APPENDIX A
LOCAL MESH MODIFICATIONS AND NODE REPOSITIONING

In this chapter the local mesh modification and node repositioning procedures used in the previously described work are described. Local mesh modification of tetrahedral meshes consist of three basic operations [15, 18]:

- Edge, face or region split - introduces one new vertex into the mesh.
- Edge swap - Does not change the number of nodes in the mesh.
- Edge collapse - Deletes one node from the mesh.

Complex transformations of tetrahedral meshes can be effected by application of one or more of these procedures. The local mesh modification procedures always maintain a valid topological connectivity of the mesh. However, additional procedures are required to maintain topological validity of the mesh with the model and geometric validity of the elements.

A.1 Edge Split

An edge split operation breaks an edge into two edges and also splits each of the connected higher order entities into two entities. The edge split for surface meshes consists of the following steps (Figure A.1):

- Create a new vertex at the split location. This vertex inherits the classification of the split edge.
- Create two new edges between the new vertex and vertices of the split edge. The new edges inherit the classification of the new edge.
- Split each face connected to the original edge with an edge between new vertex to the face vertex opposite the original edge. The faces inherit the classification of the respective original faces.
For volume meshes one additional step follows the steps in two dimensions. The regions connected to the original edge are divided into two by introducing a face between the new vertex and the two vertices of the region opposite the original edge. This is illustrated in Figure A.2.

- Split location on straight edge
- Split point pulled to model boundary

Figure A.1: Edge split on surface meshes.

Figure A.2: Edge split in volume meshes.

If the edge being split is a boundary edge, then the split point must be located on the boundary. In some situations, this may make the elements invalid according to some measure (See Section A.7). The split operation cannot create any topological incompatibility of the mesh with the model.
A.2 Face Split

A face split divides a face into three new faces. Additionally, for volume meshes, it divides each region into three new regions. To perform a face split, a new vertex is created inside the face. Three new faces are created by connecting the new vertex to two vertices of the original face in turn. If the face has tetrahedra connected to it, each of the new faces is combined with the fourth vertex of the tetrahedron to form a new region (Figure A.3a). As with an edge split, the face split operation cannot in itself produce any topological incompatibility of the mesh with the model. Also, if the face is a boundary face, the newly created point must be relocated on the model boundary and the validity of the element must be checked with respect to that location.

A.3 Region Split

A region split divides a region into four new regions. The new regions are formed by each of the faces of the original element and the newly created vertex. The newly created vertex can be classified only on the interior. This operation is not used very commonly (Figure A.3b).

Figure A.3: (a) Face split on model boundary. (b) Region split.
A.4 Edge Swap

The edge swap is a reconnection procedure that effectively deletes an edge and its connected elements and retriangulates the polygon or polyhedron without the deleted edge. For triangular meshes, the swapping procedure consists of deleting the edge and its two connected faces, and reconnecting the quadrilateral so formed with an edge between the opposite face vertices of the deleted edge (Figure A.4). The process is more involved for volume meshes and consists of the following steps in general (Figure A.5):

1. Delete the regions connected to the edge.
2. Delete the faces connected to the edge.
3. Delete the edge. At this point we have a polyhedral cavity with the two vertices of the deleted edge opposite to each other (not connected by an edge).
4. Create any boundary faces necessary if the swapped edge is a boundary edge.
5. Find the set of edges that are not connected to the vertices of deleted edge. These edges form the boundary of a closed polygon.
6. Triangulate this polygon. Since the polygon does not contain the vertices of the original edge, the original edge cannot be recreated.
7. Connect each face of the polygon to each vertex of the deleted edge to form a region.

Figure A.4: Edge swap for surface meshes.
If there are $n$ connected regions around an interior edge, then a $n$ vertex polygon (excluding the edge vertices) is formed by deletion of these regions. This polygon can be triangulated in $N_n$ ways, $N_n = \sum_{i=3}^{n} N_{i-1}N_{n+2-i}$ with $N_2 = 2$ [18]. Each triangulation has $n - 2$ triangles and therefore, swapping this edge produces $2(n-2)$ tetrahedra. If the edge is classified on a 2-manifold model face (Figure A.6a), then there is only one configuration for the new boundary edge. This new edge is on the boundary of the retriangulation polygon. The number of vertices in the polygon is $n + 1$ where $n$ is the number of regions deleted to form the polyhedral cavity. The number of triangles formed in the polygon are $n - 1$ and therefore the swapped configuration has $2(n - 1)$ tetrahedra.

