Simplification of Vector Fields over Tetrahedral Meshes

Nikos Platis and Theoharis Theoharis
Department of Informatics & Telecommunications, University of Athens,
Panepistemiopolis, 157 84 Ilissia, Greece
{nplatis|theotheo}@di.uoa.gr

Abstract

Vector fields produced by experiments or simulations are usually extremely dense, which makes their manipulation and visualization cumbersome. Often, such fields can be simplified without much loss of information.

A simplification method for 3D vector fields defined over tetrahedral meshes is presented. The underlying tetrahedral mesh is progressively simplified by successive half-edge collapses. The order of collapses is determined by a compound metric which takes into account the field and domain error incurred as well as the quality of the resulting mesh. Special attention is given to the preservation of the mesh boundary and of critical points on the vector field.

A tool has been developed for the measurement of the difference between two vector fields over tetrahedral meshes, and it is used to quantify the simplification error.

1 Introduction

Data visualization techniques deal commonly with three-dimensional scalar and vector fields defined over tetrahedral meshes. Field values are known only at the vertices of the mesh; at any other point the corresponding field value is obtained by linear interpolation of the values at the vertices of the tetrahedron containing the point. Owing to advances in computing and data acquisition techniques, these fields are very dense and detailed, at the cost of requiring considerable resources for their processing.

To alleviate this, several methods that produce simplified approximations of such fields have been developed. The complexity of the field is reduced in a controlled manner, with the aim to retain its important characteristics and to minimize the approximation error of the resulting field.

In this paper we present a simplification method for vector fields over tetrahedral meshes which applies iterative edge collapses to the mesh. To quantify the simplification error of each candidate edge collapse we propose a compound error metric that takes into account the change to the vector field and its domain as well as the quality of the produced mesh. In order to preserve important features of the vector field, we pay special attention to the boundary of the tetrahedral mesh and to critical points of the field.

To assess the actual error between the original vector field and the simplified one, we have developed a software tool that measures the difference between two vector fields in terms of various error metrics.

This work is the first application of a simplification method based on iterative edge collapses to vector fields over tetrahedral meshes. Such methods have been applied successfully to triangular surface meshes, with or without associated attributes, and more recently, to tetrahedral meshes with associated scalar fields. Our algorithm considers the special characteristics and requirements of vector fields in order to guide the simplification process; it does not merely produce a simplified visualization but it meaningfully reduces the size of the vector field.

2 Related Work

2.1 Tetrahedral Mesh Simplification

The literature on tetrahedral mesh simplification is not very extended. The benefits of edge collapse based simplification methods for triangular surface meshes [12, 9, 14] has led to their adaptation for tetrahedral meshes and several authors use them, directly or indirectly.

Staadt and Gross [15] presented the first complete simplification algorithm based on edge collapses for tetrahedral meshes. They order edge collapses using a compound function that weighs the change in the scalar field, the change in overall mesh volume and the change in tetrahedra shape. They also present a robust technique to avoid self-intersections of the boundary of the mesh during simplification. Trotts *et al.* [19] determine upper bounds for the field and domain error caused by each edge collapse and consider them separately. The most thorough treatment of tetrahedral mesh simplification is provided by Cignoni *et al.* [3]. They define formally the simplification error and

evaluate several methods for estimating it during the simplification process, trading accuracy for efficiency. Finally, Chopra and Meyer [2] simplify tetrahedral meshes by collapsing tetrahedra; their method exhibits interesting properties regarding the mesh, but their treatment of the associated field is elementary.

2.2 Vector Field Simplification

Simplification of vector fields has mostly treated, so far, scattered data vector fields: isolated points with associated vector values but without any connectivity information. Clustering approaches are very common in this case. Authors are most often guided by the need to present a simplified representation of the information contained in an involved vector field.

Heckel *et al.* [10] measure the simplification error, and guide the clustering process, by the difference in the streamlines of the initial and the simplified field. Telea and van Wijk [16] compute a representative vector for each cluster and use them to merge similar clusters. They also present an advanced visualization technique with curved vectors for the simplified fields. Tricoche *et al.* [18] are mostly concerned with fields containing excessive numbers of critical points; they create clusters based on the number of critical points contained in each of them and merge those contained in each cluster in order to simplify the field. Garcke *et al.* [8] are inspired by a physical clustering model, which they adapt to guide a simplification process for vector fields; they enhance it in order to control the shape of generated clusters and apply it mainly to flow fields.

