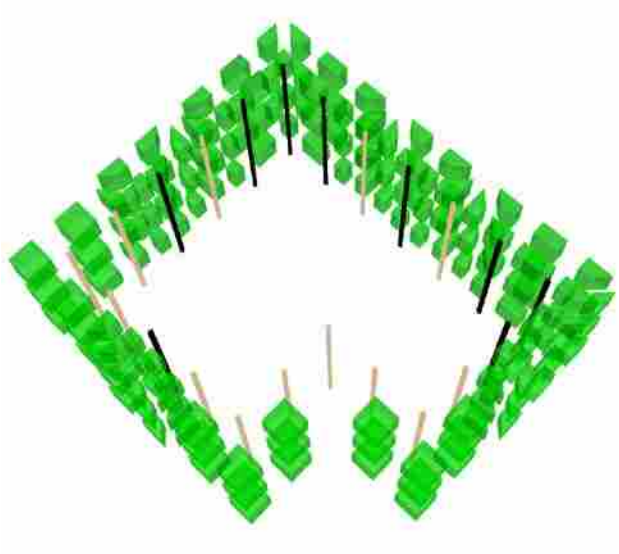
(a) Unrefined example with two concave region



(b) URZ is refined and first boundary edges and nodes are marked.

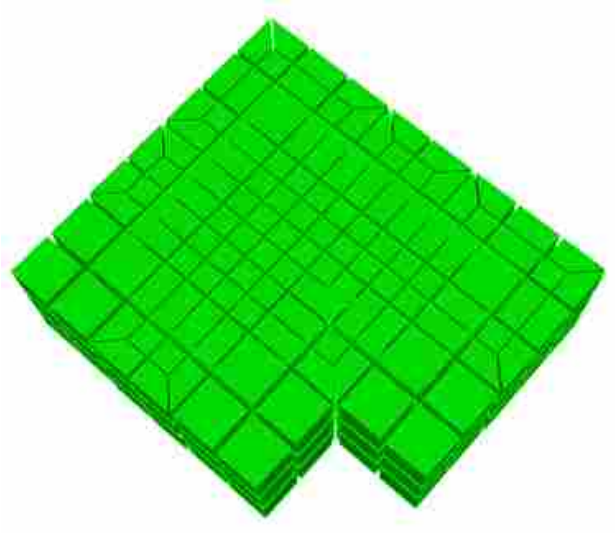


(c) Concave region and first transition zone are templated. Second boundary edges are marked.



(d) Corner edges are inverted. Second transition zone is templated.

Figure C-1: Two Concave Example



(e) Concave example completed.
Figure C-1 (continued): Two Concave Example

Also, notice that the template from Figure B-1 requires first transition and second transition elements. In order to match up the templates used in the second transition zone, the central edge of the template from Figure B-1 (shaded white) is used as a "seed" in the second boundary and starts the alternating of marking "on" or "off" starting with itself marked "on". It can be seen in Figure C-1 (c) that this central edge is marked "on" (shaded black). This algorithm does allow multiple concave regions within the same mesh with one restriction. This restriction is based on the central edge lining up from one concave region to the next. This means that the number of rows of edges in the second boundary from one concave region to the next must be even.

APPENDIX D: THREE CONCAVE TEMPLATE CREATION

The three concave template was created similarly to the two concave template. First I considered the size that I would be allowed to have for the template. Then after setting the size it became evident that after all the other templates in the model had been placed every face on the three concave template would be decided. With these faces I then created a surface model in CUBIT and I used those surfaces to create a volume. I then meshed the surface and used CUBIT's whisker weaving algorithm to mesh the interior. The first result had a total of 488 elements. This was unfeasible as the transition zone is not meant for refinement but would have more new elements introduced than the URZ. It was discovered that the first result from the whisker weaver had inserted a local dual sheet around the entire template. This sheet was introduced to raise element quality. After extracting this sheet the number of elements was reduced to 200. But this came at a cost of reduced element quality which now requires smoothing after the three concave template has been used. Before smoothing the scaled jacobian quality of the template is shown in Table D-1. A full list of nodes and their respective ids and X, Y and Z coordinates is provided in Table D-2. A full list of hexes with their respective node ids are provided in Table D-3.

Table D-1: Three Concave Template Element Quality

# Elements	Average	Std Dev	Min	Max
200	0.1708	2.92E-1	0.0000	0.9221

Table D-2: Three Concave Template Node IDs and X, Y and Z Coordinates

Node ID	X	Y	Z
1	0.90	1.60	2.65
2	0.85	1.44	3.64
3	0.79	0.77	3.65
4	0.00	0.00	2.50
5	1.63	1.63	2.65
6	1.47	1.47	3.63
7	1.43	0.83	3.63
8	1.59	0.84	2.61
9	5.00	0.00	0.00
10	3.61	0.77	0.80
11	3.52	0.40	0.97
12	5.00	0.00	0.63
13	4.61	0.72	1.34
14	3.60	0.85	1.46
15	3.51	0.42	1.40
16	5.00	0.00	1.25
17	4.38	0.00	1.25
18	4.38	0.00	0.63
19	3.75	0.00	1.25
20	3.75	0.00	0.63
21	3.13	1.25	0.00
22	2.50	1.25	0.00
23	2.50	0.63	0.00
24	3.13	0.63	0.00
25	3.75	1.25	0.00
26	3.60	1.37	0.45
27	3.61	0.94	0.43
28	3.75	0.63	0.00
29	0.67	5.65	1.28
30	0.00	7.50	1.25
31	0.00	7.50	2.50
32	0.00	6.25	2.50
33	1.30	5.65	1.29
34	1.25	7.50	1.25
35	1.25	7.50	2.50
36	1.25	6.25	2.50
37	1.25	7.50	0.00
38	1.25	6.25	0.00
39	0.00	7.50	0.00
40	0.00	6.25	0.00

