Selective Refinement Of Progressive Meshes Using Vertex Hierarchies

Yigang Wang[a], Bernd Fröhlich[b] and Martin Göbel[c]
[a,c]Fraunhofer-Institut für Medienkommunication, Germany
[b]Bauhaus-Universität Weimar, Germany
yigang.wang@imk.fraunhofer.de

ABSTRACT

This paper presents a general method for selectively refining or coarsening progressive meshes. Our method works for arbitrary meshes, deals with non-manifolds, and allows the change of manifold and genus properties of the mesh. We introduce new definitions for edge contraction and expansion operations, which make solely use of the vertex hierarchy. These definitions in conjunction with their legal conditions lead to many provable properties of the refinement process. In addition, we propose a locally re-stripping method to dynamically generate the strips during selective refinements, which efficiently forms strips for meshes coarsening or refining process.
Keywords: Selective refinement, Level-of-detail, Mesh simplification, Triangle strips, Progressive meshes.

1. INTRODUCTION

The PM approach represents an arbitrary mesh through a sequence of transformations operating on a base mesh (coarsest resolution). Almost all simplification methods, e.g. [1]-[5], allow the generation of such PM representations. Among them, the method based on quadric metrics and edge collapses gives a good balance between speed and accuracy. By introducing new definitions of the transformations and dependencies, the PM representations are developed into selective refinement method [6]. However, previous methods for selectively refining progressive meshes based on edge collapse mostly correspond to simplification that keeps the topological type, and cannot deal with meshes' topological changes such as a genus change of the surface or changes of the manifold property. For simplification with topological type kept, if the original model is topologically complex, the simplified base mesh may still have numerous triangles. As the result, some small pieces and neglectable gaps still exist whereas the model has been greatly simplified (refer to [11]). Popovic and Hoppe [11] propose a simplification that can change the topological type of the original mesh. Unfortunately, their method is time-intensive, needs tens of hours to process a medium complex model. Garland [4] presents a much fast simplification method that can deal with the case where the transformation changes the topological type of the original mesh by introducing virtual edges. However, no selective refinement method corresponding to such general simplification has been presented.

Another issue is that the underlying theory for selective refinement of progressive meshes is not yet fully developed. Hoppe [6] proposes a method without proofed theoretical background. Xia et al [8] and J.El-Sana et al [10] also propose dependencies conditions for selective refinement, however, these definitions and conditions cannot be used for our general case where meshes may be non-manifold or topological type may change.

Triangle strips are a widely used hardware supported mechanism to compactly represent and efficiently render triangle meshes. Hoppe[7] propose a method to re-strip the whole mesh each time when the mesh changed. As the cost of re-stripping highly depends on the complexity of the model, the re-stripping process may greatly compromise the frame rate if the mesh is very complex. El-Sana et al. [12] propose a data structure, skip strips, that efficiently maintains triangle strips during refinement. However, their method is not equally efficient when mesh become coarser, because edge collapse usually causes strips to be split into small ones (Figure 7).

In this paper we address these problems and present a new framework for selectively refining and coarsening progressive meshes. Our method is based on a vertex hierarchy and deals properly with non-manifold meshes and complex topological changes. The vertex hierarchy is given through the edge contraction operations of the simplification process [4], which generates the progressive mesh representation. In the paper, we first present new definitions of contraction and expansion, which, in contrast to the definitions in previous work [6][7], can properly represent changes of topological type even for non-manifold meshes. Second, we present legalities for selective refinement, which have a set of formally proofed properties. These properties form the theoretical basis for our selective refinement algorithm. In particular interesting is, that only faces from the original simplification process are used to form adaptive meshes, which guarantees a certain mesh quality consistent with the original simplification process.

To the problem of stripping, we propose a locally re-stripping method, which only re-strips newly changed faces. Obviously, it is faster than entirely re-stripping method like Hoppe's. Moreover, it can equally efficiently generate strips whenever mesh becomes coarser or finer (Figure 7).

2. RELATED WORK

Xia and Varshney [8] use *ecol/vsplit* transformations to create a simplification hierarchy that allows real-time selective refinement. Hoppe [6] proposes an approach that forms the vertex hierarchy by an unconstrained, geometrically optimized sequence of *vsplit* transformations. However, these methods keep the topological type, not deal with non-manifold meshes, even their definitions of the vsplit/ecol cannot represent topological type change.

