HHS Public Access

Author manuscript

Procedia Eng. Author manuscript; available in PMC 2016 August 23.

Published in final edited form as:

Procedia Eng. 2015 ; 124: 187–199. doi:10.1016/j.proeng.2015.10.132.

Quality Partitioned Meshing of Multi-Material Objects

Qin Zhang[a], **Deukhyun Cha**[a], and **Chandrajit Bajaj**[a,*]

[a]Institute of Computational Engineering and Sciences, Computational Visualization Center, University of Texas, Austin, TX 78712

Abstract

We present a simple but effective algorithm for generating topologically and geometrically consistent quality triangular surface meshing of compactly packed multiple heterogeneous domains in $\mathbb{R}^3$. By compact packing we imply that adjacent homogeneous domains or materials share some 0, 1, and/or 2 dimensional boundary. Such packed multiple material (or multi-material) solids arise naturally from classification/partitioning/segmentation of homogeneous domains in $\mathbb{R}^3$ into different sub-regions. The multi-materials may also represent separate functionally classified sections or just be multiple component copies tightly fused together as perhaps by layered manufacturing processes. The input to our algorithm is a geometric representation of the entire multi-material solid, and a volumetric classification map identifying the individual materials. As output, each individual material region is represented by a triangulated 2-manifold boundary, with adjacent material regions having shared boundaries. Our algorithm has been implemented, and applied to different multi-material solids, and the results are additionally presented with quantitative analysis of detection and cure of non-manifold interfaces as well as spurious small components. These meshes are useful for combined boundary element analysis, however these simulation results are not presented.

Keywords

Heterogeneous domains; Multi-material regions; Geometric flow; Manifold surfaces; Quality meshing

1. Introduction

Heterogeneous domains are at times referred to as solids made of different constituent materials (aka multi-material objects). When the solid is made of continuously varying material compositions, it is classified as functionally gradient materials (FGM) (see e.g., [1,2]), and usually exhibits multiple and mixed mechanical, electrical, acoustic properties. A discontinuous change in material compositions however generates heterogeneous regions of distinct material types in the solid, and are often classed as a multi-material object (MMO). In the review paper [3], Kou and Tan list a few typical MMOs: wear resistant coatings, solid oxide fuel cells, dental implants, bone implants, and so on. Elucidating properties of such

*Corresponding Author, bajaj@ices.utexas.edu.

MMO's through simulations motivates our problem of generating quality meshing of compactly packed multiple heterogeneous domains.

More formally: Given any geometric representation of the MMO, and additionally a 3-dimensional (3D) material voxellized classification map identifying the individual materials, we generate a quality triangular manifold surface mesh boundary representation for each individual material region. If two different material regions are adjacent, their corresponding meshes would share some 0-dim, 1-dim and/or 2-dim boundary. Each triangular mesh would be a 2-dimensional manifold, aka a 2-manifold (see Definition 4.1). The 3D material classification map, as is the case when MMOs are obtained from imaging data, while providing a voxellized partition of the entire MMO domain may suffer from non-manifold vertices (0D) and edges (1D) [3–7]. Our additional contribution, thus includes a scheme to detect and fix all potential non-manifold cases from the voxellized classification map, including detection and removal of small isolated components. Finally, after post processing with the constrained geometric flow optimization scheme we provide quality multi-material triangulated meshes. Fig. 1 shows various stages of our MMO processing and generation of quality triangular manifold surface meshes.

The rest of the paper is organized as follows. In Section 2, we briefly review related prior work. In Section 3, we formalize the sub-problems and give a sketch of the algorithm steps. Specific details of each step are given in Section 4. Experimental results of our implementation are given in Section 5, including summary quantization statistics of non-manifold detection, and cure. This statistics further demonstrates the efficacy of our solution.

2. Related Prior Work

We focus our review here on only the geometric meshing aspects of heterogeneous domains and skip the material modeling that has also been reviewed in [3]. The prior MMO domain meshing methods include voxel-based [8–12], Delaunay refinement based [4,5,13], finite element-based [7,14,15], mesh feature based [16,17], and layer-based [18, 19] techniques. A systematic approach to MMO solid modeling was proposed in [20] based on r_m sets and r_m classes. A 3D solid is subdivided into components made of unique materials and a non-manifold Boundary representation (B-rep) is used to model such objects. Each component is homogeneous and has an assigned index of material. Chiu and Tan [21] developed a material tree structure to store different composition of an object. The material tree was then added to a data file to construct a modified STL file format suitable for rapid prototyping. The MMO meshing approach implemented in CGAL is from [5,13] and uses conforming Delaunay refinement based on methods of [4,22] to capture sharp features. None of these prior schemes guarantees manifold meshes for arbitrary voxellized MMO domains, and so is one of the principal results of this paper.

For the single material object, isocontouring is usually used to extract the mesh representation for efficient visualization and computation. The most common isocontouring scheme is marching cube method proposed in [23]. This single-material contouring scheme was expanded to multi-materials case by [8] and later [24], named as the M3C algorithm. After a smoothing step, the extracted multi-material meshes look nice but volume shrinkage

issues inherent in their methods cannot be usually ignored. Also unexpected gaps may exist between the different material boundaries. In [25], Wang extracted surface meshes from multi-material volumes by three steps: a combined coarse patch extraction, signed distance fields construction and an adaptive remeshing process. It is reported in the paper that the extracted meshes are 2-manifold with high quality. Wang and Feng [26] implemented the boundary surfaces of homogeneous objects (BSHO) construction method to extract multi-material meshes on GPU. Shammaa et al. [27] combined region growing and graph-cut methods to classify the volumetric model into its component material domains and adopted a generalized marching cube method to generate triangulated mesh surfaces. In [6], an efficient pipeline is proposed to generate multi-domain quality meshes from volumetric data. The fundamental difference between the current paper and what was reported in [6] is how boundary gaps are treated in MMO domains. The MMO meshes of this paper, if they are adjacent to each other, always share some common boundaries. Even in the case when MMO domains are of non-zero topological genus having tunnels (through holes), the gaps don't exist between boundaries where the background (empty) region is considered as being an independent material. Furthermore, the meshes we generate are also guaranteed to be 2-manifolds, and don't possess spurious and/or miniscule material components.