Node ID	X	Y	Z
41	0.75	4.65	1.34
42	0.00	5.00	2.50
43	1.37	4.64	1.37
44	1.25	5.00	2.50
45	1.33	4.65	0.74
46	0.00	5.00	0.00
47	0.73	2.08	3.58
48	0.00	2.50	3.75
49	0.00	2.50	3.13
50	0.00	2.50	2.50
51	0.42	1.81	3.59
52	0.00	1.88	3.75
53	0.00	1.88	3.13
54	0.00	1.88	2.50
55	0.86	2.61	1.61
56	0.00	2.50	1.25
57	0.00	2.50	0.63
58	0.00	2.50	0.00
59	0.60	2.04	1.63
60	0.00	1.88	1.25
61	0.00	1.88	0.63
62	0.00	1.88	0.00
63	1.88	0.00	1.88
64	2.50	0.00	1.88
65	2.50	0.00	1.25
66	1.88	0.00	1.25
67	1.25	0.00	1.88
68	1.63	0.51	2.06
69	1.66	0.51	1.53
70	1.25	0.00	1.25
71	1.36	0.41	3.67
72	2.05	0.72	3.59
73	1.79	0.40	3.66
74	1.25	0.00	2.50
75	2.50	0.00	2.50
76	1.88	0.00	2.50
77	2.50	6.25	1.25
78	2.50	7.50	1.25
79	2.50	7.50	2.50
80	2.50	6.25	2.50

Table D-2 (continued): Three Concave Template Node IDs and X, Y and Z Coordinates

Node ID	X	Y	Z
81	2.50	7.50	0.00
82	2.50	6.25	0.00
83	2.50	5.00	1.25
84	2.50	5.00	2.50
85	2.50	5.00	0.00
86	4.38	0.00	2.50
87	5.00	0.00	2.50
88	5.00	0.00	1.88
89	4.38	0.00	1.88
90	3.75	0.00	2.50
91	3.54	0.75	2.08
92	3.49	0.42	1.83
93	3.75	0.00	1.88
94	0.94	0.39	3.68
95	0.63	0.00	2.50
96	4.62	1.35	1.38
97	3.61	1.48	1.48
98	3.56	1.39	2.12
99	5.00	1.25	2.50
100	1.39	3.58	2.11
101	1.25	2.50	2.50
102	0.76	3.60	2.07
103	1.47	3.62	1.48
104	1.63	2.63	1.64
105	0.85	3.64	1.45
106	4.62	1.32	0.77
107	3.61	1.43	0.87
108	0.44	1.38	3.61
109	0.00	1.25	3.75
110	0.00	1.25	3.13
111	0.00	1.25	2.50
112	0.42	0.95	3.62
113	0.00	0.63	3.75
114	0.00	0.63	3.13
115	0.00	0.63	2.50
116	0.42	3.74	1.81
117	0.00	3.75	1.88
118	0.00	3.13	1.88
119	0.00	2.50	1.88
120	0.00	3.75	2.50

Node ID	X	Y	Z
121	0.00	3.13	2.50
122	3.13	2.50	0.00
123	2.50	2.50	0.00
124	2.50	1.88	0.00
125	3.13	1.88	0.00
126	3.75	2.50	0.00
127	3.57	2.07	0.75
128	3.58	1.80	0.43
129	3.75	1.88	0.00
130	4.38	1.25	2.50
131	3.75	1.25	2.50
132	0.63	0.63	0.00
133	0.00	0.63	0.00
134	0.00	0.00	0.00
135	0.63	0.00	0.00
136	1.25	0.63	0.00
137	1.65	1.01	0.59
138	1.68	0.89	0.95
139	1.25	0.00	0.00
140	1.25	5.00	0.00
141	1.25	5.63	0.00
142	1.88	5.63	0.00
143	1.88	5.00	0.00
144	1.03	1.72	1.76
14	0.62	1.51	1.66
146	4.38	0.00	0.00
147	3.75	0.00	0.00
148	0.63	5.63	0.00
149	0.63	5.00	0.00
150	0.63	0.00	3.75
151	0.63	0.00	3.13
152	1.25	0.00	3.75
153	1.25	0.00	3.13
154	0.43	3.76	1.38
155	5.00	0.63	0.00
156	5.00	1.25	0.00
157	0.00	0.00	0.63
158	0.63	0.00	0.63
159	1.65	0.47	1.04
160	1.25	0.00	0.63

Table D-2 (continued): Three Concave Template Node IDs and X, Y and Z Coordinates

Node ID	X	Y	Z
161	0.00	1.25	6.88
162	0.00	1.25	6.25
163	0.00	1.88	6.25
164	0.00	1.88	6.88
165	0.00	1.25	7.50
166	0.66	1.27	5.64
167	0.00	2.50	6.25
168	0.00	2.50	7.50
169	1.59	2.60	0.88
170	2.50	2.50	1.25
171	1.76	1.74	1.77
172	1.75	1.70	1.04
173	2.63	1.59	0.92
174	2.63	1.64	1.66
175	2.50	0.00	0.00
176	2.62	0.89	1.61
177	0.00	0.63	6.88
178	0.00	0.63	6.25
179	0.00	0.00	7.50
180	0.00	0.00	6.25
181	2.50	1.25	2.50
182	2.50	1.25	3.75
183	0.00	3.75	1.25
184	0.00	3.13	1.25
185	0.74	1.33	4.64
186	0.00	2.50	5.00
187	0.00	1.25	5.00
188	0.00	1.88	5.00
189	1.88	0.00	0.00
190	1.88	0.63	0.00
191	0.00	0.00	5.00
192	0.00	0.63	5.00
193	1.88	0.00	0.63
194	2.50	0.00	0.63
195	2.10	3.58	1.40
196	2.50	3.75	1.25
197	2.50	3.13	1.25
198	2.05	3.58	0.77
199	2.50	3.75	0.00
200	2.50	3.13	0.00

