Visualizing Scalar Fields Represented by Adaptive Square Triangulations

José M. Ribeiro Neves[1]
Ronaldo Marinho Persiano[2]

[1]Computer Science Department, ICEB/UFOP, Ouro Preto, Brazil
jmneves@iceb.ufop.br
[2]System and Computer Science Department, COPPE/UFRJ
C.P. 68511, CEP 21945, Rio de Janeiro, Brazil
marinho@cos.ufrj.br

Abstract. Regular grids representing volume data are usually preferred to triangulation structures because they are simpler to store and faster to process. We consider here a special family of triangulations combining the flexibility of an adaptive representation of fields with some intrinsic regularity. We show that the projection algorithm to visualize scalar fields represented by adaptive square triangulations is far more efficient than to visualize general triangulations and it is competitive with regular grid visualization techniques.

Modern techniques to investigate natural phenomena or to simulate processes often generate large amounts of data, whose analysis is made much easier and enriched by the inspection of two-dimensional images that offer visual appeal and may reveal data features. The representation of scalar fields in 3D spaces is usually supported by a structure of space subdivision into regular or irregular cells. The scalar data are known just at cell vertices and their values inside the cells are defined by interpolation.

Our focus will be targeted on irregular meshes, because irregularity is an inherent condition of the ways data are collected or generated in several applications. In particular, we will treat tetrahedral meshes that compare favorably with respect to regular grid structures: the tetrahedron is the simplest polyhedron and its faces are the simplest polygons, allowing simple data representation and easy implementation of geometrical algorithms. Triangulations admit irregular subdivisions and are more suitable to represent fields with a large second order variation.

On the other hand, fields structured by irregular meshes are hard to visualize due to the expensive geometrical calculations demanded by the irregular nature of their cells. This work explores an intermediate alternative named adaptive square triangulation, a "quasi-regular" structure that preserves the flexibility of tetrahedral meshes and exibits characteristics of regularity that allow efficient implementation of visualization techniques.

The paper is organized in four sections. The first one briefly introduces the visual model and the projection methods to visualize scalar data. The following section introduces the adaptive square triangulation, describes some of its important properties, and shows how to order its tetrahedra from back to front as demanded by the projection method. The third section is devoted to the discussion of a new projection method to visualize fields on adaptive square triangulation. The last section summarizes the main features of the proposed algorithm and compares it with standard approaches.

1 Models and Methods in Volume Visualization

The former techniques to generate two-dimensional images from volumes of scalar data were 2D cuts and surface fitting (isosurfaces extraction). Both of them are still used but are restricted to convey information relative to an infinitesimal portion of the volume, requiring many images (of cuts or isosurfaces) to provide a global view of the data.

Direct visualization techniques brought broader possibilities to a single image that turned out to have the potential to present information from the volume as a whole. The process of generation of such images involves the calculation of the contribution of each cell to the color of each pixel. This is accomplished by mapping visual attributes to the domain as a function of the scalar field. The visual model is conceived to mimic the effects of light emission [Sabella 88], absorption and reflection [Max 95] providing a means of representing the field by a colored image. However, its main objective, instead of simulating "reality", is to provide fast, flexible, and effective tools to data analysts for searching and discovering relevant properties of the data.

1.1 A Visual Model of Scalar Fields

The volume is considered as a cloud of particles whose density varies linearly with the scalar field. Light absorption and emission by the cloud are proportional to its local density. Consider a ray $r(s) = bs + a(1-s)$ crossing the volume, entering at a point a and leaving it at a point b. The light intensity $I(s)$ at point $r(s)$ in the ray may be expressed by a differential equation $dI(s) = \tau(s)(C(s) - I(s))ds$, where $\tau(s)$ is the *attenuation coefficient* and $C(s)$ is the *chromaticity*, responsible for the light emission. Both the attenuation coefficient and chromaticity are supposed to be proportional to the density of particles: $\tau(s) = k_1 F(s)$ and $C(s) = k_2 F(s)$, where $F(s)$ is the scalar field at point $r(s)$.