Garland's method [4] is most closely related to our method. Recent version of *qslim* can make non-manifold changes. However, it is just a simplification process. In this paper, we present its inverse process, selective refinement algorithm, mainly based on his work.

J. El-Sana and A. Varshney [10] propose implicit dependencies for view-dependent simplification, which is the most compact representation of the dependencies for view-dependent simplification. However, their method is same as Xia's, cannot deal with generic non-manifold cases.

3. CONTRACTION AND EXPANSION

The expansion of a vertex v_s (Figure 1) generates two new vertices v_p and v_q. In a topological sense, v_p can be considered identical to v_s, and only v_q is a new vertex. All the directed edges surrounding v_s stay unchanged. Each of these edges corresponds to a neighbor face of v_s. The expansion turns some of these faces into neighbor faces of v_p, the others into neighbor faces of v_q. The corresponding edge sets for these two face sets are denoted as E_p and E_q respectively. In addition, the expansion also generates some new faces, which share the vertices v_p and v_q. We denote all the directed edges that connect v_p and a third vertex (except v_p and v_q) on new faces as a set E_d. For instances, in Figure 1,

$E_p = \{\overrightarrow{v_3v_4}, \overrightarrow{v_4v_5}\}$,

$E_q = \{\overrightarrow{v_0v_1}, \overrightarrow{v_1v_2}\}$ and $E_d = \{\overrightarrow{v_pv_o}, \overrightarrow{v_2v_p}\}$.

According to these notations, we give the following definitions.

Definition 1.

An *expansion*$(v_s, v_p, v_q, E_p, E_q, E_d)$ makes the following transformations.

a) Replace v_s by v_p, and add vertex v_q.

b) For each face that contains v_s and an edge from E_p, remap v_s to v_p.

c) For each face that contains v_s and an edge from E_q, remap v_s to v_q.

d) Add new faces that contain v_q and an edge from E_d.

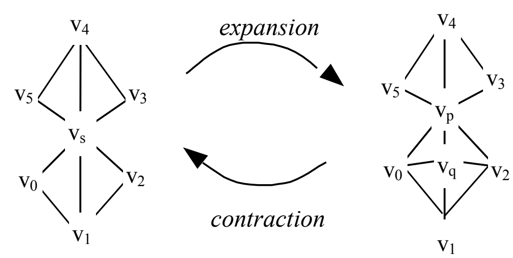


Figure 1: An expansion and its inverse, a contraction.

Definition 2.

A *contraction*$(v_s, v_p, v_q, E_p, E_q, E_d)$ makes the following transformations**.**

a) Remove all the faces containing v_q and an edge from E_d.

b) For each face that contains v_q and an edge from E_q, remap v_q to v_s.

c) For each face that contains v_p and an edge from E_p, remap v_p to v_s.

d) Replace v_p with v_s, and remove v_q.

Obviously, an expansion is the inverse of a contraction. In a simplification process, an arbitrary mesh M^* is decimated through a sequence of n edge contractions, $c_0, c_1, \cdots, c_{n-1}$ into the base mesh M^n. The simplification process can be represented in the following way:

$M^* = M^0 \xrightarrow{c_0} M^1 \cdots \xrightarrow{c_{n-1}} M^n$

Such process can be reversed, because each contraction has an inverse. The base mesh M^n can be transformed into the mesh M^* through a sequence of n edge expansions, $e_{n-1}, e_{n-2}, \cdots, e_0$, where $e_i = c_i^{-1}$. The tuple

$\{M^n, (e_{n-1}, e_{n-2}, \ldots, e_0)\}$

is a PM representation of M^*. For each expansion, an old vertex is split into two new vertices, which can be looked as a parent-child relationship. The vertices of a PM representation form a vertex hierarchy. (Figure 2).

For our application the mesh needs to vary locally and incrementally among the different resolutions. The transformations are in general performed in a different order than the original PM sequence, and they include both contraction and expansion. There are transformation sequences, which do not result in a valid mesh. We want our transformation sequences to produce valid meshes, and also require that any mesh is consistent with the local modifications from the original simplification process to ensure the same quality for the faces.