3 Simplification of Vector Fields over Tetrahedral Meshes

We will denote a vector field as $\Phi = (\mathcal{V}, \mathcal{T}, \mathcal{F})$, where $\mathcal{V}$ is a set of vertices, $\mathcal{T} = \{T_1, T_2, \ldots, T_m\}$ is a tetrahedralization of $\mathcal{V}$, and $\mathcal{F} = \{\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_m\}$ is a set of functions such that each $\mathbf{F}_i$ is defined over tetrahedron T_i as the linear interpolant of the field values at the vertices of T_i. We will denote by Ω the *domain* of the vector field, the region of space spanned by $\mathcal{T}$.

3.1 Simplification Algorithm

Our algorithm follows the general scheme of most other simplification algorithms based on iterative edge collapses:

- Each candidate collapse is assigned a *priority*, indicative of its impact on the mesh, and inserted in a priority queue.
- While legal collapses exist in the queue and the simplification target has not been reached,
  - The collapse with highest priority is removed from the queue and applied to the mesh. Tetrahedra that contain the collapsed edge are deleted; those that contain only the deleted vertex a have this vertex replaced by b.
  - All affected edges have their priorities (and positions in the queue) updated.

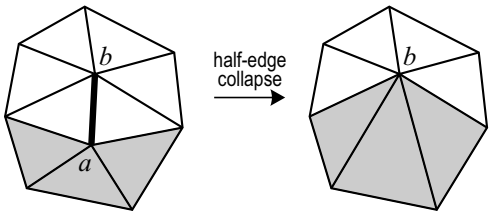

Figure 1. Half-edge collapse $(a, b) \rightarrow b$ for a triangular surface. The triangles affected by the collapse are greyed out.

The edge collapse operation is fully reversible and the original field can be recovered exactly.

In this work we allow edges to contract only to either of their endpoints and we retain the original field value at the target point. In this way, an edge $\{a, b\}$ of the mesh produces two candidate *half-edge collapses*, $(a, b) \rightarrow b$ and $(b, a) \rightarrow a$. The effect of a half-edge collapse is restricted to the tetrahedra around its origin (Figure 1 shows a half-edge collapse on a triangular mesh for simplicity). This simplification scheme was favored over general edge collapses, where the resulting point is placed freely and the associated field value must be computed so as to minimize the simplification error, for its simplicity and efficiency; also, since the original field values are used throughout, the possibility to distort the field topology is reduced.

Edge collapse validity Edge collapses may result in inconsistencies, such as non-manifoldness [7], or artifacts on the tetrahedral mesh. To avoid them, suitable tests are performed before each candidate collapse is inserted in the queue:

- The topology of the mesh is maintained using the link condition of [6, 3].
- The volume of each tetrahedron affected by the collapse must not change sign or become zero, in order to prevent creation of inverted or degenerate tetrahedra.

Simplification error The simplification error incurred as a result of an edge collapse can be analyzed into two components [3]:

- The *domain error* characterizes the change in the domain Ω of the vector field, which may be altered if the collapsed edge has one or both endpoints on the mesh boundary. The domain error can be measured by the Hausdorff distance between the boundaries of the original and the simplified mesh [4].

- The *field error* quantifies the change in the vector field. If $\Phi = (\mathcal{V}, \mathcal{T}, \mathcal{F})$ and $\Phi' = (\mathcal{V}', \mathcal{T}', \mathcal{F}')$ are the original and the simplified field respectively, then the field error can be measured by the differences $\mathbf{F}(x) - \mathbf{F}'(x)$ at points x in the domain, for example

$$\epsilon_F = \frac{1}{|\Omega|} \int_{\Omega} \|\mathbf{F}(x) - \mathbf{F}'(x)\|^2 dv.$$

  As noted above, the domain of the field may change after an edge collapse, and in this case the definitions of the interpolants $\mathbf{F}$ and $\mathbf{F}'$ should be extended appropriately to cover Ω' and Ω respectively.