Node ID	X	Y	Z
201	1.25	2.50	6.25
202	1.25	2.50	7.50
203	1.29	1.27	5.66
204	1.25	1.25	7.50
205	1.25	0.00	7.50
206	1.25	0.00	6.25
207	3.13	2.50	0.63
208	3.75	2.50	0.63
209	3.75	2.50	1.25
210	0.63	0.00	1.88
211	0.00	0.00	1.88
212	1.35	1.35	4.65
213	1.25	2.50	5.00
214	1.32	0.72	4.65
215	3.13	0.00	0.00
216	3.13	0.00	0.63
217	1.00	1.68	1.01
218	0.00	1.25	0.00
219	3.13	0.00	2.50
220	3.13	1.25	2.50
221	1.88	1.88	0.00
222	1.88	2.50	0.00
223	1.25	1.88	0.00
224	1.62	2.03	0.61
225	1.25	2.50	0.00
226	0.97	0.94	1.70
227	0.61	1.02	1.65
228	0.63	0.00	5.63
229	0.63	0.00	5.00
230	1.25	0.00	5.63
231	1.25	0.00	5.00
232	2.50	0.00	7.50
233	2.50	0.00	6.25
234	2.50	1.25	7.50
235	2.50	1.25	6.25
236	3.13	0.00	1.25
237	3.13	0.00	1.88
238	2.50	0.00	5.00
239	2.50	1.25	5.00
240	1.88	0.00	5.63

Table D-2 (continued): Three Concave Template Node IDs and X, Y and Z Coordinates

Node ID	X	Y	Z
241	1.88	0.00	5.00
242	2.50	2.50	7.50
243	2.50	2.50	6.25
244	2.50	3.75	2.50
245	2.50	3.75	1.88
246	2.50	3.13	1.88
247	2.50	2.50	2.50
248	2.50	2.50	5.00
249	1.74	0.98	1.73
250	0.00	5.00	1.88
251	0.00	5.00	1.25
252	0.00	4.38	1.88
253	0.00	4.38	1.25
254	0.00	5.00	0.63
255	0.40	3.77	0.95
256	0.78	3.67	0.79
257	1.43	3.62	0.86
258	0.00	3.75	0.63
259	0.00	4.38	0.63
260	0.00	4.38	2.50
261	0.00	0.00	1.25
262	1.25	3.75	2.50
263	1.25	4.38	2.50
264	1.88	0.00	3.75
265	1.88	0.00	3.13
266	1.88	3.75	2.50
267	1.88	4.38	2.50
268	2.50	4.38	1.25
269	2.50	4.38	1.88
270	1.88	1.25	0.00
271	1.65	1.50	0.63
272	1.25	1.25	0.00
273	2.50	4.38	0.00
274	0.63	0.00	1.25
275	2.50	0.00	3.75
276	2.50	0.00	3.13
277	1.79	3.65	0.43
278	1.36	3.67	0.44
279	0.94	3.68	0.42
280	4.38	1.25	0.00

Node ID	X	Y	Z
281	5.00	1.88	0.00
282	4.38	1.88	0.00
283	1.88	3.75	0.00
284	1.88	4.38	0.00
285	5.00	2.50	0.00
286	5.00	2.50	1.25
287	1.25	3.75	0.00
288	1.25	4.38	0.00
289	3.13	1.88	2.50
290	3.75	2.50	2.50
291	3.75	1.88	2.50
292	1.25	2.50	3.75
293	0.63	3.75	0.00
294	0.63	4.38	0.00
295	0.00	3.75	0.00
296	0.00	4.38	0.00
297	4.38	0.63	0.00
298	5.00	2.50	2.50
299	0.63	0.00	4.38
300	0.00	0.00	3.75
301	0.00	0.00	4.38
302	0.00	0.63	1.25
303	0.00	0.63	1.88
304	0.00	1.25	1.25
305	0.00	1.25	1.88
306	0.00	0.00	3.13
307	0.00	0.63	4.38
308	1.25	3.13	2.50
309	1.25	0.00	4.38
310	0.00	3.13	0.63
311	1.88	0.00	4.38
312	2.50	0.00	4.38
313	2.50	0.63	3.75
314	2.50	0.63	4.38
315	2.50	2.50	3.75
316	4.38	2.50	0.63
317	4.38	2.50	0.00
318	0.00	2.50	4.38
319	0.63	2.50	3.75
320	0.63	2.50	4.38

Table D-2 (continued): Three Concave Template Node IDs and X, Y and Z Coordinates

Node ID	X	Y	Z
321	0.00	1.88	1.88
322	0.00	1.88	4.38
323	0.00	1.25	4.38
324	0.63	2.50	0.00
325	0.63	1.88	0.00
326	0.63	3.13	0.00
327	0.00	3.13	0.00
328	6.25	1.25	2.50
329	7.50	1.25	2.50
330	7.50	0.00	2.50
331	6.25	0.00	2.50
332	5.63	1.26	1.32
333	7.50	1.25	1.25
334	7.50	0.00	1.25
335	6.25	0.00	1.25
336	1.88	3.13	0.00
337	7.50	2.50	1.25
338	6.25	2.50	1.25
339	7.50	2.50	2.50
340	6.25	2.50	2.50
341	0.00	1.25	0.63
342	0.63	2.50	3.13
343	1.25	3.13	0.00
344	5.63	0.00	1.25
345	5.63	0.00	1.88
346	0.63	1.25	0.00
347	5.62	1.26	0.68
348	7.50	1.25	0.00
349	7.50	0.00	0.00
350	6.25	0.00	0.00
351	7.50	2.50	0.00
352	6.25	2.50	0.00
353	0.00	0.63	0.63
354	5.63	0.00	0.63
355	6.25	1.25	0.00
356	6.88	1.25	0.00
357	6.88	0.63	0.00
358	6.25	0.63	0.00
359	6.88	1.88	0.00
360	6.25	1.88	0.00