Defining the *transparency* between the points $r(s)$ and $r(t)$ by $T(s,t) = \mathrm{e}^{-\int_s^t \tau(u)du}$, the solution of the differential equation yields:

$$I(s) = I_0 T(0,1) + \int_0^s C(t)\tau(t)T(t,1)\,dt \qquad (1)$$

where I_0 is the back light intensity at a in the direction of the ray. For polychromatic systems, equations like 1 hold for each relevant wavelength, e.g. red, green and blue. In this case, three components I_r, I_g, and I_b are calculated from the six transfer functions, τ_r, τ_g, τ_b, C_r, C_g, and C_b.

When the segment $[a,b]$ of the ray is subdivided by the points $a_i = bs_i + a(1-s_i)$, with $0 = s_1 < s_2 < ... < s_n = 1$, the light intensity at point a_i may be expressed by:

$$I(s_i) = I(s_{i-1})\,T(s_{i-1}, s_i) + E(s_{i-1}, s_i)$$

where $E(s,t)$ is the light emission of the segment $[r(s), r(t)]$. Assuming that C and τ are constant in each segment, the last relation is simplified to:

$$I(s_i) = I(s_{i-1})(1 - \alpha_{i-1,i}) + C_{i-1,i}\alpha_{i-1,i} \qquad (2)$$

where $\alpha_{i-1,i} = 1 - T(s_{i-1}, s_i) = 1 - \mathrm{e}^{-\tau_{i-1,i}||a_{i-1} - a_i||}$ is referred to as the *opacity* of the segment.

The last relations are often used when the data volume is represented by a cell structure. When τ and C have a low slope in each subdivision cell, we may adopt equation 2 as a fair approximation of the light intensity reaching a point in the projection plane. The values of τ and C in equation 2 are usually taken as an average of the corresponding function values at the intersection points of the ray with the cell faces. As the integration limits must cover the whole ray, the partial results from the cells are composited from back to front by the equation 2.

The effect of light reflection is included in the model to highlight isosurfaces without explicitly constructing any polygonal approximation. The contribution of light reflections can be treated as an increment to the emitted intensity, evaluated just at those cells intercepted by the relevant isosurface. See [Max 95] for details.

1.2 A Review of Direct Visualization Techniques

The direct visualization methods are classified as: object order methods (projection methods) and image order methods (ray casting). In the projection methods, the volume is traversed in a convenient order, such that when a cell is visited all cells covered by it have already been visited. For each visited cell, the method determines its contribution to the color of each pixel covered by the cell projection. In contrast, the image order technique determines for each pixel the cells affecting its color.

This paper focuses on the projection methods, searching ways to take advantage of the regularity of some adaptive triangulations, in order to achieve a procedure faster than processing generic triangulations. The projection method has two outstanding features. First, it allows a more effective exploitation of the spatial coherence inherent to the projection of a cell over several neighboring pixels, providing faster algorithms benefiting from computer-graphics techniques implemented in hardware. Second, as the cells are projected from back to front, the image is generated by overlaying layers, allowing the observation of regions that will be obscured later on. One disadvantage of the method is the artificial mechanism that makes use of the graphics hardware.

Many recent works are devoted to projection methods. In the field of regular grids, we should mention [Westover 90], [Wilhelms 91] and [Lacroute 94]. On the other hand, [Max 90] and [Shirley 90] are two fundamental works focussing on irregular meshes. In the former of these two, Max et al. present a procedure intended to rasterize each cell and to evaluate its contribution to each pixel by analytic integration. High-precision images are produced at the expense of higher computational efforts.

In the second work, P. Shirley and A. Tuchman establish an approximation algorithm, replacing the rasterization of volumetric cells by rasterization of polygons, a much cheaper process already available in commercial hardware. The idea is to mimic the effect of polyhedral cells by semi-transparent polygons that fill the region occupied by the cell projection. Each of these polygons shares just an edge with each neighbor and receives parameters of color and opacity in its vertices. These values are determined in such a way that, when the polygon is inserted in the composition process to generate the image, the effect achieved is similar to what would result from the corresponding part of the cell. This approach, that is extended and refined by the works of Williams [Williams 92], Max et al. [Max 93] and Stein et al. [Stein 94], is one of the foundations of this work, and so will receive special attention in the next subsection.