Owing to the locality of the edge collapse operation, the two fields only differ over the tetrahedra affected by the collapse, and their boundaries only on the boundary faces of these tetrahedra. However, computing the above measures on even these restricted regions at each simplification step is impractical; in the following section we present the error estimation that we used.

3.2 Error estimation

The error assigned to each candidate edge collapse is a weighted sum of several components:

$$E = w_{F_A} E_{F_A} + w_{F_L} E_{F_L} + w_D E_D + w_C E_C + w_V E_V$$

These error components are analyzed below. In brief, the first two components are estimates of the field error, the third is an estimate of the domain error, and the last two components control the quality of the simplified mesh in terms of tetrahedra shape.

We should note that, since the error components are expressed in different scales, each of them is normalized in the interval $[0, 1]$ when the original edges are inserted in the queue; the resulting normalization coefficients are used thereafter.

In the following, we will refer to a half-edge collapse $(a, b) \rightarrow b$.

Field Error E_{F_A} and E_{F_L} quantify the field error of the simplification. E_{F_A} measures the angle deviation between the original and the simplified field, and E_{F_L} their (vector) difference. The error estimate is based on the value $\mathbf{F}'(a)$ of the simplified field at the deleted vertex a (Figure 2); thus we set:

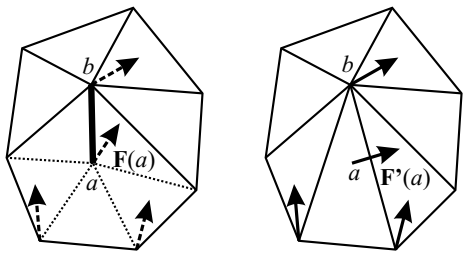

Figure 2. The field error is estimated in terms of the field values at the deleted vertex a before and after the collapse.

$$\begin{aligned} E_{F_A} &= \angle(\mathbf{F}(a), \mathbf{F}'(a)) \\ E_{F_L} &= \|\mathbf{F}(a) - \mathbf{F}'(a)\| \end{aligned}$$

To calculate $\mathbf{F}'(a)$ we form the tetrahedra that would be created by the collapse by substituting a with b on the affected tetrahedra; we find which new tetrahedron contains the deleted vertex a, and compute the new field value at a by interpolating the field values at the vertices of this tetrahedron. If the boundary of the mesh changes, a may not be contained in any of the new tetrahedra. In this case, we find the new tetrahedron whose center is closest to a and extrapolate the field values at its vertices to obtain $\mathbf{F}'(a)$; if more than one tetrahedra are at the same minimum distance to a, their respective extrapolated field values are averaged to give $\mathbf{F}'(a)$.

In order to represent the original field better, the two components of the field error are *accumulated* during the simplification process; thus each candidate edge collapse carries on recursively the errors incurred by all edges collapsed to its a.

Domain Error For the domain error E_D we initially used the maximum distance of the deleted vertex a to the boundary faces of the new tetrahedra, as an estimate of the Hausdorff distance of the boundaries. However, experimentation revealed that this measure of the boundary error failed to preserve the mesh boundary reliably (see also Section 3.3).

To overcome this, we estimate the boundary error in terms of the dihedral angles between corresponding boundary faces before and after the collapse (Figure 3). Specifically, if $\mathcal{BF}'(b)$ is the set of affected boundary faces around b after the collapse and $\theta(f)$ is the aforementioned angle for a face $f \in \mathcal{BF}'(b)$, we set

$$E_D = \max_{f \in \mathcal{BF}'(b)} \theta(f)$$

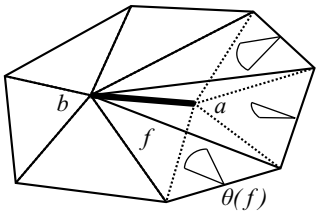

Figure 3. The domain error is estimated in terms of the dihedral angles $\theta(f)$ between corresponding faces before and after the collapse.

The domain error is also accumulated during the course of the simplification algorithm, so that the change to the boundary is estimated with respect to the boundary of the original mesh.

