Research Journal of Applied Sciences 10 (8): 436-441, 2015
ISSN: 1815-932X


# Real-Time Rendering of Dynamic Fluid Jets

Nikolay V. Spiridonov
Kazan (Volga Region) Federal University, Kremlevskaya Street 18, Kazan, Russia

**Abstract:** This study deals with the problem of an algorithm development and optimization that allows to visualize the surface of the liquid jet in dynamics by approximating its triangles. This is so-called problem of triangulation. There are many algorithms to solve such problems but most of them can not reach the speed necessary for real-time applications. The algorithm should be used in extremely loaded simulation environment and the specificity is that the program should be called in every frame. In addition to all this, the environment has a lot of other logic and calculations. This imposes the significant restrictions during operation time, the number of generated triangles and the overall complexity of the algorithm. A set of points is supplied to the program input, the global coordinates which are the centers of the liquid particles. We should get 3D object data which contains the information about the vertices forming the desired surface, the method of their decomposition by triangles, the textural coordinates of the vertices and the normal values at the point. The result is that the original problem is divided into two independent sub-tasks: the calculation of the function value forming the liquid jets at each point and the triangulation of three-dimensional scalar field. These problems are solved with the help of different approaches that in combination with parallel technologies and optimization methods allowed to achieve the high efficiency of the algorithm and use the visualization algorithm of the physics engine successfully and the simulation environment associated with medicine.

**Key words:** Triangulation, real-time rendering, visualization of fluid, computer graphics, three-dimensional

## INTRODUCTION

The problem of surface visualization or a scalar field is very important and is met in many areas: robotics, physics, mathematics, medicine, cybernetics, mapping (Bugrov *et al*., 2012; Gueziec, 1995; Semenihin, 2004; Delone, 1934; Dizhevsky, 2011, 2007; Evruhimov *et al*., 1999; Kozlov and Turlapov, 2010). Here are some application areas.

Triangulation of data obtained using different sensors, instruments mounted on a robot in order to simplify the further processing, computing, presentation convenience, position search (similar to the task of a mobile phone location search with the help of the nearest telephone stations) (Fig. 1).

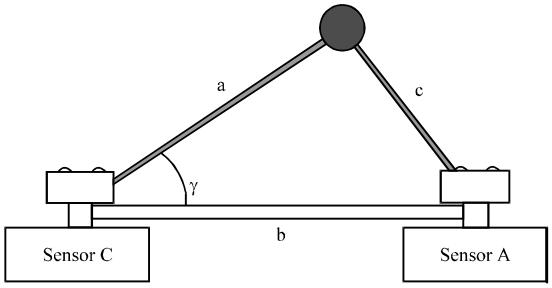


Fig. 1: Trivial scheme of location determination

**Experimental data visualization:** When you conduct physical experiments there is often the need to embrace and display the information from all sensors at once. For example, the region with a predetermined temperature, composition and so forth.

**Functional performance:** In some mathematical problems and calculations it is necessary to visualize, the geometrical object that is defined by a real function (or multiple functions set by piece, the specific example of this will be discussed below) of several variables in the form F (x) = const. (The problems may occur when the original formulation from the right also has a function but the case is obviously reduced to the offered one). Also, a common statement may appear in which the function setting a field is specified by a set of points where we know its value (the most common case are the readings of a particular sensor as the unit function) (Fig. 2). Where:

$$f(x, y, z) = (1-(x/6.5)^2-(y/4)^2)((x-3.9)^2+y^2-1.44)((x+3.9)^2+y^2-1.44)-z^8$$

The increase of computing powers for computers also gave the impetus to the development of new areas for medical research for example, computer tomography, magnetic resonance imaging, positron emission

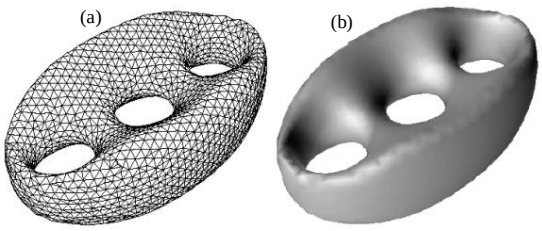


Fig. 2: Functional representation. The object SRT by the function f(x, y, z) = 0 is visualized