A third approach to projection methods applied to irregular meshes is due to Yagel et al. [Yagel 96]. Given the projection direction, the vertices of the mesh are transformed to the space of the image and the mesh is intercepted by a series of planes parallel to the projection plane. The result is a two-dimensional mesh of polygons that are rasterized and composited to produce the final image. The intensive use of graphic hardware resources for transformation, rasterization, and composition leads to a very efficient technique. The paper includes extensions to provide adaptivity and to allow image generation with progressive quality.

1.3 A Projection Algorithm for Triangulations

The method proposed by [Shirley 90] considers tetrahedral cells, into which the scalar field is linearly interpolated from its values at the cell vertices. The attenuation coefficient and chromaticity are assumed to be proportional to the field and varying linearly inside the cell. Each cell is projected onto the screen in visibility order, from back to front, as a semitransparent mass (opacity) emitting light (chromaticity). The image is progressively formed in such a way that each cell projection changes part of the previous image by modifying the pixels in its projection. The new colors of the pixels are combinations of their original colors (filtered by the mass thickness) with the light emission of the cell.

The detailed steps of the method are described below.

Projection Build a list of projections of the triangulation vertices.

Sorting Sort the triangulation tetrahedra from back to front.

For each tetrahedron in the sorted list do:

Classification Classify the projection of the tetrahedron according to the profile types illustrated in fig. 1.

Decomposition Decompose the projection into triangles according to its type and determine the thickness of the tetrahedron at the point T, the point of maximum thickness. Determine T_0 and T_1, the extreme points of the intersection segment of the tetrahedron with a projection ray passing through T.

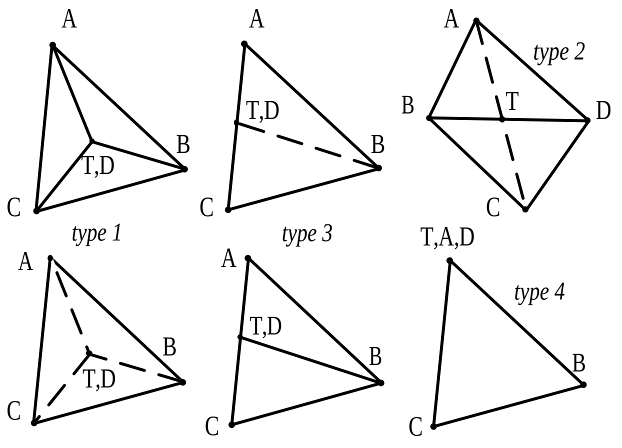

Figure 1: Possible projections of a tetrahedron with vertices A, B, C and D.

Calculation of Chromaticity Determine the chromaticity at the vertices of the decomposition. The chromaticity $C(T)$ at the point T is approximated by the mean value of $C(T_0)$ and $C(T_1)$.

Calculation of Opacity Evaluate the mean value $\bar{\tau}$ of $\tau(T_0)$ and $\tau(T_1)$. Evaluate the opacity at point T by the formula $\alpha(T) = 1 - e^{-\bar{\tau}L}$, where L is the maximum thickness $||T_1 - T_0||$. The opacity at the other vertices of the decomposition is zero.

Display Display each triangle of the decomposition by combining its color with the current image by the formula:

$$I_n(p) = I_{n-1}(p)(1 - \alpha(p)) + C(p)\alpha(p) \qquad (3)$$

where $I_{n-1}(p)$ and $I_n(p)$ are respectively the values at the pixel p of the previous and the new image and $\alpha(p)$ and $C(p)$ are respectively the opacity and chromaticity at p interpolated from the values at the vertices of the corresponding triangle.

Remarks For non degenerated Delaunay triangulations, the sorting order coincides with the depth order of the centers of the circumsphere of the tetrahedra. In general triangulations, the sorting may require that some tetrahedra be subdivided in order to break cycles of overlapping cells ([Williams 92]). The classification step is very time consuming, demanding several geometrical comparisons and evaluations. The approximation of formula 3 is fairly good if there is a low variation of the field inside the cell. The display according to that formula benefits from features

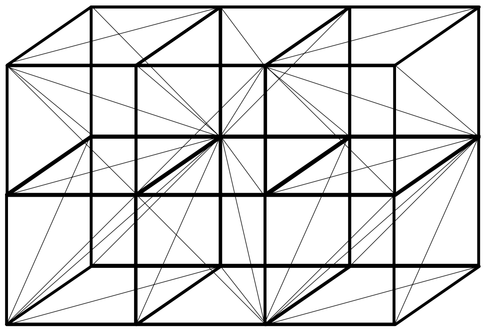

Figure 2: J1 triangulation of an array of 2x3 cubes.

of commercial hardwares (called *alpha blending*) or graphic libraries (like OpenGL).