Tetrahedra Compactness Error E_C penalizes the creation of slivery tetrahedra, which are generally undesired since they produce artifacts during rendering and interpolations. As a measure of *compactness* of a tetrahedron T we use the value [13]

$$c(T) = \frac{6\sqrt{2}S(T)}{L_\mu^3(T)}$$

where $S(T)$ is the volume of T and $L_\mu(T)$ is its average edge length. This formula yields 0 for a flat tetrahedron and 1 for a regular one.

We calculate the minimum compactness c_{min} of the tetrahedra around (a, b) before the collapse and the minimum compactness c'_{min} of the tetrahedra around b after the collapse, and take

$$E_C = \frac{c_{min}}{c'_{min}}$$

as a measure of compactness deterioration due to the collapse.

Valence Error E_V penalizes the creation of vertices of high *valence* on the mesh, vertices shared by many tetrahedra. For a half edge collapse, E_V is simply the number of tetrahedra around b after the collapse,

$$E_V = \text{Valence}'(b).$$

3.3 Boundary preservation

Tetrahedral meshes used in data visualization commonly have rectangular or curvilinear boundary, corresponding to the space in which the experiment or simulation takes place; in such cases it is desirable to preserve the shape of the boundary as much as possible.

To this end, we seeked to simplify the tesselation of planar boundary regions while allowing only small deterioration of curvilinear parts. Thus we have adopted the following strategy:

- An edge lying on the mesh boundary is only allowed to collapse if the domain error incurred is below a user-specified threshold; this threshold is 0 for fields with planar boundaries and very small for curvilinear boundaries. In this context, the domain error estimation based on the deviation angle of boundary faces is an intuitive measure of accepted boundary deterioration.

- An edge with one boundary vertex is only allowed to collapse toward its boundary vertex. Thus if b is a boundary vertex but a is not, the half-edge collapse $(a, b) \rightarrow b$ is permitted whereas the inverse collapse is disallowed, since it would compromise (shrink) the boundary. For greater simplification of the boundary, the threshold mentioned in the previous case could be considered here as well.

- An edge may have both its endpoints on the boundary, but it may be on the interior of the mesh (for example, an interior edge of a cube with endpoints on two different sides); in this case its collapse is disallowed completely.

Other authors [15] have described sophisticated and involved techniques to avoid self-intersections during modifications of the field boundary. Our assumption for almost regular boundaries and our provision for small modifications during simplification make such techniques unnecessary.

3.4 Critical points

Critical points, at which the value of a vector field becomes zero, are important for its visualization since they correspond to distinctive features of the field [11]. The topology of vector fields containing many and dense critical points can be analyzed as described in [17], and simplified properly with techniques such as those found in [18, 5]. Our simplification scheme using edge collapses can be augmented with the method presented in [1] to preserve the topology of the field. However, for vector fields containing a moderate number of critical points we have successfully applied the following strategy:

- An edge whose both endpoints are critical is only allowed to collapse if its length is below some user-specified threshold; this threshold is set rather small,

so that only very closely neighboring critical points are merged and the field topology is not affected significantly.

- An edge with one critical endpoint is only allowed to collapse toward its critical endpoint. Thus if b is a critical point but a is not, the half-edge collapse $(a, b) \rightarrow b$ is permitted whereas the inverse collapse is disallowed, since it would destroy a critical point.

It is assumed that all critical points lie on vertices of the original mesh. Otherwise they can be detected, in a preprocessing step, by examining the field, and the tetrahedra that contain critical points should be split on them so that the above strategy can be applied.

4 Error Measuring Tool

In order to measure the actual field error of our simplified meshes, we have developed a software tool that measures the difference between two vector fields defined over tetrahedral meshes.

Our tool samples the two fields Φ and Φ' at a common set of sample points S and collects their respective values, $\{\mathbf{F}(x), x \in S\}$ and $\{\mathbf{F}'(x), x \in S\}$. In a similar manner to Section 3.2 above, field values are computed by finding the tetrahedron that contains the sample point x and interpolating the field values at its vertices; if the sample point is external to the domain of the field, its value is estimated by extrapolation with respect to the nearest tetrahedra. A regular grid covering the field domain is used throughout to help locate the containing or nearest tetrahedra for a sample point.