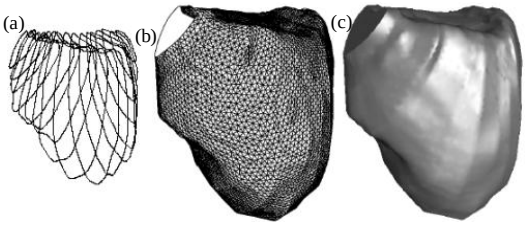


Fig. 3: The process of a bladder image restoration from the resulting cross sections during the ultrasound results study

tomography and so on. With the help of special scanners, the images of individual organs or body parts in a cross-section are performed. They characterize the physiology peculiarities or the intervention result. The imaging techniques allow you to restore, the three-dimensional structure of an object under survey according to the array of parallel cross-sections of an object. In most cases, an expert should have a full picture of the organ under investigation and should not consider individual sections. Of course at the time of imaging, we can not get the information that we did not take from sensors but this greatly simplifies the decision-making, increases the visibility of information and in some cases eliminates the need for immediate surgery (Semenihin, 2004) (Fig. 3 and 4). Also, triangulation is used to create custom maps of locations and geological prospecting (Fig. 5).

**Classical setting of an issue:** The visualization of three-dimensional scalar fields means the surface visualization set by the function of three variables and a fixed value of this function level:

$$\{(x, y, z)|f(x, y, z) = c\}$$

Where:

f (x, y, z) = The given function

c = The predetermined level

The set of points that satisfy this condition is the desired surface. We will develop not the surface itself but

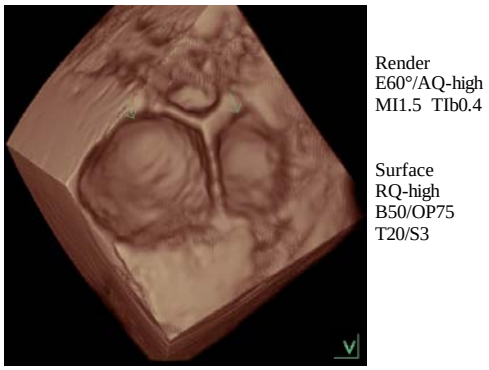


Fig. 4: The 3D reconstruction of the right ovary with maturing follicles

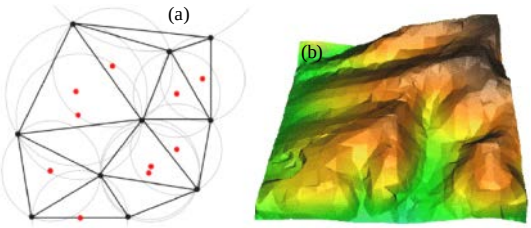


Fig. 5: Obtained image of the area surface

the approximating desired quantity using the triangles. This imaging method is called triangulation.

Such formulation of the problem is well studied. It offers a great number of methods solving this problem or a similar one. The most famous of them are the following: Caneira Method and Skala (2000) Method, the marching cubes. These methods are referred to the class of cell techniques. There are other classes of algorithms solving this problem. They will be stated in the appropriate study.

Let's define some conditions which our algorithm should met, let's also specify the object of visualization. We need to visualize the dynamics of some liquid and jet flight. For example, it may be a fountain an arterial blood jet, a flow of water from a faucet. At that the specificity is that rendering must take place in real time. So, we have a very strict time limit concerning the rendering of each frame. It is also worth understand that in addition to our logic of triangulation each frame should have enough time for other more important calculations without the damage for the general imaging and image smoothness (such requirements are dictated primarily by the needs of modern simulation environments which have large time-consuming calculations and sometimes programmers sacrifice quality for speed). Do not forget as well that the computation time and the rate of post-processing by a

graphics card (shader rendering, the imposition of post-effects, interactions with other scene objects) depends directly on the number of generated triangles. So, the basic requirement of the algorithm is high performance speed.

The program input is supplied by the global coordinates of the liquid molecule centers. We have to obtain a 3D object data at output which should contain the coordinates of all triangle vertices, the method of their partitioning, normals and texture coordinates for each vertex. It is clear that an initial problem is divided into 2 independent sub issues. The calculation of the function value at each point of the desired area and the triangulation of a three-dimensional scalar field in the required cells.

## MATERIALS AND METHODS

**Concretization of 3D scalar field:** So, we need to calculate the function values describing the particles of water but there is no explicit formula setting that describes the surface, neither the function values at the grid nodes. From the input data, we have the points that are calculated on the basis of math model. We need to develop a triangulation grid around these centers. The result of this development is the geometrical object called metasphere (Fig. 6).

Let's consider the simplest way to describe this function. We use a piecewise way of a function definition. It is not difficult to guess that we have two areas essentially.