Node ID	X	Y	Z
361	4.38	1.88	2.50
362	2.50	0.63	3.13
363	1.88	3.13	2.50
364	0.00	6.25	1.25
365	0.00	6.88	1.25
366	0.00	6.88	1.88
367	0.00	6.25	1.88
368	0.00	6.88	0.63
369	0.00	6.25	0.63

Table D-3: Three Concave Template Hex IDs and Respective Node IDs

Hex ID	Node 1	Node 2	Node 3	Node 4	Node 5	Node 6	Node 7	Node 8
1	1	2	3	4	5	6	7	8
2	9	10	11	12	13	14	15	16
3	17	16	12	18	19	15	11	20
4	21	22	23	24	25	26	27	28
5	29	30	31	32	33	34	35	36
6	33	34	37	38	29	30	39	40
7	41	29	32	42	43	33	36	44
8	43	33	38	45	41	29	40	46
9	47	48	49	50	51	52	53	54
10	55	56	57	58	59	60	61	62
11	63	64	65	66	67	68	69	70
12	71	7	72	73	74	8	75	76
13	33	34	35	36	77	78	79	80
14	33	34	78	77	38	37	81	82
15	43	33	36	44	83	77	80	84
16	43	33	77	83	45	38	82	85
17	86	87	88	89	90	91	92	93
18	94	3	7	71	95	4	8	74
19	13	14	91	87	96	97	98	99
20	100	101	50	102	103	104	55	105
21	9	10	14	13	106	107	97	96
22	108	109	110	111	112	113	114	115
23	116	117	118	119	102	120	121	50
24	122	123	124	125	126	127	128	129
25	130	99	87	86	131	98	91	90
26	132	133	134	135	136	137	138	139
27	45	38	82	85	140	141	142	143
28	144	145	111	1	55	59	54	50
29	18	12	9	146	20	11	10	147
30	45	38	141	140	46	40	148	149
31	94	150	151	95	71	152	153	74
32	116	119	56	154	102	50	55	105
33	155	27	10	9	156	26	107	106
34	135	134	157	158	139	138	159	160
35	161	162	163	164	165	166	167	168
36	104	169	123	170	171	172	173	174
37	10	175	176	14	107	173	174	97
38	177	178	162	161	179	180	166	165
39	5	6	7	8	181	182	72	75
40	154	183	184	56	116	117	118	119

Table D-3 (continued): Three Concave Template Hex IDs and Respective Node IDs

Hex ID	Node 1	Node 2	Node 3	Node 4	Node 5	Node 6	Node 7	Node 8
41	185	166	167	186	187	162	163	188
42	189	175	23	190	139	138	137	136
43	185	166	162	187	191	180	178	192
44	50	101	5	1	55	104	171	144
45	193	194	175	189	160	159	138	139
46	195	196	197	170	198	199	200	123
47	168	167	201	202	165	166	203	204
48	205	206	180	179	204	203	166	165
49	207	170	123	122	208	209	127	126
50	210	211	4	95	67	68	8	74
51	212	203	201	213	185	166	167	186
52	185	166	180	191	212	203	206	214
53	215	175	194	216	147	10	11	20
54	144	145	59	55	217	218	62	58
55	27	23	175	10	26	22	173	107
56	219	75	181	220	90	91	98	131
57	24	23	175	215	28	27	10	147
58	221	124	123	222	223	224	169	225
59	226	227	115	4	144	145	111	1
60	191	180	228	229	214	206	230	231
61	14	176	65	15	91	75	64	92
62	232	233	206	205	234	235	203	204
63	236	65	64	237	19	15	92	93
64	214	206	233	238	212	203	235	239
65	216	194	65	236	20	11	15	19
66	214	206	230	231	238	233	240	241
67	242	243	235	234	202	201	203	204
68	244	245	246	247	195	196	197	170
69	212	203	235	239	213	201	243	248
70	1	2	47	50	111	108	51	54
71	10	175	194	11	14	176	65	15
72	171	104	101	5	174	170	247	181
73	66	65	194	193	70	69	159	160
74	209	170	123	127	97	174	173	107
75	8	5	181	75	249	171	174	176
76	249	69	159	138	176	65	194	175
77	250	116	154	251	42	102	105	41
78	116	117	252	250	154	183	253	251
79	195	170	247	244	103	104	101	100
80	254	255	256	46	251	154	105	41