3. Notation, Problem Statement and Algorithm Sketch

Notation

A *grid* G is composed of *grid points* as $G = \{G_{ijk} = (x_i, y_j, z_k); 0 \le i < G_x, 0 \le j < G_y, 0 \le k < G_z\}$, where (G_x, G_y, G_z) is the *grid dimension*. Without loss of generality, we assume $G_x = G_y = G_z$ and $d_x = x_i - x_{i-1}$ is identical to $d_y = y_j - y_{j-1}$ and $d_z = z_k - z_{k-1}$ for all $(i, j, k) \in (0, G_x) \times (0, G_y) \times (0, G_z)$. The multi-material object is represented by a volumetric representation or a *classification map* $\mathcal{V}$ as

$$\begin{aligned} f \quad & :G \subset \mathbb{R}^3 \to \mathbb{N} \\ & (x_i, y_j, z_k) \to f(x_i, y_j, z_k) = f_{ijk}, \quad i, j, k = 0, \ldots, G_x - 1, \end{aligned}$$

where $\mathbb{N} = \{0, 1, 2, \ldots\}$ is the set of natural numbers with 0 representing the background. If the object is made of N materials, then $0 \le f_{ijk} \le N$, $\forall (i, j, k) \in [0, G_x - 1]^3$ and f_{ijk} is the material *ID* of the grid point G_{ijk}. For a grid point G_{ijk}, we denote by V_{ijk} the *dual voxel* or *voxel*, that is $V_{ijk} = \left[x_i - \frac{1}{2}d_x, x_i + \frac{1}{2}d_x\right] \times \left[y_j - \frac{1}{2}d_x, y_j + \frac{1}{2}d_x\right] \times \left[z_k - \frac{1}{2}d_x, z_k + \frac{1}{2}d_x\right]$. One voxel consists of six faces **f**, twelve edges **e**, eight vertices **v**, and the space enclosed by the six faces. Obviously, the center of the voxel V_{ijk} is the grid point G_{ijk}.

If $f_{ijk} = f_{(i+1)jk}$, we say that voxels V_{ijk} and $V_{(i+1)jk}$ share a face **f**. If $f_{ijk} = f_{(i+1)(j+1)k}$, we say that voxels V_{ijk} and $V_{(i+1)(j+1)k}$ share an edge **e**. If $f_{ijk} = f_{(i+1)(j+1)(k+1)}$, we say that voxels V_{ijk} and $V_{(i+1)(j+1)(k+1)}$ share a vertex **v**. Similar definitions apply to the other allowable cases. Thus, one voxel V_{ijk} has six faces, twelve edges and eight vertices that could be shared with other voxels.

For the classification map $\mathscr{V}$, we can generate a *partition graph* $P^{\mathbf{G}}(\mathscr{V})$, which is the boundary representation of the multi-material object. A partition graph $P^{\mathbf{G}}$ is composed of *partition face set* $P^{\mathbf{f}}$, *partition edge set* $P^{\mathbf{e}}$, and *partition vertex set* $P^{\mathbf{v}}$, that is, $P^{\mathbf{G}}(\mathscr{V}) = P^{\mathbf{f}}(\mathscr{V}) \cup P^{\mathbf{e}}(\mathscr{V}) \cup P^{\mathbf{v}}(\mathscr{V})$. The partition face set $P^{\mathbf{f}}(\mathscr{V})$ is defined as the union of faces with each face shared by two voxels having different material IDs, that is,

$$P^{\mathbf{f}}(\mathscr{V}) = \left\{ \cup \mathbf{f} \middle| \begin{array}{l} \mathbf{f} \in V_{ijk} \quad \text{and} \quad f_{ijk} \neq f_{lmn}, |l-i|+|m-j|+|n-k|=1, \\ i,j,k,l,m,n=0,\ldots,G_x-1 \end{array} \right\}.$$

Similarly, the partition edge set $P^{\mathbf{e}}(\mathscr{V})$ is defined as the union of edges with each edge shared by two voxels having different material IDs, that is,

$$P^{\mathbf{e}}(\mathscr{V}) = \left\{ \cup \mathbf{e} \middle| \begin{array}{l} \mathbf{e} \in V_{ijk} \quad \text{and} \quad f_{ijk} \neq f_{lmn}, |l-i|+|m-j|+|n-k|=2, \\ 0 \leq |l-i|, |m-j|, |n-k| \leq 1, i,j,k,l,m,n=0,\ldots,G_x-1 \end{array} \right\}.$$

The partition vertex set $P^{\mathbf{v}}(\mathscr{V})$ is defined as the union of vertices with each vertex shared by three or more voxels having different material IDs (counting the background as an independent material) that is,

$$P^{\mathbf{v}}(\mathscr{V}) = \left\{ \cup \mathbf{v} \middle| \begin{array}{l} \mathbf{v} \in V_{ijk} \quad \text{and} \quad f_{ijk} \neq f_{lmn}, |l-i| * |m-j| * |n-k|=1, \\ i,j,k,l,m,n=0,\ldots,G_x-1 \end{array} \right\}.$$

Then a *partition mesh* P_t for a particular material ID t is a part of the partition graph $P^{\mathbf{G}}(\mathscr{V})$, and is defined by

$$P_t(\mathscr{V}) = \left\{ \cup \mathbf{f} | \mathbf{f} \in V_{ijk} \cap p^{\mathbf{f}}(\mathscr{V}), f_{ijk}=t, i,j,k,=0,\ldots,G_x-1 \right\}.$$

See for example Fig. 1 for snapshots of our MMO processing pipeline.

Problem statement

Our goal is to generate quality surface meshes $M_i (i = 1, \ldots, N)$, representing the N materials and satisfying the following two criteria:

1. Each mesh $M_i (i = 1, \ldots, N)$ is a 2-manifold.
2.
   - **(a)** If $f_{ijk} \neq f_{(i+1)jk}$, then $H^2(V_{ijk} \cap V_{(i+1)jk}) > 0$ and $H^2(M_{f_{ijk}} \cap M_{f_{(i+1)jk}}) > 0$.
   - **(b)** If $f_{ijk} \neq f_{(i+1)(j+1)k}$, then $H^1(V_{ijk} \cap V_{(i+1)jk}) > 0$ and $H^1(M_{f_{ijk}} \cap M_{f_{(i+1)jk}}) > 0$.
   - **(c)** If $f_{ijk} \neq f_{(i+1)(j+1)(k+1)}$, then $H^0(V_{ijk} \cap V_{(i+1)jk}) > 0$ and $H^0(M_{f_{ijk}} \cap M_{f_{(i+1)jk}}) > 0$.