2 Adaptive Square Triangulations

This section introduces a special family of triangulations that allow many shortcuts in their processing by the projection method. We will consider in the sequel simplicial subdivisions of regular arrays of cubes. A 3D cube may be subdivided in 6 similar tetrahedra by three planar cuts through one diagonal and one edge of the cube. From a regular subdivision of the space in cubes, a regular triangulation of the same domain may be built by the subdivision of the cubes. If the simplicial subdivision of a cube is (recursively) reflected by its faces to its neighbors, a regular triangulation of the space (usually called J1 [Persiano 93]) is assembled (see Fig. 2), whose cells are of the same particular shape (type A shape). The combinatorial and geometrical properties of J1 triangulations have benefited many applications ([Allgower 87], [Castelo 90], [Persiano 94]). They are Delaunay triangulations of their vertices.

Simple general simplex bisections lead to regular refinements of J1 triangulations. A *halving bisection* of a simplex will be attained by splitting it with the plane passing by the midpoint of its longest edge and containing the opposed edge. If all simplices of a J1 triangulation are bisected, a finer regular triangulation (called R1) is obtained whose simplices have a distinct shape (type B shape). An even finer and regular triangulation (called S1) with simplices of a third shape (type C shape) emerges if all simplices of an R1 triangulation are bisected with halving bisections. The bisections of all simplices of an S1 triangulation produce a finer J1 triangulation (with type A shape). Simplices of those three types have good shape in the sense they are relatively "fat" simplices (refer to [Persiano 93]).

2.1 Adaptive Triangulations

A 3D *square triangulation* is any triangulation in 3D space whose simplex shapes are of type A, B, or C. The restricted face shapes of simplices in a square triangulations impose restrictions to its combinatorics. An even more restricted family includes only the square triangulations obtainable from J1 triangulations by successive refinements resulting from the application of the halving bisection to some simplices as described below. Triangulations in this special family will be called *adaptive square (AS) triangulations* [1].

A few definitions allows a better understanding of the refinement process of AS triangulations. The *star of an edge* is the set of all simplices in the triangulation sharing that edge. A *nucleus* of an AS triangulation is a star of an edge which is the longest edge of all simplices in the star. In a J1 triangulation, any set of simplices belonging to the same cube is an example of a nucleus. The simplest refinement of an AS triangulation results from doing halving bisections of all simplices of one of its nuclei (a *nucleus refinement*). Any AS triangulation is obtainable from a J1 triangulation (which is called its *root triangulation*) by successive refinements of nuclei (see next subsection and Fig. 3).

As a consequence of the intrinsic construction of AS triangulations, they have many interesting properties, some of them listed below[2]:

- any simplex of a given type is an affine transformation of any other simplex of that type by a translation, a scale transform, followed by a "reorienting transform";
- the scale factor of any scale transform is a power of 1/2;
- any reorienting transform is composed of one or two reflections relative to one of the 9 planes that contain 4 vertices of a cube of the root triangulation;
- for each simplex type there is a total of 24 distinct reorienting transforms.

By choosing one simplex of each type in the J1, R1, and S1 configuration of the highest level as a reference for that type, we can characterize any simplex in an AS triangulation by a level (the exponent of

[1] AS triangulations were formerly named *adaptive CFK triangulations* in [Persiano 93]

[2] It is beyond the scope of this paper to justify those properties although they are based on unpublished material. We will restrict ourselves here only to their implications on the visualization algorithms.

its scale factor relative to the reference simplex) and orientation (the reorienting transform relative to the reference simplex). Considering the three types, a total of 72 distinct orientations of simplices may be found in an AS triangulation.