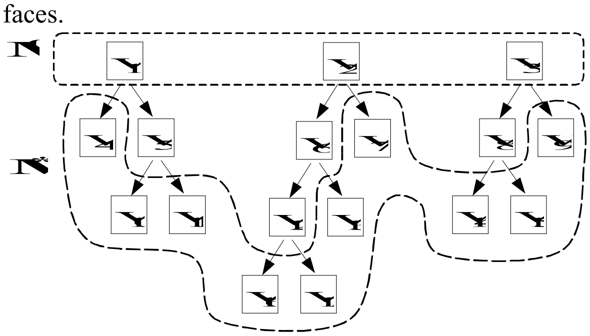


Figure 2: A vertex hierarchy (vertex forest) corresponds to a PM representation. In the vertex hierarchy, the root nodes are the vertices of the coarsest mesh, and the leaf nodes are the vertices of the finest mesh.

Contraction and expansion transformations must obey some legality conditions to make each modification valid and consistent with the PM generation stage. Let tuple

$\{M^n, (e_{n-1}, e_{n-2}, \ldots, e_0)\}$

be a progressive mesh with $c_i = e_i^{-1}$. A selective refining or coarsening process can be represented by

$P^0 = M^n \xrightarrow{t_0} P^1 \cdots P^k \xrightarrow{t_k} P^{k+1} \cdots$.

t_k is a legal contraction or expansion. The actual mesh in this process is in general different from in the meshes created during the PM generation stage. The vertices on the current mesh are called *front vertices*. We denote the set of *front vertices* as F. If a vertex is on the current mesh, it is called *active vertex*, otherwise it is called *inactive vertex*. We call all the faces lying on the current mesh *active faces*.

Legality conditions for expansion and contraction transformations:

An *expansion*($v_s, v_p, v_q, E_p, E_q, E_d$) is legal if v_s is active, and all the vertices contained in E_p, E_q and E_d are active.

A *contraction*($v_s, v_p, v_q, E_p, E_q, E_d$) is legal if v_p and v_q are active, all the vertices contained in E_p, E_q, and E_d are active, and

num($N(v_p)$) + *num*($N(v_q)$)

= *num*(E_p) + 2 *num*(E_d) + *num*(E_q)

$N(v)$ represents a set of neighboring faces of a vertex v. *num(X)* is the number of elements in a set *X*. This condition assures that the number of neighboring faces of the vertices v_p and v_q on the current mesh is equal to the number of neighboring faces during the contraction operation of the original simplification process. Note that each edge in E_p, E_q, and E_d correspond to a face, and the sets of neighboring faces of v_p and v_q overlap.

Properties

a) Each local modification in the selective refining process is exactly the same (or the corresponding inverse) as in the original simplification process.
b) If $v_0 \in F$ and the activity of vertex v_1 is a necessary condition for the legality of the expansion of v_0, then v_1 must be on F or below F in the vertex hierarchy.
c) If $v_0 \in F$, and the activity of vertex v_1 is a necessary condition for the legality of the contraction of v_0, then v_1 must be on F or above F in the vertex hierarchy.
d) If there is a vertex in F, which is not a leaf node, there must exist a vertex $v_0 \in F$, whose expansion is legal.
e) If there is a vertex in F, which is not a root node, there must exist a vertex $v_0 \in F$, whose contraction is legal.

These properties have been formally proved by us. Property 1 ensures that our selective refining or coarsening process is consistent with the original simplification process. Properties 2 and 3 make the recursive refining and coarsening feasible. Properties 4 and 5 ensure that the refining (coarsening) process is able to reach the finest (coarsest) mesh.

Note that our legalities are mostly similar as Xia's [8], the apparent difference lies in the legality of contraction. This results from the generality of our selective refinement, which may change the mesh's topological type or deal with non-manifold mesh. Only neighboring vertices cannot guarantee that the neighboring faces are same as that in simplification stage because the neighbors of a vertex may include more than one separate part.

For manifold mesh with topological type kept, Hoppe's legal conditions [6] allow more flexible contraction or expansion operations than ours. However, local modifications may occur in his method, which are different from the original simplification process. This might not be desirable for certain application domains. In particular, the quality of newly generated faces may not be equivalent to those created during the simplification process. In addition, unexpected cases may happen, such as normal flipping etc. In our method, we keep the local modifications completely consistent with the original simplification process, so the quality of the mesh can be kept. Moreover, our definitions of the expansion and contraction are more general than his, which can express the contraction of any edge in the mesh, even virtual edges, which are used for topology simplification [4].