Here $H^d(A)$ is the d-dimensional Hausdorff measure [28] of a set A, which is intuitively the generalization of the number of points in a finite set ($d = 0$), the length of a curve ($d = 1$), the area of a surface ($d = 2$), the volume of a solid object ($d = 3$), etc. We can describe Criterion 2 succinctly as the collection of generated surface meshes are required to keep the MMO partition topology as specified by the voxellized classification map. That means, if two differently classified voxels share a face, the two different material meshes, and individually containing their respective voxels, share a 2D face. If two differently classified voxels share an edge, the two material surface meshes also share a 1D edge. If three or more differently classified voxels (counting background as independent material) share a vertex, the MMO surface meshes also share a 0D vertex. Furthermore, we can simplify this to say that if two grid points are adjacent in the classification map having different material IDs, the final meshes are *gaps-free*, and share a common boundary (a face, an edge, or a vertex). Given the grid-based representation of the object, we know that one face can be shared at most by two different material meshes, one edge can be shared at most by four different material meshes and one vertex can be shared at most by eight different material meshes.

Sketch of the multi-material meshing algorithm

The input 3D multi-material classification map is often obtained by multi-domain segmentation or classifiers using normalized graph cut operating on a voxellized 3D reconstructed image of the MMO [29,30]. Softwares such as AsymSeg and SymSeg [31,32] implemented and publicly available from VolRover [33] and Segger [34] can be used to generate such material classification maps. Generally, if a face is shared by two voxels with the same material ID i, then the face will be removed and the two voxels are clustered into the same i material region. Otherwise, the face will be a part of the partition mesh P_t and will be partitioned into two different material regions. Unfortunately, the partition meshes $\{P_t\}_{t=1}^{N}$ generated by this simple way usually suffer from the following problems: (a) non-manifold problem, including non-manifold vertices and edges; (b) tiny component problem, like isolated and extremely tiny independent material regions. Our proposed algorithm is:

1. Fix the tiny component problem.
2. Fix the non-manifold problem.
3. Partition the voxel classification map and generate the multi-material mesh.
4. Optimize the generated partition mesh.

More details of each step are provided in sub-sections 4.1, 4.2, 4.3 and 4.4, respectively. When we process our input solid domain, we perform steps 1 and 2 repeatedly and in that order, i.e., we first fix the tiny component problem, and then fix the non-manifold problem. If the non-manifold fix step creates tiny components, we repeat the tiny component fix steps as necessary.

4. Implementation Details

4.1. Fixing the tiny component problem

A straightforward though cumbersome way to remove the tiny regions of the multi-material partitioned meshes, is to collapse them to a neighboring material region, after measuring their respective sizes (usually volume or H^3-measure) relative to the desired partition mesh. Here, as later, we present a method that operates directly on the voxellized classification map to very efficiently remove all isolated tiny material regions. We call our approach as voxel cluster growing (VCG). Based on the continuity of the material ID (including background) of each grid point, we classify them into multiple regions. If the region size is less than a threshold, then we change the material ID of this region with its neighbor. This is done voxel by voxel in this region. For example, for each voxel, we count the duplications of the material ID of its 6 neighbor voxels. We change the material ID of the voxel to the material ID with biggest duplication. This process is iteratively carried out, which means the region size is changed gradually. We utilize a geometric series, that is, the region size threshold is set to be 1, 2, 4, 8, … in sequence. During all our experiments, we find that our VCG method works very effectively in removing tiny individual regions, without leaving any holes or voids.

In Figure 2, we show one example for the Brodmann area data (region 16, see Subsection 5.1). In figure (a), we show the section view of the partitioned mesh, where one can see there are multiple tiny regions, all separately classified. In figure (b), we show the section view after our tiny component removal (cure) process. Note not only are there no tiny regions, the artifacts caused by the density map segmentation have been additionally eliminated. Figure (c) shows a section view of the result.

4.2. Fixing the non-manifold problem

For multi-material surface meshing, Wang and Feng [26] devised a complicated vertex classification and merging lookup table to preserve the 2-manifold feature, but the resulting mesh quality was not very high. Moreover the 2-manifold property is not easy to be preserved when meshes are smoothed. Wang [25] pays attributes to the regularization theory [35] for generating 2-manifold models from non-manifold objects. In [36], several approaches, such as blending and chamfering, sweeping, joining and splitting along edges, planar sectioning, are provided to handle non-manifold modeling. However, these approaches are not suitable for our multi-material cases since these approaches were developed for only one material domain [37].

Definition 4.1—A surface mesh is a 2-manifold if the local neighborhood of every point on the mesh is topologically equivalent or homeomorphic to a single disk.

Intuitively, it means the local neighborhood of every point may be continuously transformed into a single disk by stretching and bending, but without cutting, tearing or gluing. For example, in Fig. 1 (a) the vertex P is a manifold vertex but the vertex Q is a non-manifold vertex, and the edge L is a non-manifold edge. For a mesh with non-manifold vertices or edges, it is often non-trivial to fix or repair it. There are several published approaches to

fixing the non-manifold problem on triangulated surface meshes, such as cutting and stitching [38] and simulated annealing [37]. In this paper, however we first repair (cure) the 3D voxellized classification map by removing all non-manifold vertices and edges, generating a manifold voxellized partition MMO mesh and then generate a manifold MMO surface mesh keeping the same topology as the manifold voxellized partition mesh. One should be reminded that the manifold criterion applies only one mesh at a time. That means, our goal to guarantee each mesh with the same material ID is a 2-manifold. For example, if one edge is shared by four meshes with different IDs (background ID 0 included), then this edge will be treated as manifold edge other than non-manifold edge since this edge is a manifold edge for each material ID. But for one edge shared by two or three material IDs, the edge could be a manifold edge or a non-manifold edge. For example, in Fig. 6 (a), the red edge is shared by meshes with two different material IDs (background ID 0 and blue dot ID) but this edge is a non-manifold edge for mesh with blue dot ID and also a non-manifold edge for the virtual mesh with background ID 0. Similarly, if one vertex is shared by eight meshes with different IDs, then this vertex is definitely a manifold vertex for each mesh. However, if a vertex is shared by two meshes with different material IDs, then this vertex could be a non-manifold vertex or a non-manifold vertex. For example, in Fig. 3 (a), the red vertex is shared by meshes with two different material IDs (background ID 0 and blue dot ID) but this vertex is a non-manifold vertex for blue dot ID mesh but a manifold vertex for the virtual background mesh.