2.2 Adaptive refinement

Binary refinements of AS triangulations based on halving bisections of simplices may profit from the good shape of the generated simplices; that is an essential feature for approximation purposes. A basic procedure to refine AS triangulations by simplex bisections was proposed by [Persiano 93]. By just using simplex bisections, the *basic refinement* may be expressed recursively by (see Fig. 3):

```
Basic Refinement of S:
Let E be the longest edge of S;
Step 1:
  while there is a simplex S' in the star
        of E with an edge longer than E
     do a Basic Refinement of S' ;
Step 2:
  do halving bisections of
     all simplices sharing E.
```

The basic refinement procedure has some interesting properties (see [Persiano 93]). For example, when step 2 is executed, the starting simplex S is part of a nucleus. So, nucleus refinement is the only topological procedure needed by the refinement process which assures that the triangulation structure is always preserved. Nucleus refinement and simplex removal are the only procedures needed for housekeeping in a data structure designed for the representation of all AS triangulations.

Regular representation of scalar fields are often built from scattered data by interpolation methods. Similar techniques can be used to represent scalar fields by AS triangulations. The refinement scheme described in [Persiano 93] relies on the adaptive refinement above and refinement criteria to build a virtually optimal triangulation adapted to a given problem. Such a scheme supported by a suitable criterion is a good starting point to create AS triangulation structures modeling scalar fields from scattered data.

2.3 Hierarchical Decomposition and Directional Ordering

AS triangulations may be regarded as an array of BSP (binary space partition) trees. Each simplex in the root triangulation of an AS triangulation is the root of a BSP tree that describes all the successive bisections applied to simplices during the refinement

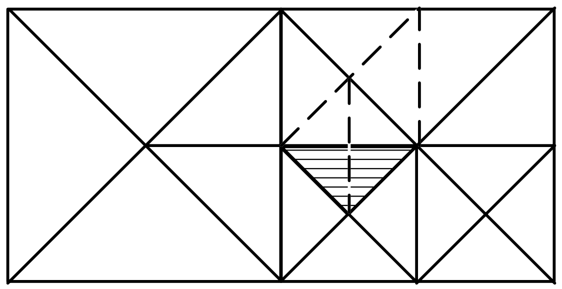

Figure 3: The basic refinement applied to the hatched triangle of a 2D AS triangulation. The dashed lines represent the three nucleus refinements.

that led to the AS triangulation. To avoid confusion, we name *sectors of a triangulation* the simplicial regions that are internal nodes of the BSP tree (but not a simplex of the triangulation), reserving the term "simplex" to refer to the triangulation cells (which are leaves of the BSP tree).

In any BSP tree of an AS triangulation, the splitting plane of a sector is implicitly defined by its own geometry, so the explicit storage of the plane in the node (necessary in general BSP trees) may be avoided. Thanks to the rigid rules of sector splitting, the orientation of the two sons of a sector may be determined by the orientation of the sector, defining a relation easily retrievable from a look-up table.

The BSP representation of AS triangulations is valuable to support many algorithms to solve searching (like finding the simplex that contains a given point) and traversing problems. A simple algorithm guided by the BSP tree may be devised to provide an ordered traversal of the AS triangulation. The directional order is the basis for the projection algorithm to visualize scalar fields.

Given a 3D direction d, the *directional order* defined by d is a partial relation order $\succ$ such that two simplices T^1 and T^2 satisfy $T^1 \succ T^2$ if the interior of T^1 and a translation of T^2 by λd are disjoint for any $\lambda > 0$. That is, $T^1 \succ T^2$ if T^2 projects itself in the direction d "behind" T^1. The *ordered traversal* of a triangulation in the direction d is any traversal of the triangulation such that a simplex is visited before any greater (according to the relation $\succ$) simplex.

It is well known that not every triangulation allows a directional order for every direction ([Max 90]). If an order does not exist, some simplices should be split to allow a correct projection. AS triangulations always yield a directional order for any direction.