Having collected the field values at the sample points, various error measures may be computed. In order to represent the simplification error, we implemented the following:

- **Max and mean angle deviation:**
  If $\epsilon_A(x) = \angle(\mathbf{F}(x), \mathbf{F}'(x))$, we take

$$\begin{aligned} \epsilon_A^{max} &= \max_{x \in S} \{\epsilon_A(x)\} \\ \epsilon_A^{mean} &= \frac{1}{|S|} \sum_{x \in S} \epsilon_A(x) \end{aligned}$$

- **Max and mean vector deviation:**
  If $\epsilon_L(x) = \|\mathbf{F}(x) - \mathbf{F}'(x)\|$, we take

$$\begin{aligned} \epsilon_L^{max} &= \max_{x \in S} \{\epsilon_L(x)\} \\ \epsilon_L^{mean} &= \frac{1}{|S|} \sum_{x \in S} \epsilon_L(x) \end{aligned}$$

For our purposes Φ is the original and Φ' is the simplified field, and the points of S are sampled on a dense regular grid covering the domain of the original model. In the tables of Section 5 that follows, angle deviations are expressed in degrees. Also, to be more meaningful, vector deviations are expressed relatively as percentages of the range r of measures in the field Φ,

$$r = \max_{x \in \mathcal{V}} \{\|\mathbf{F}(x)\|\} - \min_{x \in \mathcal{V}} \{\|\mathbf{F}(x)\|\}.$$

5 Results

Our simplification algorithm was implemented as an OpenDX module, but it may be easily adapted to other visualization frameworks. We tested the algorithm on several different datasets.

The first field is a linear field created at the vertices of a uniformly subdivided cube. Inside the cube its values are given by $\mathbf{F}(x, y, z) = (x, 2x, 0)$. The initial field is comprised of 3645 tetrahedra. The simplified field (Figure 4) consists of 10 tetrahedra and it represents the original field exactly. The boundary threshold was set to 0 for this simplification.

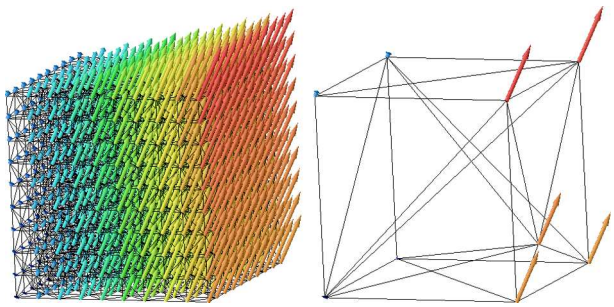

Figure 4. The linear field, original (3645 tetrahedra) and simplified (10 tetrahedra).

The second dataset (**gravity**) represents the gravitational field produced by three spheres. It has 121,945 tetrahedra, rectangular boundary and three critical points at the locations of the spheres. For its simplification we imposed a boundary threshold of 0 and a critical point merge threshold of 0 as well. As expected, the boundary was preserved exactly and the critical points were not merged. Figure 5 shows a cutting plane through the center of the field at various levels of detail. It is clear that the field was simplified more at the areas where it is more uniform, away from the three spheres.

The other two datasets are vector components of the **bluntfin** dataset[1] and of a fluid flow simulation (**pipe**)[2],

[1] The bluntfin model was obtained from NASA

[2] The pipe model was obtained from the OpenDX BonusPak.

comprised of 187,318 and 10,872 tetrahedra respectively. Both these datasets have curvilinear boundary and no critical points. For their simplification we imposed a low boundary threshold of $\pi/4$, which led to very good preservation of the boundary shape. Figures 6 and 7 show cutting planes through these two fields in various resolutions. Again, the fields were simplified more over their more uniform regions while their important characteristics were preserved.

Table 1 provides simplification times and error measurements for the last three test models. Each model was simplified at 50%, 25% and 10% of its original tetrahedra. The weights applied to the error components are the italicized ones in table 2. Timings include the complete simplification process, starting with the original field. Our tests were performed on an AMD Athlon XP 2000+ PC with 512 MB RAM running Linux; the code was compiled with gcc 3.2.