We handle the non-manifold vertex and non-manifold edge problems separately. For the non-manifold vertex case, we detect all the possible cases as shown in Figure 3. Assume the left-bottom grid point is G_{ijk}. Generally, for each vertex v $\in$ V_{ijk} in the partition mesh, there are 8 grid neighbors indexed as G_{lmn}. Here we show the case when $l = i + 1$, $m = j + 1$, $n = k + 1$. Basically we consider two situations: the voxel V_{ijk} is background (see Fig. 3 (c) and (d)) or not background (see Fig. 3 (a) and (b)). In this figure, the blue, green and orange points are grid points with non-zero material IDs. The red vertex is a non-manifold vertex of the partition mesh in each of the four Cases.

In Figure 4, we depict the detection and cure of Case-1 non-manifold vertex caused by two incident voxels possessing the same material ID but meeting only at a single point. The numbers 1, … , 6 represent the remaining six neighboring voxels which are all background with ID 0 and 1, 2, 3 voxels are on the bottom layer and 4, 5, 6 voxels are on the top layer (shown in Figure 4 (b)). Figure 4 (c) shows the result of the cure step, where voxel 1 is given the same material ID. Figure 4 (d) shows the remaining 5 neighborhood voxels. In slightly more details, we first find a voxel V_{ijk} that is not a background voxel its upper right voxel has the same material ID. Then we check the six neighbors of this voxel V_{ijk}. If none of them has the same material ID with voxel V_{ijk} and none of them is background, then we replace the material ID of that voxel with the material ID of voxel V_{ijk}. If all of them have material IDs other than background, then we randomly change one voxel with the material ID of voxel V_{ijk}. Similar approaches apply to Case-2 to Case-4 non-manifold vertices detection and cure.

In Figure 5, we show two examples of non-manifold vertex detection and cure for the Brodmann area data (region 16, see Subsection 5.1), where (a) and (c) show two non-

manifold vertices in the red circle. Figures (b) and (d) show the cure results of (a) and (c) respectively for this Brodmann region.

Compared to the non-manifold vertex case, the detection and cure of non-manifold edges are relatively simpler. In Figure 6, we show the scheme of detecting and curing a potential non-manifold edge. In Fig. 6 (a), the red edge is detected to generate potential non-manifold edges and we repair it as shown in figure (b).

Theorem—Our non-manifold vertex/edge cure (removal) algorithm converges, and correctly yields a manifold partition mesh.

Proof: Non-manifold vertices of the partition mesh are caused by three different material voxels (background having material ID 0), with a single partition vertex in common (the non-manifold vertex). Let the non-manifold vertex be *red* and the neighboring voxel material IDs be *blue*, *green* or *yellow*. The four cases for partition mesh non-manifold vertices we list above are the only independent possibilities (after clustering symmetric cases) because a partition mesh has eight voxel neighbors incident at a partition vertex, with four in each of the two half-spaces, relative to a partitioning medial-plane. Now a non-manifold vertex in a MMO partition mesh necessarily needs a different incident voxel color, say *green* and *blue* assigned to each half-space, and furthermore only one voxel can be labelled *green* and 1 voxel labelled *blue* in each half-space. Otherwise, two or more labelled voxels of the same color *green* or *blue* in each half-space would additionally share a common edge or face, contradicting the requirement for a vertex to be non-manifold. The remaining six voxels would be assigned material ID *yellow*. Since there are four different labels of a half-space voxels with a single color, and the assignment of one half-space restricts the assignment of the other color to the voxel which is diagonally opposite in the other half-space, there can be only 4 different cases of non-manifold vertices. Similar arguments explain the completeness of the non-manifold edge detection. To complete the proof it suffices to see that our non-manifold cure algorithm is based on a one-time re-labelling of the third colored material voxel incident to the non-manifold vertex or edge to one of the three colors. This re-labelling is locally rectifies the non-manifoldness, and given the number of partition mesh voxels are finite, our algorithm is correct and shall converge.

4.3. Partitioned mesh generation

After fixing all the tiny-regions and non-manifold problems, we can generate the boundary representation of each material by partitioning the classification map. For a grid point $G_{i\,jk}$ with material ID $f_{i\,jk}$, we produce a dual cube with six faces with face ID $f_{i\,jk}$. If two dual cubes have common faces with the same face ID, say $f_{i\,jk}$, then we combine these two cubes into one bigger cuboid, and so on. If two dual cubes have common faces with different face IDs, we then produce different surfaces. We call these surfaces indented partitioned meshes or just partitioned meshes. Now each partitioned mesh is topologically a 2-manifold, albeit with piecewise planar facets approximating the MMO boundaries. One should be reminded that each partitioned mesh may include several components. See Figures 5 (b) and (d), Figure 2 (b) for some examples of partitioned meshes.

4.4. Constraint geometric flow smoothing of partitioned meshes

Our next step is to improve the mesh quality of the partitioned meshes while keeping the same topology of the MMO mesh. We adapted a strategy called constrained geometric flow smoothing (CGFS) by invoking a dimensionality based traversal of our partitioned mesh graph. The basic idea is to reposition the vertices of individual meshes according to their connectivity property. We first classify the vertices into three classes:

- single-material vertex: the vertex only belongs to one material.
- double-material vertex: the vertex is only shared by two materials.
- triple-over-material vertex: the vertex is shared by three or more materials.

Our smoothing strategy is accordingly

- For single-material vertex, we adapt surface diffusion flow with tangential regularization to smooth it.
- For double-material vertex, we need to further classify them into two cases. Usually two adjacent materials share one common surface patch, which is a two-dimensional manifold with boundary. Then we classify double-material vertices into two categories:
  - Boundary vertices: the vertices which are on the boundary of the two-dimensional manifold patch. These vertices construct a curve without endpoints. We smooth this piecewise linear curve by taking a convex weighted (partition of unity) average of three vertices. The weight for the central vertex we adopted is 0.6 and the weight for the other two vertices are 0.2. This class of vertices is processed prior to the interior vertices.
  - Interior vertices: the vertices which are not on the boundary of the two-dimensional manifold patch. For this class of vertices, we apply the same smoothing method as for a single-material vertex.
- Triple-over-material vertices are fixed without repositioning.