4. SELECTIVE REFINING OR COARSENING

Our application, the interactive visualization of geo-scientific data, requires the rendering of high resolution polygonal surfaces. A typical surface consists of approximately 500000 triangles and there are often a few of these surfaces of interest. It is clear that these surfaces need to be drastically simplified to achieve high frame rates. Users are often interested in local details in certain regions of the model. In our system, they locally refine their model in these regions until they are satisfied with the resolution. Our users specify the region of interest by positioning a sphere with adjustable radius in the region of interest. The vertices inside the sphere are expanded or contracted.

We use Garland's simplification method (qslim) [4] to create the PM representation for arbitrary meshes. Our method creates a vertex hierarchy and all the expansion transformations defined in Section 3 from this PM representation (note: contraction and expansion have the same parameters). According to our definitions in section 3, the legality of transformations for a certain vertex on the mesh can be easily checked.

The coarsening process happens in two ways, either "naturally" or "forced". The natural contraction contracts a selected vertex if its contraction is legal. The forced contraction works recursively until the selected vertex is legally contracted. The vertices required for the legality of the contraction are made active by recursively contracting their children. The expansion transformation works in a similar way. The properties (2) –(5) provide the theoretical background for these recursive methods. They guarantee that the recursive process cannot be trapped in a dead lock.

5. LOCAL RE-STRIPPING

Our local Re-Stripping process mainly includes two stages, collecting newly changed faces and re-stripping these faces. During a serial of contractions or expansions, the mesh contains two kinds of strips, one kind of strips has not changed, and the other kind of strips has changed as some faces are added or removed, our algorithm incrementally collects the faces whose strips have changed. After the serial of transformations and before the resulted model are sent to the rendering pipeline, our algorithm re-strips the collected faces.

Collecting newly changed faces. This process is done incrementally. Obviously, a transformation at least influences the strips that correspond to newly added (removed) faces and their neighbors. In order to make the re-stripped faces as less as possible, we only consider such strips. Note that the collected faces must be kept on the current mesh, if some faces are deleted, the corresponding faces in the collected face arrays also should be deleted.

Re-stripping the local part of the mesh. Our re-stripping method is similar as the method in [13], except that our algorithm deals with the part of the mesh instead of the whole model. The algorithm first chooses the seed triangle with the least number of neighbors as the starting triangle of a strip, and then greedily extends the strip as possible as it can. If there are more than two triangles that can append to the strip, the algorithm always choose the one with less neighbors.

6. RESULTS

The progressive mesh representations are generated using the qslim simplification software[4]. We have integrated the proposed algorithm into our virtual environment framework Avango (Previously called Avocado)[9]. The generation of adaptive meshes happens in a separate process and the results are passed on into the rendering process. This approach avoids that the graphics pipeline stalls while we are updating our meshes. In our application, users manually specify the area of interest and specify if the mesh should be refined or coarsened.

We demonstrate our method is able to selectively refine a model that is topologically simplified in Figure 5. The change of topological type can simplify the topology of the original model, however, it usually cause the mesh to become non-manifold. All such cases are considered in our method, so our method is very general, allow any pair of vertices contract.

Figure 6 shows some pictures from a sequence of refining and coarsening operations. For each refine or coarsening operation all the affected vertices are forced to contract. The images clearly show that for these cases the mesh around the selected area also needs to adapt to make the contraction or refinement of the selected vertices legal.

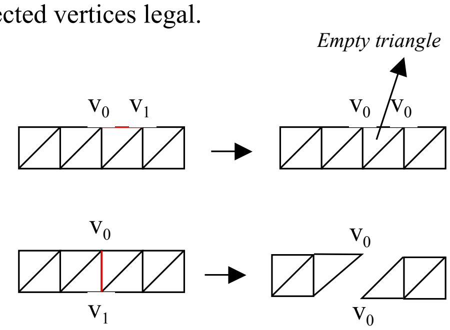


Figure 3: Without re-stripping, contracting v_0 and v_1 usually causes an existed strip to be split into small ones or generates swap faces in the strip.

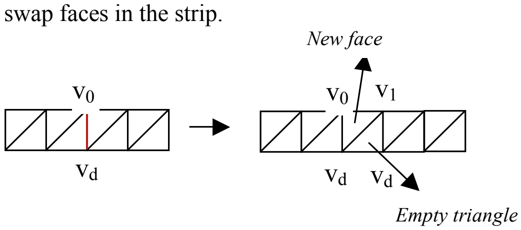


Figure 4: Without re-stripping, expanding v_0 usually generates swap faces in the existed strips.