![Figure A.5: Edge swap in the interior of a volume mesh.](image)

Swapping an edge on a non-manifold face, on the other hand, requires a more careful look. Since the edge is on a non-manifold face, the new boundary edge can be created only between two vertices classified on the closure of the face (Figure A.6b). Also, because the swapped edge had a connected set of regions completely surrounding it, the polygon that needs to be retriangulated has $n$ vertices where $n$ is the number of regions connected to the edge. Therefore, the $n$ vertex polygon is divided into two polygons with $n_1 + 1$ and $n_2 + 1$ vertices respectively where $n_1$ and $n_2$ are the number of regions connected to the edge on the two sides of the non-manifold model face. The number of regions formed is still $2(n - 2)$ but the number of topologically possible triangulations is reduced.
Figure A.6: Edge swap on boundary of volume mesh. (a) Edge swap on 2-manifold model face. (b) Edge swap on non-manifold boundary face.

Not all of the different triangulations possible topologically in an edge swap operation may be geometrically valid. Therefore, each triangulation must evaluated to ensure that all the created elements will be valid (See Section A.7).

The topological constraints in an edge swap are as follows:

1. An edge classified on a model edge may not be swapped since this will cause the mesh to violate topological compatibility with the model.

2. An edge classified on a model face may be swapped with the restriction that the quadrilateral cavity formed on the boundary is retriangulated in the only other way possible.
A.5 Edge Collapse

Edge collapsing is the process of deleting a vertex from the mesh while keeping the mesh geometrically and topologically valid. Conceptually, edge collapsing can be thought of as the process of deleting all the elements connected to the vertex to be removed and retriangulating the resulting polygon or polyhedron. In actual implementations, it is more efficient to carry out a collapse by the following steps (Figure A.7):

1. Delete the regions around the edge to be collapsed including the edge itself.
2. Merge the vertex to be removed with the vertex to be retained.

Figure A.7: Edge collapse. (a) Collapse on surface mesh. (b) Collapse in volume mesh.
3. Merge the entities of the polygon or polyhedron connected to the vertex to be removed with the entities of the connected to the vertex to be retained.

Since the shape of the elements connected to the vertex to be removed changes after the collapse, they must be checked for geometric validity.

The topological restrictions on collapsing are the most stringent of all local mesh modification operations since they have the potential to cause topological incompatibility of the mesh with the model and also cause dimensional reduction of the mesh ([19]). The conditions under which an edge can be collapsed are as follows:

1. If the two vertices of the edge are classified on equal order entities then the two entities must be the same and the edge must be classified on the same entity.

2. If the two vertices are classified on different order model entities,
   
   (a) The vertex to be removed must be classified on a higher order model entity than the vertex to be removed.

   (b) The edge must be classified on the higher order entity.

3. If the two vertices of the edge to be collapsed are connected to two edges sharing a third vertex, then the three vertices must bound a face classified on the same entity as the edge to be collapsed. If this condition is not satisfied, there will be coincident edges in the mesh after the collapse.

4. (For volume meshes only) If the two vertices of the edge to be collapsed are connected to two faces sharing a common edge, then the two edge vertices and the common edge must bound a region. If this condition is not satisfied, there will be coincident faces in the mesh after the collapse. In addition, both faces should not be classified on model faces or else their collapse will cause a dimensional reduction.

A.6 Node Repositioning

Node repositioning is commonly used to improve element quality and mesh gradation in the mesh [20, 21, 36]. The node repositioning criteria used in this thesis
are improvement of mesh gradations by weighted Laplacian smoothing and equidistribution of nodes through the thickness. The discussion of node repositioning here focuses on the considerations in repositioning of a node from the current location to a target location particularly on model boundaries.

Reposition a node classified in the interior of a model is a straightforward process. The node is attempted to be moved from its current location to the target location subject to constraints on element validity or quality. If these constraints are violated, then the node is attempted to be moved to the midpoint of the line joining the current and target locations. This process of bisection continues until a valid target location for the node is found with a limit on the number of bisections (typically 3 to 5).