Table D-3 (continued): Three Concave Template Hex IDs and Respective Node IDs

Hex ID	Node 1	Node 2	Node 3	Node 4	Node 5	Node 6	Node 7	Node 8
81	198	123	170	195	257	169	104	103
82	154	183	253	251	255	258	259	254
83	217	218	133	134	144	145	227	226
84	102	120	260	42	116	117	252	250
85	44	100	102	42	43	103	105	41
86	46	256	257	45	41	105	103	43
87	249	69	68	8	226	261	211	4
88	100	262	263	44	102	120	260	42
89	83	195	244	84	43	103	100	44
90	85	198	195	83	45	257	103	43
91	71	152	153	74	73	264	265	76
92	100	262	266	244	44	263	267	84
93	195	196	268	83	244	245	269	84
94	134	217	144	226	138	172	171	249
95	190	23	22	270	136	137	271	272
96	198	199	273	85	195	196	268	83
97	158	157	261	274	160	159	69	70
98	73	264	265	76	72	275	276	75
99	143	277	198	85	140	278	257	45
100	149	279	278	140	46	256	257	45
101	280	156	281	282	25	26	128	129
102	277	283	284	143	198	199	273	85
103	281	128	26	156	285	127	107	106
104	286	209	127	285	96	97	107	106
105	278	287	288	140	277	283	284	143
106	220	181	247	289	131	98	290	291
107	5	6	292	101	1	2	47	50
108	279	293	294	149	278	287	288	140
109	256	295	296	46	279	293	294	149
110	146	9	155	297	147	10	27	28
111	298	290	209	286	99	98	97	96
112	255	258	259	254	256	295	296	46
113	7	214	191	3	71	231	229	94
114	94	150	299	229	3	300	301	191
115	227	302	303	115	145	304	305	111
116	6	212	185	2	7	214	191	3
117	108	187	192	112	2	185	191	3
118	112	113	114	115	3	300	306	4
119	3	300	301	191	112	113	307	192
120	226	261	211	4	227	302	303	115

Table D-3 (continued): Three Concave Template Hex IDs and Respective Node IDs

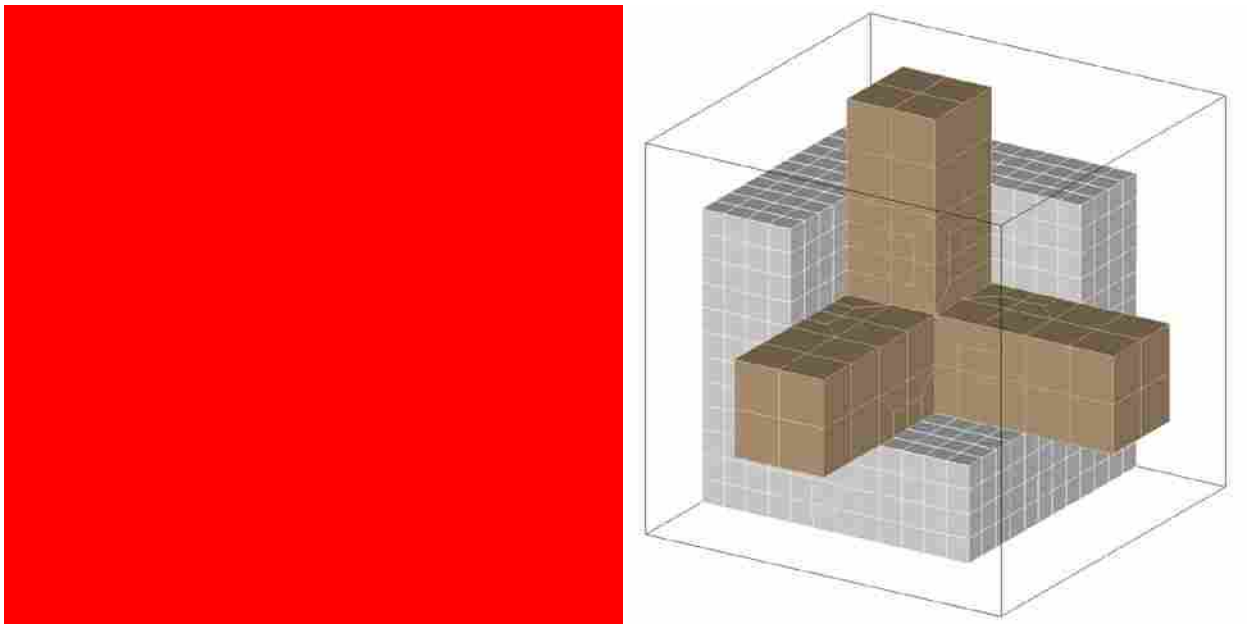
Hex ID	Node 1	Node 2	Node 3	Node 4	Node 5	Node 6	Node 7	Node 8
121	102	120	121	50	100	262	308	101
122	71	152	309	231	94	150	299	229
123	7	214	231	71	72	238	241	73
124	7	214	238	72	6	212	239	182
125	255	258	310	57	154	183	184	56
126	172	271	137	138	217	218	133	134
127	73	264	311	241	71	152	309	231
128	72	275	312	238	73	264	311	241
129	111	108	112	115	1	2	3	4
130	270	22	124	221	272	271	224	223
131	72	275	313	182	238	312	314	239
132	6	212	239	182	292	213	248	315
133	292	213	186	47	6	212	185	2
134	3	300	306	4	94	150	151	95
135	316	286	209	208	317	285	127	126
136	104	169	172	171	55	58	217	144
137	125	124	22	21	129	128	26	25
138	47	48	318	186	292	319	320	213
139	145	304	305	111	59	60	321	54
140	51	188	187	108	47	186	185	2
141	217	218	62	58	172	271	224	169
142	51	52	322	188	47	48	318	186
143	108	109	323	187	51	52	322	188
144	172	271	22	173	138	137	23	175
145	175	173	172	138	176	174	171	249
146	112	113	307	192	108	109	323	187
147	324	58	62	325	225	169	224	223
148	279	293	326	324	256	295	327	58
149	13	14	15	16	87	91	92	88
150	89	88	16	17	93	92	15	19
151	328	329	330	331	332	333	334	335
152	297	155	156	280	28	27	26	25
153	198	199	200	123	277	283	336	222
154	332	333	337	338	328	329	339	340
155	335	13	87	331	332	96	99	328
156	59	60	61	62	145	304	341	218
157	340	298	286	338	328	99	96	332
158	277	198	257	278	222	123	169	225
159	5	6	182	181	101	292	315	247
160	4	1	5	8	226	144	171	249