In the CGFS strategy, the geometric flow adopted is the surface diffusion flow with tangential regularization [6,39], that is

$$\frac{\partial \mathbf{x}}{\partial t} = \Delta H \mathbf{n} + v\mathbf{t}, \quad (1)$$

where **n**, **t** are the normal and tangent vectors to the vertex **x**, respectively. H is the mean curvature, Δ is the Laplace-Beltrami operator, ν is the velocity in the tangent direction and t is a marching parameter. We solve this geometric partial differential equation (1) over a triangular mesh using a discrete scheme [40]. In the temporal dimension, we use a forward Euler scheme. In the spatial dimension, each term is discretized, including the mean curvature H and the Laplace-Beltrami operator [40,41]. This geometric flow is volume-

preserving, that means, the volume enclosed by the surface will be invariant during evolution. The surface diffusion flow with tangential regularization works very well based on our experience.

5. Experimental Results

The pipeline proposed in this paper has been implemented and tested on several examples.

5.1. Brodmann areas

A Brodmann area is a region of the cerebral cortex defined on its cytoarchitectonics, or structure and organization of cells ([42], English version translated by Garey [43]). It's about 50 areas for human and non-human primates. Our data includes 41 regions. In [6], we have previously built multi-material meshes with gaps between each material region. In Figure 1 and Figure 7, we show two different views of the boundary representation of the Brodmann areas.

We list a number of tiny components and non-manifold vertices and edges number in Table 1. In the first iteration, we need 13 loops to make the size of each component over a threshold and then we needed 5 loops to fix the non-manifold vertices and edges. Note the random selection in our cure algorithm. In the second iteration, we need 10 loops to make the size of each component to be under the same threshold as with 70 components (earlier). No non-manifold vertices and edges were detected, i.e., they had all been repaired.

We also computed the volume for each region and the total volume to show volumetric preservation properties of our constrained geometric flow optimization scheme. The statistics is reported in Table 2. The normal volume of a human brain is $1300 - 1500 cm^3$.

5.2. Penicillium stoloniferum virus slow cryo-EM data

We selected an example of the Penicillium stoloniferum virus slow (PsV-S) at 7.3 Angstrom resolution cryo-EM data (EMDB ID:1459) [44] as the second example. The virus capsid is a closed packed shell of individual proteins in a spherical arrangement. The virus capsid houses the viral genome (RNA/DNA). In Figure 8, we show the original density map in (a) by shaded rendering with in-house software VolRover (http://cvcweb.ices.utexas.edu/software). Figures (b) and (c) show the multi-material meshing results of the 60 monomers without boundary points and with boundary points between two adjacent materials, respectively. Figure (d) shows the section view of the capsid. Figure (e) shows the mesh quality and (f) shows one monomer protein mesh with boundary points.

5.3. P22 data

This example shows the multiple material gap-free meshing of bacteriophage P22 at 9.5 Angstrom resolution [45] in Figure 9. A bacteriophage is a virus that infects bacteria. The virus capsid is composed of 60 hexamer and 12 pentamer arrangements of proteins. Figure (a) shows the original volumetric density map by shaded rendering. Figures (b) and (c) show the multi-material mesh result without and with highlighted boundary vertices. Figures (d)

and (e) show the section views of the mesh without and with boundary vertices respectively. Figure (f) shows three hexamers with all boundary vertices and edges.

6. Conclusions

We present a simple but effective algorithm for generating topologically and geometrically consistent quality triangular surface meshing of compactly packed multi-material domains. Each individual material region is represented by a triangulated 2-manifold boundary mesh, with adjacent material regions having shared boundaries. Our algorithm has been implemented, and applied to different multi-material solids, and the results are additionally presented with quantitative analysis of detection and cure of non-manifold interfaces as well as spurious small components. This MMO meshing is in the process of being parallelized on multi-core and many-core computers, and furthermore, coupled to a boundary element Poisson-Boltzmann electrostatic solver. This meshing code has been implemented in VolRover and made freely available from our CVC software page http://cvcweb.ices.utexas.edu/software.

References

[1]. Miyamoto, Y.; Kaysser, WA.; Rabin, BH.; Kawasaki, A.; Ford, RG., editors. Materials Technology Series. 1 ed. Springer; 1999. Functionally graded materials: design, processing and applications.

[2]. Reynolds, NJ., editor. Materials Science and Technologies. Nova Science Pub Inc.; 2011. Functionally graded materials.

[3]. Kou XY, Tan ST. Heterogeneous object modeling: A review. Computer-Aided Design. 2007; 39:284–301.

[4]. Pons J, Segonne F, Boissonnat J, Rineau L, Yvinec M, Keriven R. High quality consistent meshing of multi-label images. Information Processing in Medical Imaging. 2007:198–210. [PubMed: 17633700]

[5]. Boltcheva D, Yvinec M, Boissonnat J. Mesh generation from 3d multi-material images. Medical Image Computing and Computer-Assisted Intervention, LNCS. 2009; 5762:283–290.

[6]. Zhang, Q.; Subramanian, B.; Xu, G.; Bajaj, C. Quality multi-domain meshing for volumetric data. Proc. Int. Conf. Biomed. Eng. Inform.; 2010; p. 472-476.

[7]. Bronson J, Levine J, Whitaker R. Lattice cleaving: A multimaterial tetrahedral meshing algorithm with guarantees. IEEE Transactions on Visualization and Computer Graphics. 2014; 20:223–237. [PubMed: 24356365]

[8]. Hege H, Seebab M, Stalling D, Zockler M. A generalized marching cubes algorithm based on non-binary classification. Technical Report. 1997

[9]. Chen M, Tucker J. Constructive volume geometry. Computer Graphics Forum. 2000; 19:281–293.

[10]. Bischoff S, Kobbelt L. Extracting consistent and manifold interfaces. Bildverarbeitung fur die Medizin. 2006

[11]. Whitaker, R.; Kirby, R.; Sinstra, J.; Meyer, M. Mutlimaterial meshing of mri head data for bioelectric field simulations. Proceedings of the 17th International Meshing Roundtable; 2008; p. 1-5.

[12]. Zhang Y, Hughes T, Bajaj C. An automatic 3D mesh generation method for domains with multiple material. Computer Methods in Applied Mechanics and Engineering. 2010; 199:405–415. [PubMed: 20161555]

[13]. Boltcheva, D.; Yvinec, M.; Boissonnat, J. Feature preserving Delaunay mesh generation from 3D multi-material images. Computer Graphics Forum (special issue for EUROGRAPHICS Symposium on Geometry Processing); 2009. p. 1455-14645.

[14]. Bechly ME, Clausen PD. Structural design of a composite wind turbine blade using finite element analysis. Comput. Struct. 1997; 63:639–646.

[15]. Jackson, TR. Ph.D. thesis. Massachusetts Institute of Technology; 2000. Analysis of functionally graded material object representation methods.