Table 1. Simplification results for various resolutions.

%	time	ϵ_A^{max}	ϵ_A^{mean}	ϵ_L^{max}	ϵ_L^{mean}
			gravity		
50	165 s	19.7958	0.9320	0.9041	0.0221
25	227 s	21.9034	2.2026	1.1491	0.0682
10	269 s	36.2294	3.6761	5.6567	0.1977
			bluntfin		
50	316 s	8.8754	0.1342	8.3856	0.2438
25	431 s	12.8705	0.2554	11.1031	0.4550
10	502 s	20.6755	0.4560	15.7483	0.8389
			pipe		
50	13 s	6.4064	0.3291	6.7925	0.3617
25	18 s	22.6138	0.5481	27.8060	1.0679
10	21 s	67.2400	1.0346	81.3354	2.5356

Choosing suitable weights for the error components is inherently difficult and data-dependent. Our results reveal that E_{F_L} is the most effective error component, and can produce reasonable results even if used alone. The shape control error components E_C and E_V are always helpful, with moderately lower weights. Finally, the effect of E_{F_A} and E_D depends on the model and their use requires some experimentation. Table 2 presents some of our results for the **gravity**, **bluntfin** and **pipe** models.

6 Conclusions and Further Work

We presented a robust method, based on half-edge collapses, for the simplification of vector fields defined over tetrahedral meshes. We proposed efficient techniques for estimating the various errors during simplification, and effective strategies for treating the field boundary and critical points in order to obtain high quality results. Finally, we constructed a simple measuring tool in order to quantify the actual error incurred by our simplification algorithm. Our results manifest that our algorithm can successfully simplify complex vector fields.

Our current work can be extended toward several directions. Initially we would like to generalize the edge collapses and select the resulting vertex position and field value so as to minimize the distortion of the field. Several such algorithms are already available for surface meshes, but they have not been applied to tetrahedral meshes, either with scalar or with vector fields associated.

Moreover, several fields used in scientific applications have multiple quantities sampled at each vertex, both scalar and vector. A simplification algorithm that would try to minimize the simplification error with respect to all their components simultaneously would be beneficial.

Finally, more robust treatment of the field topology would be needed in order to apply our simplification algorithm to more complex vector fields; techniques similar to the ones referred to in Section 3.4 should be incorporated in our algorithm.

References

[1] Y.-J. Chiang and X. Lu. Progressive Simplification of Tetrahedral Meshes Preserving All Isosurface Topologies. *Computer Graphics Forum*, 22(3):493–504, 2003.

[2] P. Chopra and J. Meyer. Tetfusion: An algorithm for rapid tetrahedral mesh simplification. In *Proc. IEEE Visualization 2002*, pages 133–140, 2002.

[3] P. Cignoni, D. Constanza, C. Montani, C. Rocchini, and R. Scopigno. Simplification of Tetrahedral Meshes with Accurate Error Evaluation. In *Proceedings of IEEE Visualization 2000*, pages 85–92.

[4] P. Cignoni, C. Roccini, and R. Scopigno. Metro: Measuring Error on Simplified Surfaces. *Computer Graphics Forum*, 17(2):167–174, 1998.

[5] W. de Leeuw and R. van Liere. Collapsing Flow Topology Using Area Metrics. In *Proceedings of IEEE Visualization '99*, pages 349–354.

[6] T. K. Dey, H. Edelsbrunner, S. Guha, and D. V. Nekhayev. Topology Preserving Edge Contraction. *Publications de l'Institut Mathématique (Beograd)*, 66(80):23–45, 1999.

[7] C. Forest, H. Delingette, and N. Ayache. Removing Tetrahedra from a Manifold Mesh. In *Proc. Computer Animation 2002*, pages 225–229, 2002.

[8] H. Garcke, T. Preußer, M. Rumpf, A. Telea, U. Weikard, and J. J. van Wijk. A Phase Field Model for Continuous Clustering on Vector Fields. *IEEE Transactions on Visualization and Computer Graphics*, 7(3):230–241, 2001.