Table D-3 (continued): Three Concave Template Hex IDs and Respective Node IDs

Hex ID	Node 1	Node 2	Node 3	Node 4	Node 5	Node 6	Node 7	Node 8
161	292	319	342	101	47	48	49	50
162	277	283	336	222	278	287	343	225
163	344	16	88	345	335	13	87	331
164	325	62	218	346	223	224	271	272
165	332	333	334	335	347	348	349	350
166	347	348	351	352	332	333	337	338
167	350	9	13	335	347	106	96	332
168	338	286	285	352	332	96	106	347
169	14	176	75	91	97	174	181	98
170	278	287	343	225	279	293	326	324
171	227	302	353	133	226	261	157	134
172	237	64	75	219	93	92	91	90
173	354	12	16	344	350	9	13	335
174	347	348	349	350	355	356	357	358
175	347	348	356	355	352	351	359	360
176	358	155	9	350	355	156	106	347
177	360	281	156	355	352	285	106	347
178	279	256	58	324	278	257	169	225
179	290	247	170	209	98	181	174	97
180	249	69	65	176	8	68	64	75
181	361	298	99	130	291	290	98	131
182	346	218	133	132	272	271	137	136
183	282	281	285	317	129	128	127	126
184	172	271	224	169	173	22	124	123
185	72	275	276	75	182	313	362	181
186	128	124	22	26	127	123	173	107
187	100	262	308	101	244	266	363	247
188	51	52	53	54	108	109	110	111
189	256	295	327	58	255	258	310	57
190	256	58	169	257	105	55	104	103
191	226	261	157	134	249	69	159	138
192	255	57	58	256	154	56	55	105
193	364	365	366	367	29	30	31	32
194	76	75	64	63	74	8	68	67
195	145	304	341	218	227	302	353	133
196	29	30	39	40	364	365	368	369
197	251	364	367	250	41	29	32	42
198	59	60	321	54	55	56	119	50
199	41	29	40	46	251	364	369	254
200	274	261	211	210	70	69	68	67

APPENDIX E: THREE CONCAVE TEMPLATE AND PILLOWING EXAMPLE

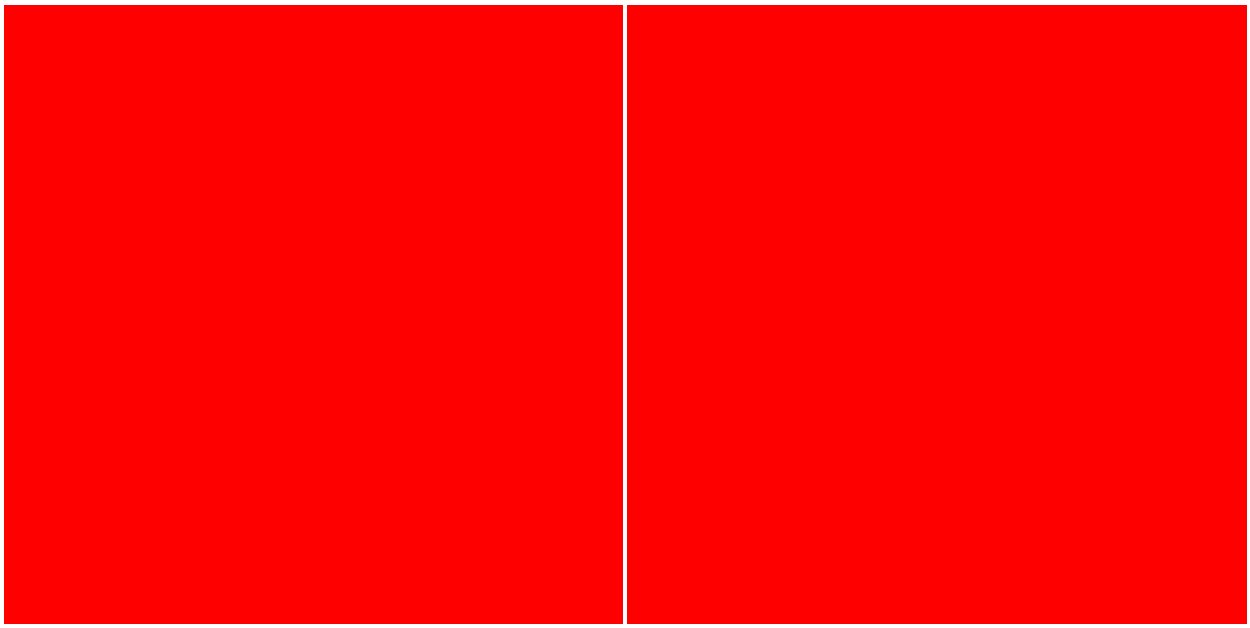
The example used for three directional 2-refinement is an eight by eight by eight structured mesh with a three-concave region that is four by four by four hexes as seen in Figure E-1 (a). The remaining hexes are not drawn for illustration purposes but the extents of the model are shown. This same problem is solved using three different methods. The first method is the three directional 2-refinement algorithm using templates discussed in section 3.3. The three-concave template from Figure 3-11 is used.



(a) Refined concave region before smoothing the entire model.

(b) Three concave template inserted before smoothing the entire model.