To do an ordered traversal of an AS triangulation, do an ordered traversal of the simplices of its root J1 triangulation, visiting each of its simplices by an ordered traversal of the corresponding BSP tree. J1 triangulations are Delaunay triangulations, which always allow an ordered traversal ([Max 90]). Each sector of an AS triangulation allows a directional order: the two sons of the sector may certainly be ordered by the relation $\succ$ and the tree of the smaller son should be traversed first. The small number of orientations of simplices of an AS triangulation suggests that the traversals of the BSP trees are driven by a table storing for each father's orientation whose son should be traversed first. This table is built based on the given ordering direction d.

3 A Projection Algorithm to Visualize Data Structured by an AS Triangulation

Visualization of volume data structured by regular grids of cubes benefits from regularity mainly because all subdivision cells (hexahedra) have the same shape and are regularly arranged in space. In the projection algorithm, all intermediate calculations that depend only upon the geometry of the cell projections may be pre-calculated reducing the cost of the repetitive process of projecting each cell of the array. Besides, regular grids are easily sorted according to the direction of projection ([Wilhelms 91]).

General triangulations require more computational effort on the projection algorithm because each cell usually has a distinct shape. Besides, not all 3D triangulation allows a directional order and, to overcome this, additional subdivisions of tetrahedra are necessary.

AS triangulations are an intermediate instance between regular grids and general triangulations. As stated in section 2, AS triangulations contain tetrahedra of at most 72 orientations. The projection of tetrahedra of an AS triangulation depends on its position, its level (size), and its orientation. The first two aspects affect the projection just in position and scale. The orientation affects not only the shape of the projection but also the evaluation of the opacity and chromaticity of critical points. Considering the small number of orientations, many intermediate values may be calculated in an initialization step and stored in tables. In the following, we describe the main data structure that helps the projection algorithm for AS triangulations.

The Orientation Catalog It is a table describing all geometrical features depending only on the orientation of a tetrahedron. For each given orientation, the catalog associates a standard tetrahedron with that orientation and stores the orientation of its two sons and references to its vertices and face plane normals. A list of the vertex coordinates and a list of plane normals complement the catalog. A total of 15 vertices and 18 normals are referenced in the table. The orientation catalog is fixed and independent of the specific triangulation, of the volume data, and of the visualization parameters.

The Volume Data It is represented by an AS triangulation and a table of triangulation vertices. The triangulation structure may be stored as an array of BSP trees. The array has one entry for each tetrahedron in the root J1 triangulation of the AS triangulation. The inner nodes of the trees contain just references to its two sons. Each leaf of a tree is a tetrahedron of the AS triangulation and stores its orientation, its level of refinement, and the references to its four vertices. Each entry of the table of triangulation vertices contains the vertex coordinates and the scalar field value at the vertex.

The following structures depend upon the direction of projection.

The Guide to BSP Tree Traversal The guide is a table built from the orientation catalog for the given direction of projection. For each possible orientation, the table indicates to the traversal algorithm which son (of a tetrahedron with that same orientation) should be traversed first. To build the table, for each orientation, the corresponding standard tetrahedra is retrieved from the catalog and its two sons are compared with the direction to decide which one is hidden. This requires geometrical tests.

The Projection Catalog It contains a list of the projections of all triangulation vertices and a table of projection data of the standard tetrahedra of the orientation catalog. For each possible standard tetrahedron, indexed by its orientation, the corresponding entry of the table stores the type (according to Fig. 1) of its projection, the decomposition of its projection in triangles and its maximum thickness. For each vertex of the decomposition, the table also stores the reference to the corresponding vertex of the tetrahedron or the baricentric coordinates of the tetrahedral boundary points projecting on that 2D vertex. The catalog is built by applying the standard procedures of the projection method described in section 1.3 for each standard tetrahedron in the orientation catalog.

3.1 The Algorithm

Given the direction of projection, the initialization step of the algorithm builds the guide to BSP tree traversal and the projection catalog. Then, all tetrahedra in the triangulation are processed from back to front by traversing the triangulation driven by the guide without any geometrical calculation. The steps to process a tetrahedron are:

Projection Shape Evaluation Retrieve from the projection catalog all data associated to the orientation of the tetrahedron; scale the maximum thickness according to the level of the tetrahedron; retrieve from the projection catalog the projection of the vertices of the tetrahedron; according to the projection type, use the baricentric coordinates in the projection catalog (if any) to calculate the scalar field values at the tetrahedral boundary points projecting onto the point of maximum thickness.