We compare our locally re-stripping method with the method proposed in [12], the results of a model through a same sequence of contraction operations are shown in Figure 7. It shows that our stripping method is much better than the method in [12] in case the mesh is contracted. The method in [12] can efficiently form strips when the model becomes finer, however, it cannot form strips in same efficiency way when the model becomes coarser. The main reason is that old strips they try to re-use often need to be split into small ones if a contraction happens (see Figure 3). On the contrary, our method abandons changed old strips, and re-strips them. Experiments show that our stripping method form equally efficient strips whenever the model are refined or coarsened. In addition, our method generates strips without swapping faces, there are no extra empty triangles for OpenGL rendering [14]. In contrast, the method in [12] often generate strips with swapping faces, bring extra empty triangles for OpenGL rendering (see Figure 3 and 4). From our experiments, the swapped triangles in their method usually occupy 30-40%, or even more, this greatly compromises the performance of their algorithm sometimes.

7. CONCLUSIONS

In this paper, we have carefully studied the theoretical background of locally and incrementally refining or coarsening progressive meshes. We present new definitions of contraction, expansion, and their preconditions. These definitions result in proofed properties of the refinement process and provide a sound basis for our method. We can deal with any transformations for manifold or non-manifold meshes, and even with transformations that introduce complex topology changes. In addition, our method keeps the refining or coarsening process exactly consistent with the original simplification process, so that the properties of locally refined meshes is mostly determined by the original simplification process.

Our second contribution is that we propose a locally re-stripping method, which equally efficiently generates strips whenever the model is refined or coarsened.

REFERENCES

1. Hugues Hoppe. Progressive Meshes. Computer Graphics(SIGGRAPH'96 proceedings), pages 99-108, 1996.
2. P.S.Heckbert and M.Garland, Survey of Polygonal Surface Simplification Algorithms. Tech.report, Carnegie Mellon University, 1997.
3. M.,Garland and P.S.Heckbert. Surface Simplification Using Quadric Error Metrics. Computer Graphics(SIGGRAPH'96 Proceedings), pages 209-216, 1996.
4. Michael Garland. Quadric-Based Polygonal Surface Simplification.Ph.D.Dissertation. http://graphics.cs.uiuc.edu/~garland/CMU/thesis/.
5. Hugues Hoppe. New Quadric Metric for Simplifying Meshes with Appearance Attributes. IEEE Visulization'99 proceedings.
6. Hugues Hoppe. View-Dependent Refinement of Progressive meshes. Computer Graphics (SIGGRAPH’97), pages 189-198, August 1997.
7. Hugues Hoppe, Smooth View-Dependent Level-of-detail Control and its Applications to Terrain

Rendering. Proceedings of IEEE Visualization'98, pages 35-42, October 1998.
8. J. C. Xia and A. Varshney, Dynamic view-dependent simplification for polygonal models. In Visualization'96 Proceedings (1996), IEEE, pages 327-334.
9. H. Tramberend, Avocado: A Distributed Virtual Reality Framework. Proceedings of VR'99 Conference, Houston, Texas, pages 14-21, March 1999.
10. J.EI-Sana and A. Varshney, Generalized View-Dependent Simplification, in the proceedings of EUROGRAPHICS'99.
11. Jovan Popovic and Hugues Hoppe, Progressive Simplicial Complexes, SIGGRAPH'97, page 217-224.
12. J. El-Sana, E. Azanli, A. Varshney, "Skip Strips: Maintaining Triangle Strips for View-dependent Rendering", In IEEE Visualization 1999, pages131-138.
13. K. Akeley, P. Haeberli, and D. Burns, tomesh.c : C Program on SGI Developer's Toolbox CD, 1990.
14. F.Evans, S.Skiena and A.Varshney, "Optimizing Triangle Strips for Fast Rendering", in IEEE Visualization'96 Proceedings.

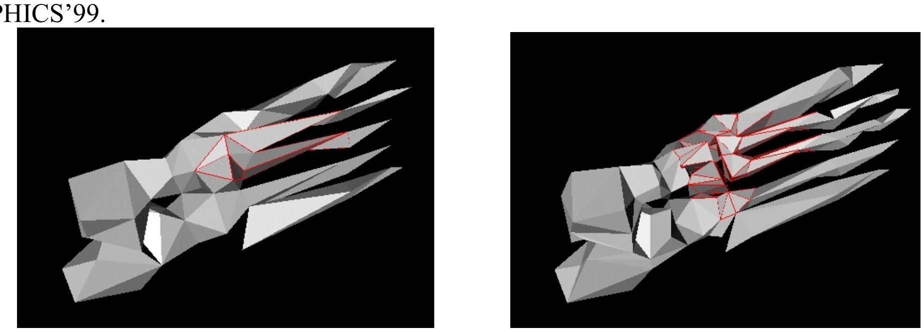


Figure 5: A sequence of refining operations that change the mesh's topological type, the connected parts become separated when the mesh become finer.