Figure E-1: Three-Concave Example Using Three Directional 2-Refinement Algorithm

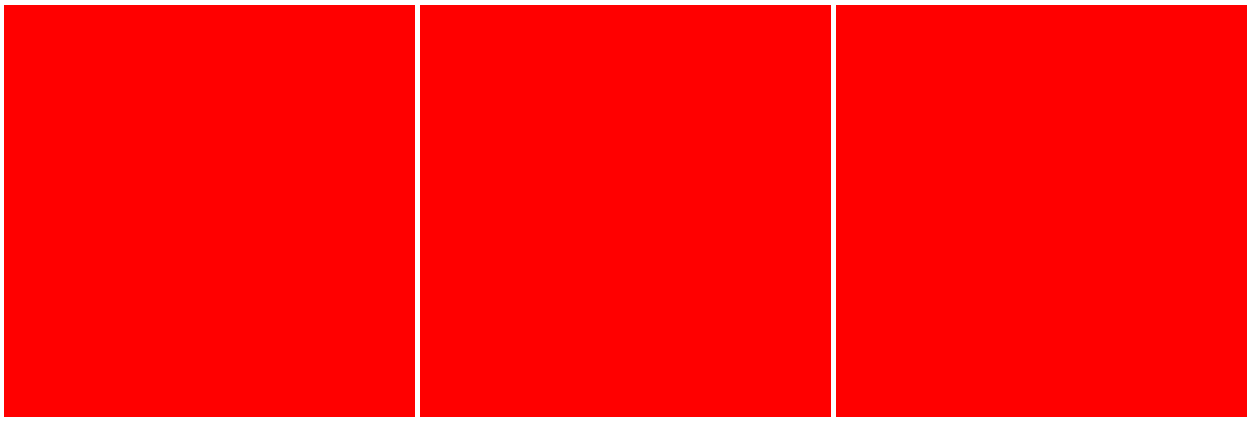


(a) Refined concave region after smoothing the entire model.

(b) Three concave template inserted after smoothing the entire model.

Figure E-1 (continued): Three-Concave Example Using Three Directional 2-Refinement Algorithm

Figure E-2 is solving the exact same three-concave refinement problem as in Figure E-1 (a) but instead uses a pillowing method. It is pillowed using only a single layer for the transition zone. Only the hexes being pillowed are shown for illustrative purposes but the extents of the model are outlined.

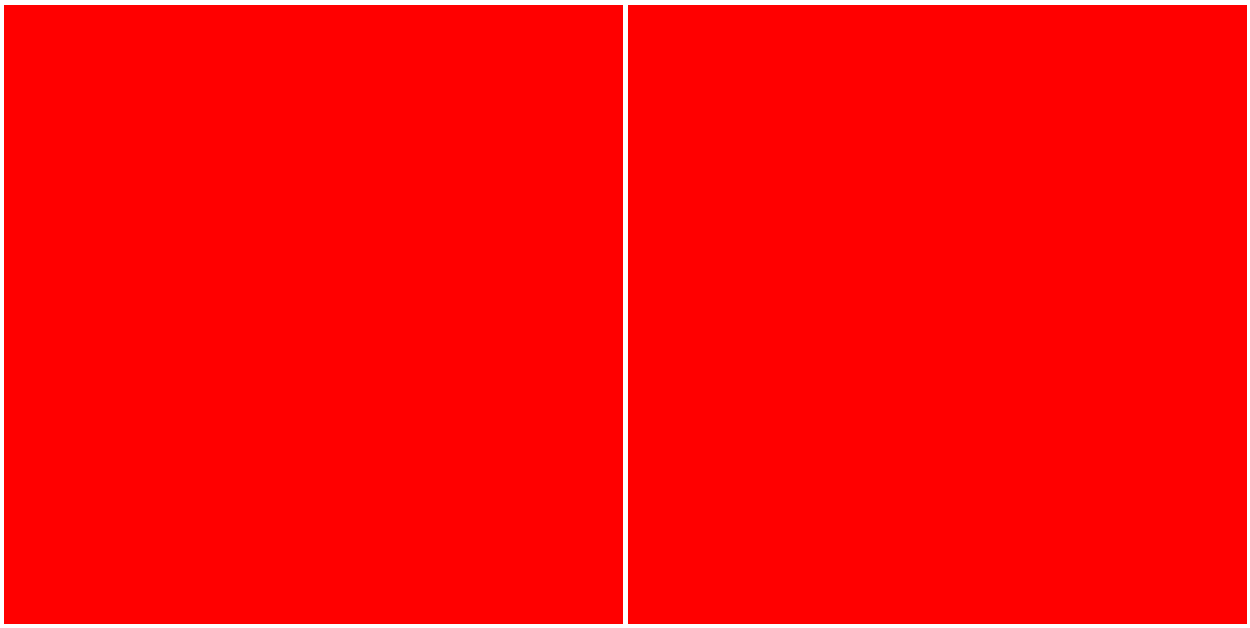


(a) First pillow

(b) Second pillow

(c) Third pillow

Figure E-2: Three-Concave Example Using Pillowing Method With One Transition Layer