Chromaticity Evaluation Retrieve the scalar field values at the vertices of the tetrahedron and calculate the corresponding chromaticity at the vertex projections and at the point of maximum thickness;

Opacity Evaluation Determine the opacity at the point of maximum thickness;

Display Display the triangles in the decomposition of the tetrahedron projection with the corresponding chromaticity and opacity.

3.2 Privileged Directions

The singular cases of projections of type 3 and 4 in fig. 1 demand less geometrical calculations. The projection of tetrahedra in an AS triangulation presents more singularities when the direction of projection coincides with the direction of an edge of some AS triangulation tetrahedra. There are 26 such special directions.

Taking one special direction as the direction of projection, all tetrahedra with an edge in that direction will have a projection of type 4 (see fig. 1 and fig. 4) and will be called here of class q_1; the tetrahedra of class q_2 have a face containing the direction and will project symmetrically as type 3; the tetrahedra of class q_3 will project as a type 2 diamond shape; those of class q_4 are the remaining tetrahedra projecting on a quadrilateral; and those of class q_5 project on a triangle.

To analyze the benefits of a special direction, we crossed all 72 standard tetrahedra against each possible special direction and found the following incidences of the 5 classes of projections:

q_1	q_2	q_3	q_4	q_5
864	576	144	240	48
46.1%	30.8%	7.7%	12.8%	2.6%

The predominant class q_1 requires no interpolation of field values; the second class q_2 requires just one

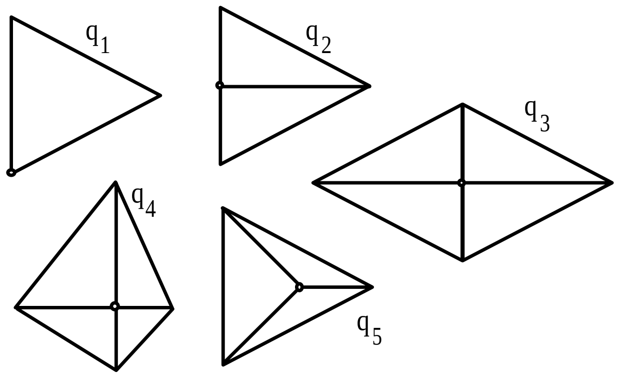

Figure 4: The 5 possible projections of AS triangulation tetrahedra for special directions.

interpolation; the remaining classes demand more expensive calculations but they are much less frequent.

[Williams 92] reported the incidence of around 40% of projections of type 1 (equivalent to class q_5) and around 60% of projections of type 2 (equivalent to classes q_3 and q_4) in his experiments with arbitrary directions and generic triangulations. The incidence of singular projections were very scarce. We should expect similar proportions with AS triangulations projected in arbitrary directions, suggesting that the use of the special directions should reduce significantly the processing time. A practical way to explore that peculiar character of the structure is to offer to the system user a faster special mode restricting his observations to the 26 special directions. He can switch to the free interaction mode when a more precise observation is needed.

4 Conclusions

To compare the proposed method with the standard projection method for general triangulations, it is convenient to distinguish three phases in the process: the directional ordering, the projection of tetrahedra, and the rasterization for display. The directional ordering demands a traversal of all BSP trees, requiring a fixed number of non arithmetic operations for each sector. So, the order of this process is the number of tetrahedra of the triangulation, being comparable to the ordering of a regular grid with the same number of cells and significantly less demanding than the sorting of general triangulations.

Due to the pre-calculation of critical parameters, AS triangulations present clear advantages over general triangulation in the projection phase, including the steps from classification to calculation of the opacity at the maximum thickness point. Nevertheless, it is worth it to compare them by experimental tests from implementations in order to quantify the speed-up. The rasterization task is accomplished by hardware and takes only a small amount of the processing time.

[Williams 92] reports a rate of 1:4 between the times spent by ordering and projection in the general projection method. These are the critical phases unavoidably attributed to the software. This observation increases the relevance of the proposed interactive scheme with privileged directions, whose supplementary gains are incident over 80% of the total time of software processes. As a general feature of benefiting from hardware polygon rendering, it is relevant to point out that projection method efficiency is dependent on the number of pixels covered by the projection of each cell. The larger this number is, the stronger the advantage; this indicates their best usage to treat medium size data volumes or to zoom in large volumes.