[9] M. Garland and P. S. Heckbert. Simplifying Surfaces with Color and Texture using Quadric Error Metrics. In *Proceedings of IEEE Visualization '98*, pages 263–270.

[10] B. Heckel, G. Weber, B. Hamann, and K. I. Joy. Construction of Vector Field Hierarchies. In *Proceedings of IEEE Visualization '99*, pages 19–25.

Table 2. Effect of error weights, for simplification at 25% of the original tetrahedra; the best approximations are italicized.

w_{F_A}	w_{F_L}	w_D	w_C	w_V	ϵ_A^{max}	ϵ_A^{mean}	ϵ_L^{max}	ϵ_L^{mean}
					gravity			
10					60.3598	3.1799	45.7937	0.1539
	10				22.2023	2.0851	5.6567	0.0798
	10		*0.5*	*0.5*	*19.7958*	*2.0892*	*1.1433*	*0.0675*
1	10		0.5	0.5	19.7958	2.1312	5.6556	0.0669
					bluntfin			
10					15.6074	0.3839	21.0197	1.2166
	10				11.7518	0.3093	923.2150	0.6125
	10		5	5	14.8024	0.2587	8.4035	0.4924
10	10		5	5	15.3043	0.3346	13.2514	0.7290
10	10	10	5	5	14.6855	0.2962	12.7824	0.6143
	10	*10*	*5*	*5*	*12.8705*	*0.2554*	*11.1031*	*0.4550*
					pipe			
10					50.7635	0.5656	51.7177	1.6452
	10				12.8889	0.6291	40.8704	1.1285
	10		5	5	20.1284	0.5678	32.7031	1.0931
5	*10*		*5*	*5*	*22.6138*	*0.5481*	*27.8060*	*1.0679*
5	10	5	5	5	16.0302	0.5816	33.5341	1.1970
	10	5	5	5	20.1284	0.5692	32.7031	1.0920

[11] J. Helman and L. Hesselink. Visualizing vector field topology in fluid flows. *IEEE Computer Graphics and Applications*, 11(3):36–46, 1991.

[12] H. Hoppe. Progressive Meshes. In *SIGGRAPH 96 Proceedings*, pages 99–108.

[13] D. L. Marcum and N. P. Weatherill. Unstructured Grid Generation Using Iterative Point Insertion and Local Reconnection. *AIAA Journal*, 33(9):1619–1625, 1995.

[14] N. Platis and T. Theoharis. Progressive Hulls for Intersection Applications. *Computer Graphics Forum*, 22(2):107–116, 2003.

[15] O. G. Staadt and M. H. Gross. Progressive Tetrahedralizations. In *Proceedings of IEEE Visualization '98*, pages 397–402.

[16] A. Telea and J. J. van Wijk. Simplified Representation of Vector Fields. In *Proceedings of IEEE Visualization '99*, pages 35–42.

[17] Y. Tong, S. Lombeyda, A. N. Hirani, and M. Desbrun. Discrete Multiscale Vector Field Decomposition. In *SIGGRAPH 2003 Proceedings*, pages 445–452.

[18] X. Tricoche, G. Scheuermann, and H. Hagen. A Topology Simplification Method For 2D Vector Fields. In *Proceedings of IEEE Visualization 2000*, pages 359–366.

[19] I. J. Trotts, B. Hamann, and K. I. Joy. Simplification of Tetrahedral Meshes with Error Bounds. *IEEE Transactions on Visualization and Computer Graphics*, 5(3):224–237, 1999.