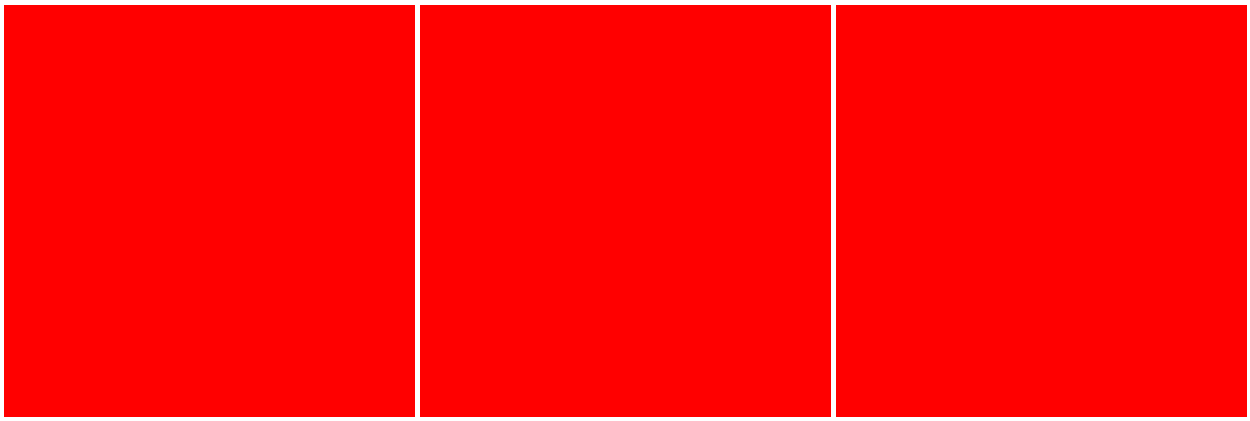


(d) Final result of refined zone after smoothing.

(e) Final templates created in three-concave zone.

Figure E-2 (continued): Three-Concave Example Using Pillowing Method With One Transition Layer

Figure E-3 is solving the exact same three-concave refinement problem as in Figure E-1 (a) but uses a different pillowing method. It now uses two layers for the transition zone. Only the hexes being pillowed are shown for illustrative purposes but the extents of the model are outlined. Tables E-1 and E-2 show the comparison between the three different methods.

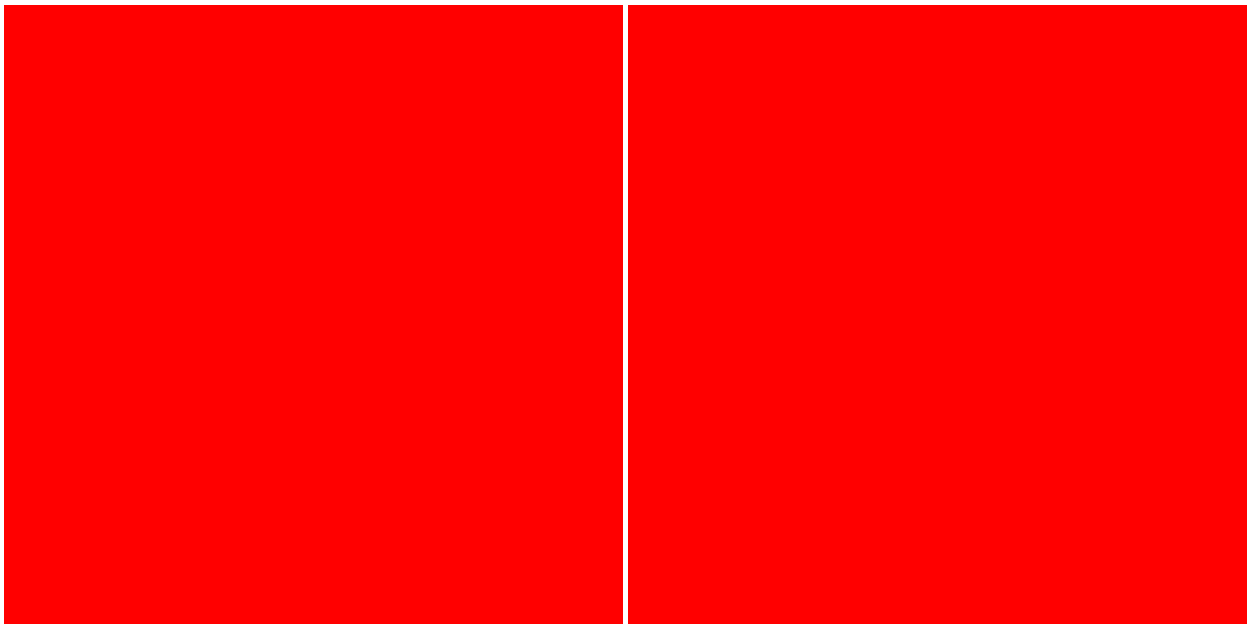


(a) First pillow

(b) Second pillow

(c) Third pillow

Figure E-3: Three-Concave Example Using Pillowing Method With Two Transition Layers



(a) Final result of refined zone after smoothing.

(b) Final templates created in three-concave zone.

Figure E-3 (continued): Three-Concave Example Using Pillowing Method With Two Transition Layers

Table E-1: Comparison of Results for the Entire Model for Three Directional Example

	# Elements	Ave	Std Dev	Min	Max
Three directional 2 -refinement	2,320	0.8804	1.46E-01	0.4531	0.9999
Pillow 1 Layer	2,330	0.8885	1.45E-01	0.4481	0.9998
Pillow 2 Layers	2,452	0.8926	1.32E-01	0.4322	0.9995

Table E-2: Comparison of Results for the Concave Region Elements for Three Directional Example

	# Elements	Ave	Std Dev	Min	Max
three directional 2 -refinement	200	0.7015	1.33E-01	0.4741	0.9851
Pillow 1 Layer	324	0.7550	1.29E-01	0.4481	0.9777
Pillow 2 Layers	332	0.7663	1.26E-01	0.4322	0.9708

It is shown that the three directional 2-refinement algorithm introduces the fewest number of elements and has comparable quality to the other two methods. In addition, the three directional 2-refinement algorithm employs templates which keeps computational time linear, compared to

pillowing which has scalability issues. Parish [4] solved these scalability issues by using templates in convex areas and pillowing only when necessary in concave areas. This helped mitigate the exponential computational time required given the amount of refinement. The proposed three directional 2-refinement algorithm here solves the problem completely by using templates in both convex and concave regions.