[16]. Jackson TR, Liu H, Patrikalakis NM, Sachs EM, Cima MJ. Modeling and designing functionally graded material components for fabrication with local composition control. Materials and Design. 1999; 20:63–75.

[17]. Kahnt, M.; Ramm, H.; Lamecker, H.; Zachow, S. Feature preserving, multi-material mesh generation using hierarchical oracles. In: Levine, J.; Paulsen, R.; Zhang, Y., editors. MeshMed 2012. Springer-Verlag; 2012. p. 101-111.

[18]. Siu YK, Tan ST. 'source-based' heterogeneous solid modeling. Computer-Aided Design. 2002; 34:41–55.

[19]. Siu YK, Tan ST. Modeling the material grading and structures of heterogeneous objects for layered manufacturing. Computer-Aided Design. 2002; 34:705–716.

[20]. Kumar, V.; Dutta, D. An approach to modeling multi-material objects. In: Hoffmann, C.; Bronsvoort, W.; Allen, G.; Pratt, M.; Rosen, D., editors. Proceedings of the Fourth ACM SIGGRAPH Symposium on Solid Modeling and Applications; New York: ACM; 1997. p. 336-345.

[21]. Chiu WK, Tan ST. Multiple material objects: from CAD representation to data format for rapid prototyping. Computer-Aided Design. 2000; 32:707–717.

[22]. Cheng, S.; Dey, T.; Ramos, EA. Delaunay refinement for piecewise smooth complexes. Symposium on Discrete Algorithms; 2007; p. 1096-1105.

[23]. Lorensen, WE.; Cline, HE. Marching cubes: A high resolution 3D surface construction algorithm. SIGGRAPH '87: Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques; 1987; p. 163-169.

[24]. Wu Z, Sullivan JM. Multiple material marching cubes algorithm. International Journal for Numerical Methods in Engineering. 2003; 58:189–207.

[25]. Wang CCL. Direct extraction of surface meshes from implicitly represented heterogeneous volumes. Computer-Aided Design. 2007; 39:35–50.

[26]. Wang M, Feng J-Q. 2D-manifold boundary surfaces extraction from heterogeneous object on GPU. Journal of Computer Science and Technology. 2012; 27:862–871.

[27]. Shammaa, MH.; Suzuki, H.; Ohtake, Y. Extraction of isosurfaces from multi-material CT volumetric data of mechanical parts. Proceedings of the 2008 ACM Symposium on Solid and Physical Modeling; 2008; p. 213-220.

[28]. Federer, H. Geometric Measure Theory. Vol. ume 153 of Die Grundlehren Der Mathematischen Wissenschaften. Springer; 1969.

[29]. Sharma O, Zhang Q, Anton F, Bajaj C. Fast, streaming 3D level set on the GPU for smooth multi-phase segmentation. Transactions on Computational Sciences. 2011; 13:72–91.

[30]. Chan, TF.; Vese, LA. A level set algorithm for minimizing the Mumford-Shah functional in image processing. Proceedings of the 1st IEEE Workshop on Variational and Level Set Methods in Computer Vision; 2001; p. 161-168.

[31]. Yu Z, Bajaj CL. Automatic ultrastructure segmentation of reconstructed Cryo-EM maps of icosahedral viruses. IEEE Transactions on Image Processing: Special Issue on Molecular and Cellular Bioimaging. 2005; 14:1324–1337.

[32]. Baker ML, Yu Z, Chiu W, Bajaj CL. Automated segmentation of molecular subunits in electron cryomicroscopy density maps. J. Struct. Biol. 2006; 156:432–441. [PubMed: 16908194]

[33]. Zhang Q, Bettadapura R, Bajaj C. Macromolecular structure modeling from 3DEM using VolRover 2.0. Biopolymer. 2012; 97:709–731.

[34]. Pintilie GD, Zhang J, Goddard TD, Chiu W, Gossard DC. Quantitative analysis of cryo-EM density map segmentation by watershed and scale-space filtering, and fitting of structures by alignment to regions. J. Struct. Biol. 2010; 170:427–438. [PubMed: 20338243]

[35]. Mortenson, ME. Geometric Modeling. 2 ed. John Wiley & Sons; 1997.

[36]. Stroud I. Boundary modelling with special representations. Computer-Aided Design. 1994; 26:543–550.

[37]. Wanger, M.; Labsik, U.; Greiner, G. Repairing non-manifold triangle meshes using simulated annealing. Proceedings of the 4th Israel-Korea Bi-National Conference on Geometric Modeling and Computer Graphics; 2003; p. 88-93.

[38]. Guéziec A, Taubin G, Lazarus F, Horn B. Cutting and stitching: Converting sets of polygons to manifold surfaces. IEEE Transactions on Visualization and Computer Graphics. 2001; 7:136–151.

[39]. Zhang Y, Xu G, Bajaj C. Quality meshing of implicit solvation models of biomolecular structures. Computer Aided Geometric Design. 2006; 23:510–530. [PubMed: 19809581]

[40]. Xu G, Pan Q, Bajaj C. Discrete surface modelling using partial differential equations. Computer Aided Geometric Design. 2006; 23:125–145. [PubMed: 19830268]

[41]. Xu G, Bajaj CL. Regularization of B-spline objects. Computer Aided Geometric Design. 2011; 28:38–49. [PubMed: 21218183]

[42]. Brodmann, K. Vergleichende Lokalisationslehre der Grosshirnrinde. Johann Ambrosius Bart; Leipzid: 1909.

[43]. Garey, LJ. Brodmann's Localisation in the Cerebral Cortex. Springer; New York: 2006.

[44]. Ochoa WF, Havens WM, Sinkovits RS, Nibert ML, Ghabrial SA, Baker TS. Partitivirus structure reveals a 120-subunit, helix-rich capsid with distinctive surface arches formed by quasisymmetric coat-protein dimers. Structure. 2008; 16:776–786. [PubMed: 18462682]

[45]. Jiang W, Li Z, Zhang Z, Baker ML, P. E. P. Chiu W. Coat protein fold and maturation transition of bacteriophage P22 seen at subnanometer resolutions. Nature Structural Biology. 2003; 10:131–135. [PubMed: 12536205]

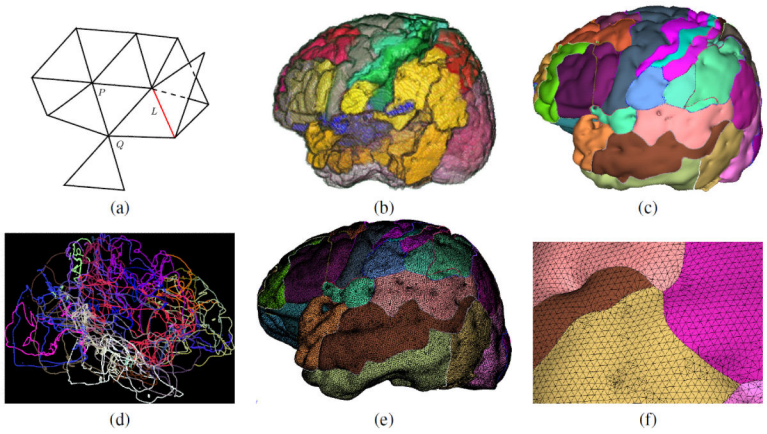

Fig. 1.