It should be noted that AS triangulations do not guarantee to position the nodes over irregular boundaries. This characteristic constitutes its main limitation, especially in those cases where the critical regions are located on the boundaries. Although this problem is weakened by the inherent adaptivity of the structure, it remains as a potential restriction to its use. In general, a prospective application of the described triangulation is to replace regular grids as auxiliary structure for visualization, profiting from its nature of (quasi) regularity and adaptivity, which offers a good compromise between efficiency and quality of representation.

Acknowledgment This research is part of GEOTEC, a project supported by PROTEM/CNPq. It is also part of the research for the Doctoral degree of José M. Ribeiro Neves who was sponsored by a scholarship of PICD/CAPES.

References

[Allgower 87] E. Allgower and S. Gnutzmann: "An Algorithm for piecewise-linear approximation for an implicit defined two-dimensional manifolds". *SIAM Journal on Numerical Analysis*, 24,2,1987.

[Castelo 90] A. Castelo, S. Freitas and G. Tavares: "Approximation to manifolds and its application to implicit ODES". *Lectures in Applied Mathematics*, 26 (29), AMS, 1990.

[Lacroute 94] P. Lacroute and M. Levoy: "Fast Volume Rendering Using a Shear-Warp Factorization of the Viewing Transformation", *Proceedings of Siggraph'94*, pp. 451-457, 1994.

[Max 90] N. Max, P. Hanrahan and R. Crawfis: "Area and Volume Coherence for Efficient Visualization of 3D Scalar Functions", *Computer Graphics*, vol. 24, n. 5, pp. 27-33, 1990.

[Max 93] N. Max, B. Becker and R. Crawfis: "Flow Volumes for Interactive Vector Field Visualization", *Proceedings of Visualization'93*, pp. 19-25, oct. 1993.

[Max 95] N. Max: "Optical Models for Direct Volume Rendering", *IEEE Transactions on Visualization and Computer Graphics*, vol. 1, n. 2, pp. 97-108, 1995.

[Persiano 93] R. M. Persiano, J. L. Comba, and V. Barbalho: "An adaptive triangulation scheme and construction". *SIBGRAPI 93*, Recife, 1993.

[Persiano 94] R. M. Persiano and A. Apolinario: "Boundary Evaluation of CSG Models by Adaptive Triangulation". *CSG94 Set-teoretic Modeling Techniques and Applications* , Information Geometers, Winchester, 1994.

[Sabella 88] P. Sabella: "A Rendering Algorithm for Visualizing 3D Scalar Fields", *Computer Graphics*, vol. 22, n. 4, pp. 51-58, 1988.

[Shirley 90] P. Shirley and A. Tuchman, "A Polygonal Approximation to Direct Scalar Volume Rendering", *Computer Graphics*, vol. 24, n. 5, pp. 63-70, 1990.

[Stein 94] C. Stein, B. Becker and N. Max: "Sorting and Hardware Assisted Rendering for Volume Visualization", *ACM Volume Visualization Symposium'94* , pp. 83-89, 1994.

[Westover 90] L. Westover: "Footprint Evaluation for Volume Rendering", *Computer Graphics*, vol. 24, n. 4, pp. 367-376, 1990.

[Wilhelms 91] J. Wilhelms and A.V. Gelder: "A Coherent Projection Approach for Direct Volume Rendering", *Computer Graphics*, vol. 25, n. 4, pp. 275-284, 1991.

[Williams 92] P. L. Williams: "Interactive Direct Volume Rendering of Curvilinear and Unstructured Data", PhD Thesis, *University of Illinois at Urbana-Champaign*, 1992.

[Yagel 96] R. Yagel, D. M. Reed, A. Law, P. Shih and N. Shareef: "Hardware Assisted Volume Rendering of Unstructured Grids by Incremental Slicing", *Proceedings 1996 Symposium on Volume Visualization*, San Francisco, CA, sep. 1996, pp. 55-62.