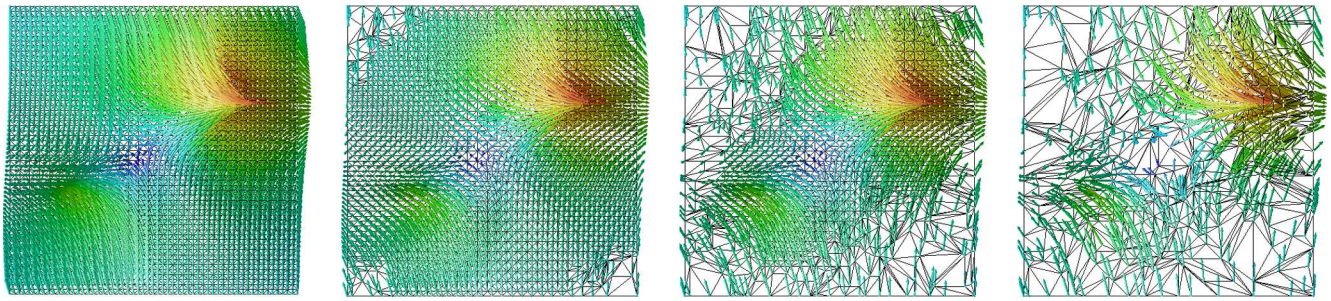

Figure 5. The gravity field, original and simplified at 50%, 25% and 10%.

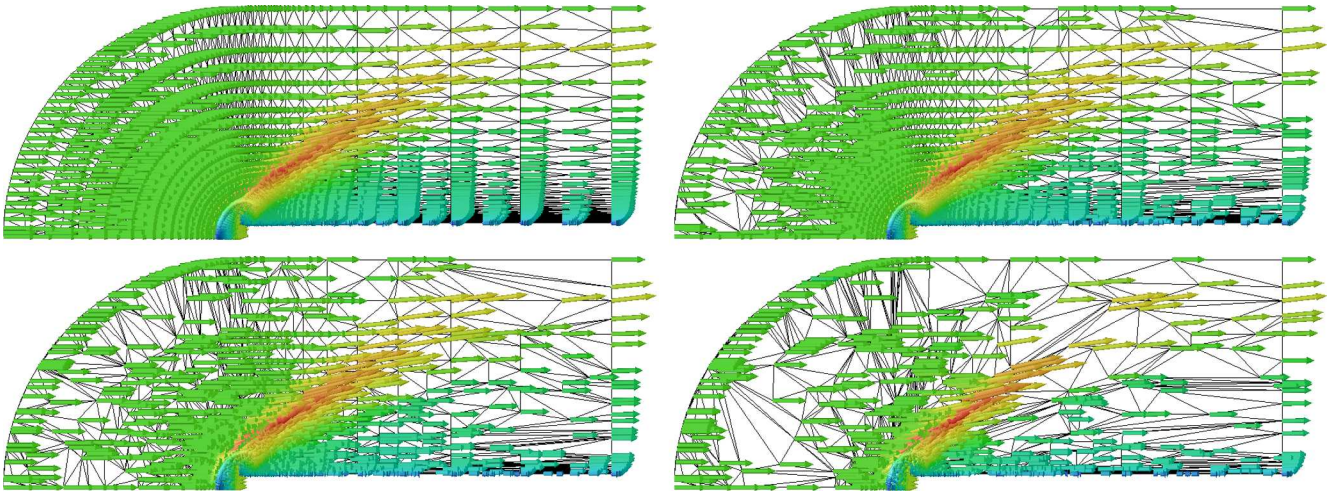

Figure 6. The bluntfin model, original and simplified at 50%, 25% and 10%.

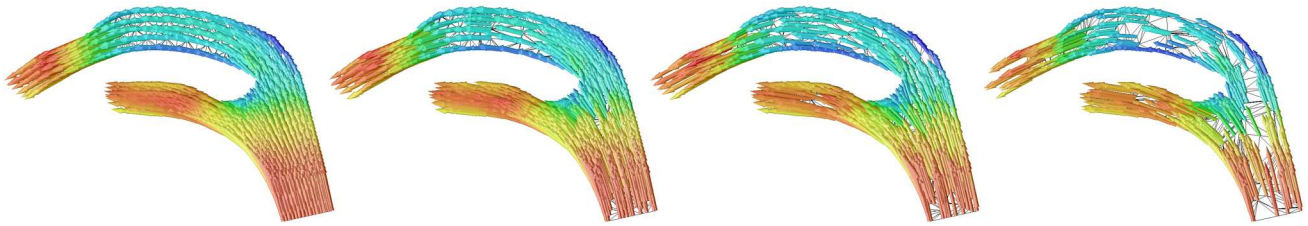

Figure 7. The pipe model, original and simplified at 50%, 25% and 10%.