The area in which a desired function behaves like a sphere and in fact is determined by the distance to the nearest point of the input data:

$$x^2+y^2+z^2 = \text{const}$$

The field of deflection in which the distance to the segment connecting the adjacent center points is equal to cos (l) where l is the distance to the nearest point from the input data:

$$R(x, y, z) = \cos(l)$$

R the distance from the point on a surface to the segment connecting the adjacent central points. Thus, we have an object that will continue to approximate and the function specifying this object. There are of course, other ways of such an object setting, however, this object was chosen because of its relative simplicity and the ease of scale change (a spherical region constant is meant) (Fig. 7).

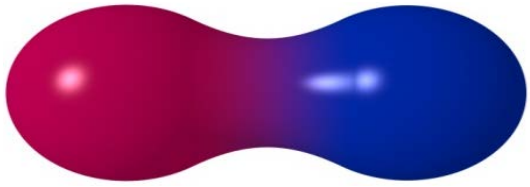

Fig. 6: Example of a two-dimensional metasphere

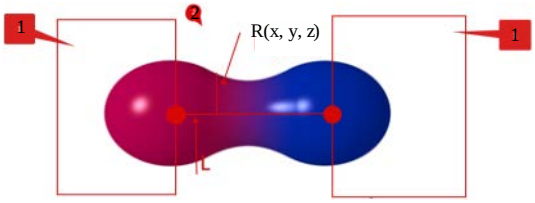


Fig. 7: 1) Spherical area; 2) Deflection area

**Algorithm essence:** The basis of our algorithm is the method proposed by Lorensen (1987). Essentially, the triangulation algorithm may be divided into two stages:

- The split of the minimum area that contains the entry points with the necessary vicinity on a finite set of cells. Then the search for the cells that interest us where the triangulation will be performed
- Approximation of metasphere surface in the desired cells

Let's consider the first stage in more detail. At the beginning, it is necessary to obtain the partition of the area into cells. At this stage, we choose a cube as a partition. Then, the sampling is performed from the entire set of those cells in which the desired function intersects it. We may assume that the surface crosses a cell if there are two vertices such that:

$$F(p_1)<c<F(p_2)$$

Where:

$c$ = The chosen level of the function $F(x, y, z)$

$p_1, p_2$ = The vertex coordinates

Then, the second stage takes place. As we already mentioned a cube was selected as a minimum cell. Let's calculate the number of surface approximation options at this choice of a cell. Suppose, we have a cube which intersects a surface. In this case, we paint with white the vertices, the function value of which is lower than the level and those that are higher with black. It is obvious that the method of triangulation depends on the method

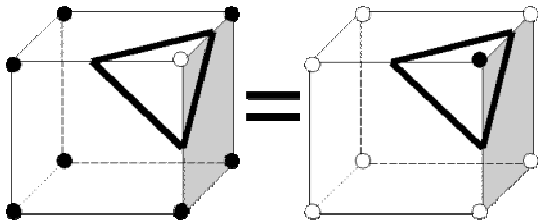

Fig. 8: Illustration of 2 coloring options equivalence

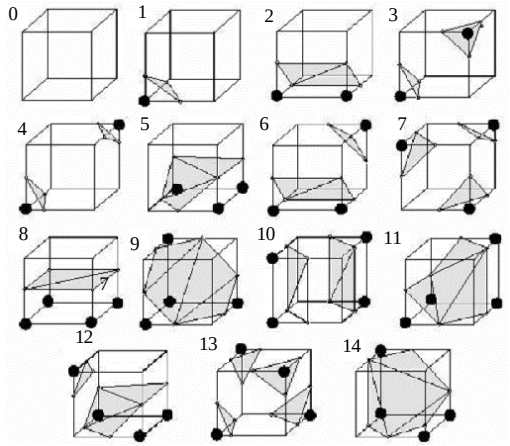


Fig. 9: Triangulation options in a cube

of cell coloring. Then, it is not difficult to calculate that we have $2^8 = 256$ different ways to fill. However, Fig. 8 tells us that 2 triangulation methods 2 are equivalent if they are denying each other. Then, using the symmetry and rotating the shapes, we may reduce 128 options to 14 ones (Fig. 9).

At this moment, we obtained the method of triangulation in a cell but still there is quite a lot of them. So, let's go further and divide the cube into 6 tetrahedrons according to the method proposed by Guezek (1995).

So, we got a tetrahedron as a minimum cell which certainly simplifies triangulation. Let's use the method of counting the number of triangulation methods set forth above and obtain only 3 different options (Fig. 10).

Having obtained the triangulation method a surface may be approximated. At this point, the number of triangles is already known and the cell edges are known for each triangle. You just need to find a place where the function crosses the edge. This can be done by finding the function root, you can also use the method of linear interpolation for two peaks (Fig. 11).