Multi-material mesh of the Brodmann classified regions of the human cortex. (a) Non-manifold triangulated surface mesh cases. Here P is a manifold vertex, Q is a non-manifold vertex and L is a non-manifold edge. (b) Region based colored rendering of the input voxellized classification map. (c) Multiple material mesh representation with highlighted boundary vertices (0D), curvilinear boundary edges (1D), and patch-surface faces (2D). (d) Wireframe boundary graph with vertices and smoothed curvilinear edges. (e) Shaded rendering with wireframe of the final multiple material mesh. (f) A zoomed in view of the final multiple material mesh as shown in (e).

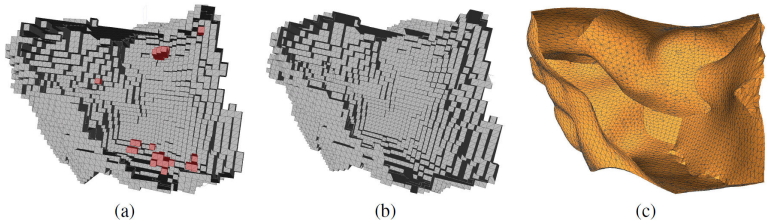

Fig. 2.

Tiny component detection and removal (cure). (a) The red small regions are detected. (b) Section view after tiny regions removed. (c) The resulting mesh after the final smoothing and quality improvement step.

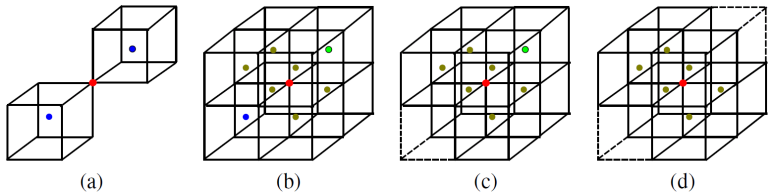

Fig. 3.

Four cases of multi-material voxel classifications causing a red non-manifold vertex of the partition mesh. The blue, green and orange points are grid points with non-zero material IDs. The other grid points are with zero material ID. (a) Case-1: the red vertex is non-manifold. (b) Case-2: the red vertex is non-manifold as the situation is similar to case 1. (c) Case-3: the red vertex is non-manifold when the lower left voxel is empty. (d) Case-4: the red vertex is non-manifold when the upper right voxel is additionally empty.

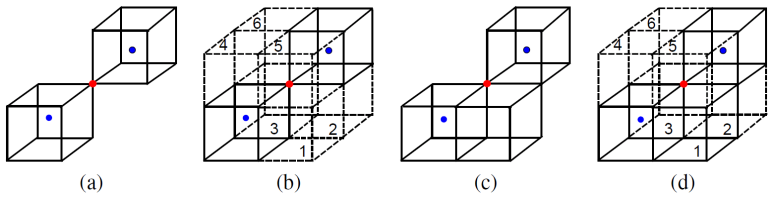

Fig. 4.

Partition mesh Case-1 non-manifold vertices detection and cure. (a) The red vertex is a non-manifold vertex. (b) The neighbors of the non-manifold vertex. (c) After cure, the red vertex is a manifold vertex as the added voxel 1 in the lower right has the same material ID as blue. (d) The neighbors of the cured manifold vertex.

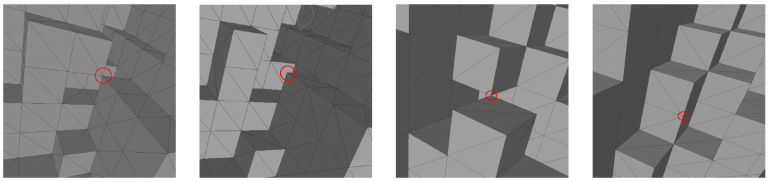

Fig. 5.

Partition mesh non-manifold vertex detection and cure applied to the brain Brodmann area region 16 data. (a) and (c) The vertices in the red circles are non-manifold vertices that were detected. (b) and (d) The vertices in the red circles are manifold vertices after cure.

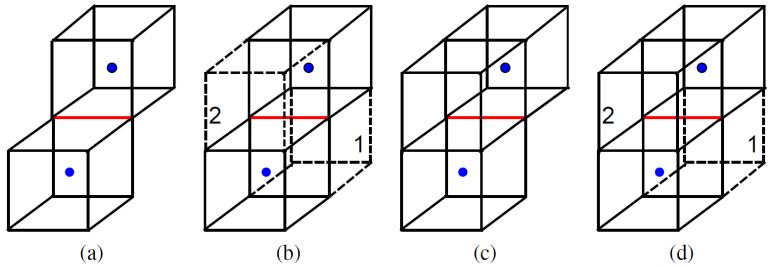

Fig. 6.

Partition mesh non-manifold edge detection and cure. (a) The red edge is a non-manifold edge. (b) All the neighboring voxels. (c) After addition of the upper left voxel 2, the red edge becomes a manifold edge. (d) Neighborhood status after cure.