Repositioning of nodes on model faces is done differently for model faces with a continuous and discontinuous (in particular, periodic) parametric spaces. If the initial move on model face with a continuous parametric space fails, the process of bisecting the line segment between the current and original target locations is done in the parametric space. The midpoint of the current and original target parameter locations is picked iteratively as the next target location. Given the target parametric location, the location of the node on the surface in real space is computed and the local mesh checked for validity. On the other hand, if the face is periodic, the line segment joining the current and target locations may cross the periodic jump in the parametric space. Using an average of the two parameter values to compute a new target location gives erroneous results and results in the point being pulled to a location diametrically opposite to the desired location in real space. To account for this, the points on the face corresponding to the average parameter and the average parameter added with half the parametric range are computed. Of the two locations, the one closest to the current location is chosen. Note that this is equivalent to doing a closest point search on the model face but is considerably more efficient. The underlying assumption of this strategy is that no edge in the mesh spans more than half the parametric space of the model face.

Repositioning of nodes on model edges is similar to the repositioning on model faces except that only one parameter needs to be dealt with.
A.7 Element Validity

The geometric validity of elements in a volume mesh can be easily checked by checking if all the elements in the mesh have positive volume. However, an equivalent check is harder to define for a surface mesh. While it is simple to check whether a triangle has zero area or not (if necessary within some tolerance) checking whether the triangle has “positive” or “negative” area is poorly defined for general three-dimensional surfaces. Schroeder and Shephard [58, 66] defined rigorous conditions for the validity of meshes and in particular imposed the condition that the mesh should be geometrically similar to the model with reference to an appropriately defined parametric space. This means that in that chosen parametric space no elements can overlap each other. However, how to choose an appropriate parametric space such that a mesh that is geometrically similar to the model in that space results in an acceptable mesh in the real space is still an open question.

Violation of geometric similarity of the mesh of a curved surface results in a large difference between the “smoothness” of the discretization of surface relative to the local smoothness of the surface itself. This discrepancy in the “smoothness” of the discretization may be approximated in a number of ways, some of which are listed below:

1. Measure the difference between the mesh face normal and the model face normal sampled at a suitable point within the parametric boundaries of the mesh face. This is an error prone check, particularly for coarse discretizations of highly curved surfaces, since the model and mesh face normals might differ significantly.

2. Construct a local parametric space by projecting all the faces in the local neighborhood onto a plane. The projection plane may be defined by the model face normal or by an appropriate average of the mesh face normals. The projected triangles can be checked for inside-out condition or negative area with respect to this plane. This method is sensitive to the choice of the projection plane and it is very easy to perceive a triangle as having “negative” area due to sharp changes in the mesh face normals.
3. Measure the dihedral angles between mesh faces to determine if the surface discretization is folded thereby causing a large “change” in the mesh face normals when the surface normals itself are not changing that dramatically. The advantage of this measure over the first method, is that it does not rely on comparison of the mesh and model face normals avoiding some of the problems with coarse meshes. Its advantage over the second method is that the extent of the approximation and therefore the possibility of an undesirable decision is much lesser since only two mesh faces are involved at any time in the check. The dihedral angle check for surface “smoothness” involves setting a tolerance for the allowable angles and is dependent on how strictly one wants to control the mesh generation or modification procedures.

In the context of mesh modification procedures, a meaningful alternative to checking the absolute validity of the surface mesh is to check if the modified mesh deviates considerably from the original. This ensures that given a good discretization of the model to start with, each local mesh modification procedure in itself does not cause drastic changes in the mesh. In fact, such a criterion may also be used to preserve important geometric features in the initial discretization. In particular, edge swapping, edge collapsing and node repositioning are prone to the problem of eliminating geometric features in the surface mesh and the above criterion can be successfully used to preserve them [13, 14].

Another important consideration in surface meshing and surface mesh modifications is the self-intersection of the mesh. While a self-intersecting surface mesh in itself is not a problem, it may be unusable in the context of using it as the boundary of a volume mesh. Therefore, procedures to ensure that mesh is not self-intersecting are necessary. One such procedure is presented in [13, 16] and is found to work well. This procedure is based on the assumption that in the limit of refinement the mesh cannot self intersect is the model is not self intersecting.