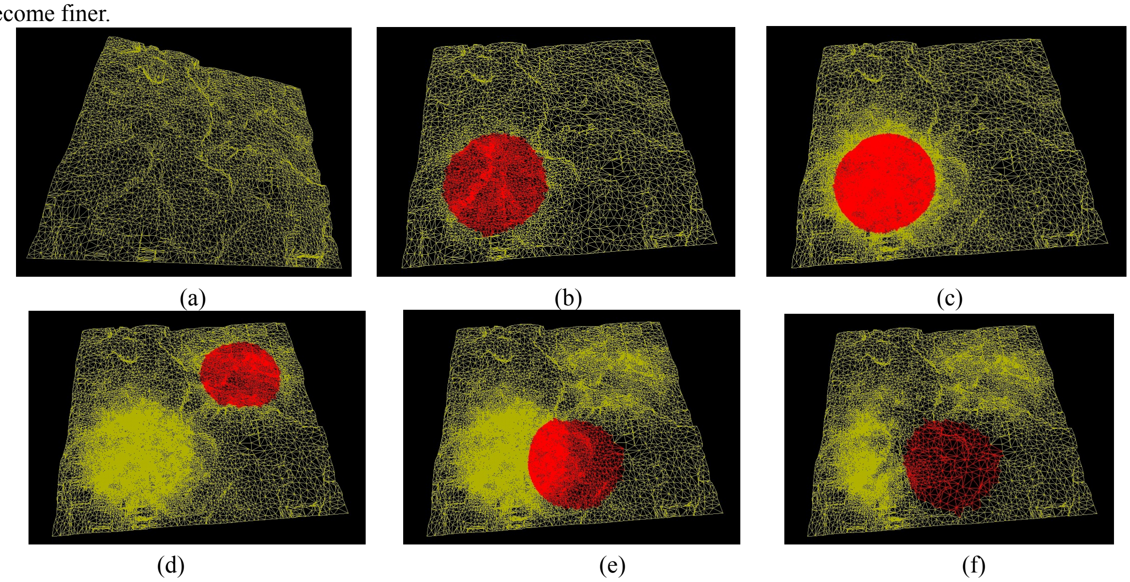
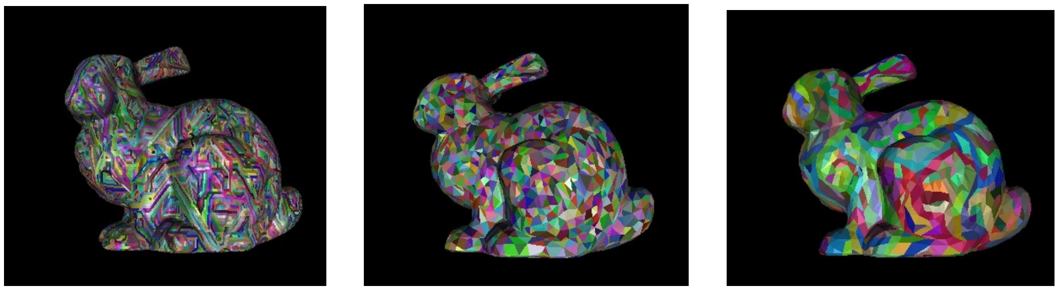


Figure 6: A sequence of refining and coarsening operations from our geo-scientific application showing a complex subsurface structure. The parts drawn in red correspond to the areas of interest. (a-c) A sequence of refinement operations starting from the coarsest mesh. (d) another area is additionally refined. (e-f) The density of the mesh is reduced. (f) shows the effect of forced contractions in the vicinity of the area of interest.



(a) (b) (c)

Figure 7: Comparing our local re-stripping method with the method in [12]. (a) The finest model is stripped using the method in [13]. The models in (b) and (c) are obtained by the model in (a) changed through a same sequence of contractions. The strips in (b) are generated by our implemented method as skip strips in [12], while the strips in (c) are produced by our local re-stripping method.