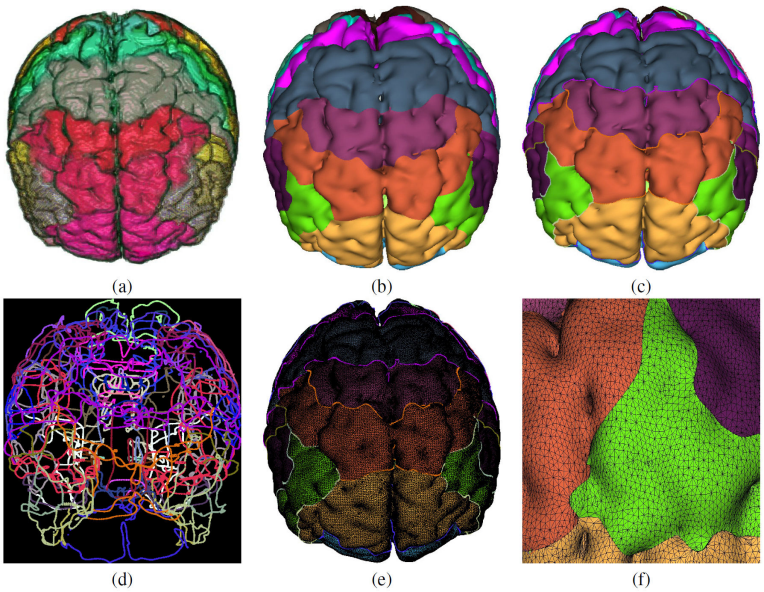

Fig. 7.

Brodmann areas of brain. (a) Original partition map. (b) Solid rendering of the multiple material mesh. (c) Multiple material mesh representation with enhanced rendering of the boundary vertices. (d) Boundary vertices linked by smoothed boundary curves (edges). (e) Solid rendering with wireframe of the multiple material mesh. (f) A zoomed in view of a portion of meshes in (e).

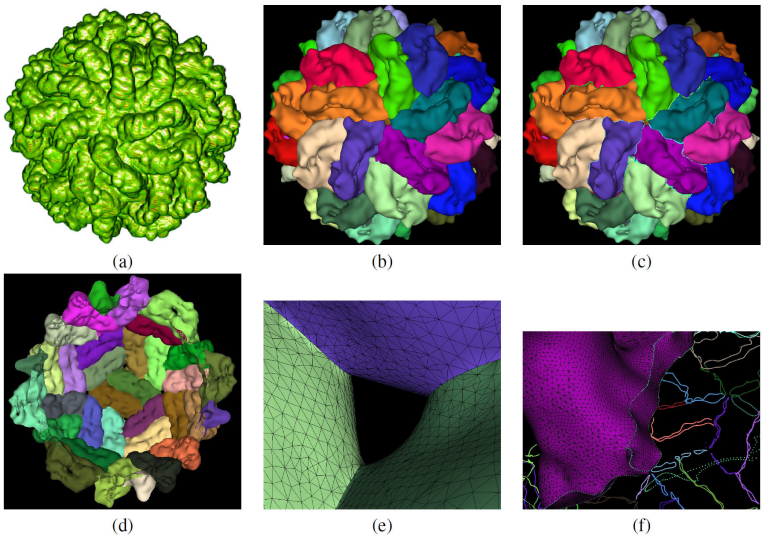

Fig. 8.

PsV-S cryo-EM data at 7.3 Angstrom resolution. (a) Original volumetric density map of the spherical capsid shell shown using shaded volume rendering. (b) Multiple material gap-free meshing of each monomer (60 monomers). (c) Multiple material mesh representation with enhanced rendering of boundary vertices. (d) Section view. (e) A zoomed in view for mesh quality visualization. (f) One monomer protein mesh with boundary edges and points.

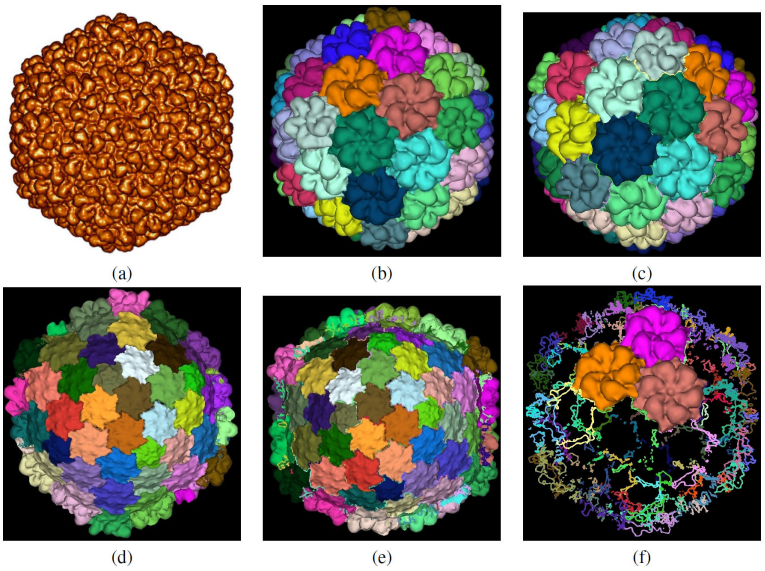

Fig. 9.

Bacteriophage P22 cryo-EM data at 9.5 Angstrom resolution. (a) Original volumetric density map of the spherical capsid shell shown using shaded volume rendering. (b) Multiple material gap-free meshing of 60 hexamers and 12 pentamers. (c) Multiple material mesh representation with boundary vertices. (d) Section view. (e) A section view with enhanced rendering of boundaries. (f) Three hexamers with enhanced rendering of boundaries.

Table 1

Brodmann areas with tiny component fixing and non-manifold fixing statistics.

Iteration	Loop	# components	Loop	# non-manifold vertices and edges
1	1	993	1	2867
	2	492		
	3	347		
	4	255	2	598
	5	182		
	6	129		
	7	102	3	48
	8	93		
	9	83		
	10	78	4	6
	11	82		
	12	75	5	0
	13	72		
2	1	164	1	0
	2	130		
	3	114		
	4	98		
	5	88		
	6	81		
	7	74		
	8	72		
	9	72		
	10	70		

Table 2

The volumes of individual Brodmann areas. Unit mm^3.

area	volume	area	volume	area	volume	area	volume
1	24993.190980	12	83306.690610	22	18922.652301	32	4626.289581
2	14238.734365	13	18462.201872	23	37193.895606	33	50042.611639
3	19529.177004	14	1186.165355	24	34544.761322	34	46977.748187
4	77075.120754	15	66535.767843	25	15218.418055	35	30659.084403
5	161986.839854	16	6788.072364	26	27051.367103	36	4759.113220
6	98607.168977	17	25681.920613	27	33032.876507	37	22497.331052
7	13010.120724	18	1668.009589	28	28609.124907	38	32466.247486
8	32168.631313	19	10311.779490	29	2402.269414	39	27998.900245
9	81971.079039	20	7579.996076	30	5720.112915	40	36009.643504
10	15881.479013	21	28036.965678	31	5239.390048	41	6506.035475
11	90137.361823						
total	1349634.346						