## RESULTS AND DISCUSSION

**Main ways of optimization:** Therefore, we obtained the algorithm that may solve tasks quickly enough. However, there are many ways of its optimization. Here are just a few of them.

Let's pay attention to the fact that triangulation is performed in each cell independently of adjacent ones. It is quite an advantage and allows us to use the technologies of parallel computation: OpenMP, Concurrency Runtime, MPI. But, they all make the calculation by CPU which is in contrast to the graphics card is not able to create a very large number of flows. At that a processor is always overloaded with other calculations that are impossible or very difficult to move to a video card. Therefore, it is recommended to take advantage of NVIDIA CUDA or ATI STREAM technology.

Also, the knowledge of some input data features may give us some advantages. For example, we know that the points of entry may contain many separate areas and the shapes far from each other. You may calculate the triangulation of a sphere according to a pre-designed template that will be certainly faster than the interactive triangulation. You may also divide the input data into shapes that may reduce overhead expenses in the form of certain areas exclusion without the cells crossed by a desired surface. It is possible to delete, the duplicate connections between points or replace them with more convenient ones which also allows us to win some time in the performance.

**Algorithm performance example:** We developed an efficient not too difficult algorithm for programming that is suitable for interactive real-time visualization. Here are two examples of the algorithm for a detailed study of the results.

Figure 12 shows an example of input data obtained artificially. It serves to illustrate the special features of the algorithm. You may notice some separate differing spheres which are calculated according to the template, the computation of which takes time within the error.

In Fig. 13, we have a real example of the program operation. The input has 125 entry points which define the jet. It is understood that the algorithm operation time strongly depends on the input data. We measured an average value of calculation during the operation of a real application and about 2-3 ms were spent for each frame which certainly meets the task.

**Review of alternative approaches to solution:** The proposed method relates to a class of cell techniques. This class is the most developed and most algorithms belong to it. However, there are other approaches to triangulation. The methods of predictor-corrector class are

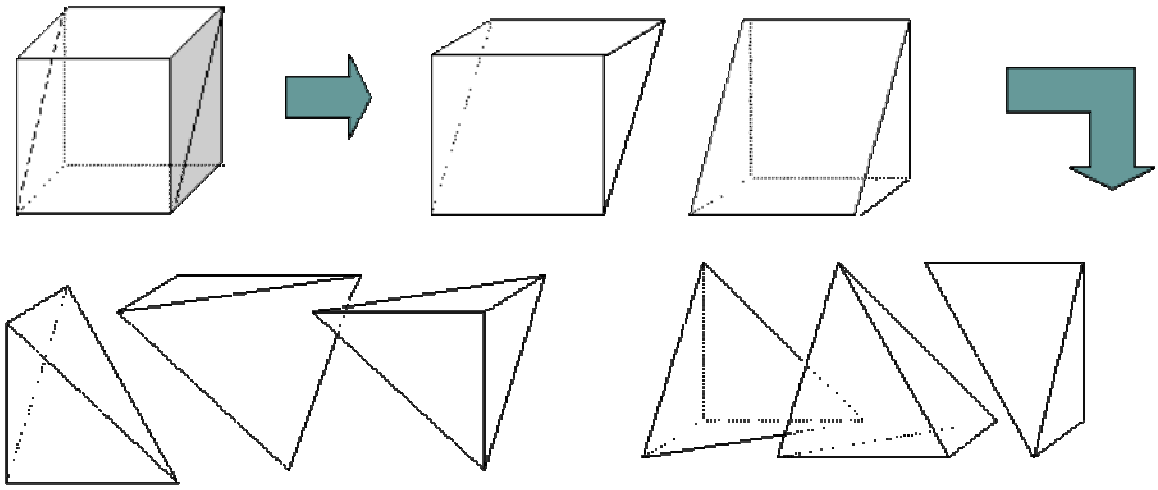

Fig. 10: Cube partitioning method into tetrahedrons

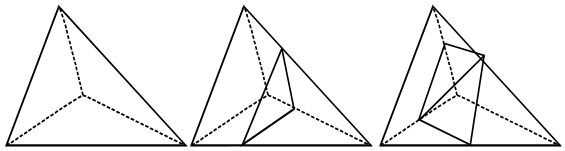

Fig. 11: Triangulation methods in the tetrahedron

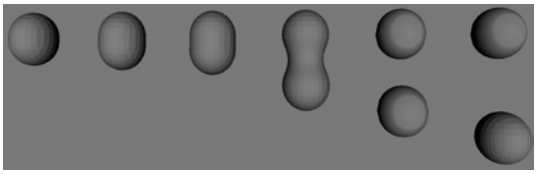

Fig. 12: Artificial example

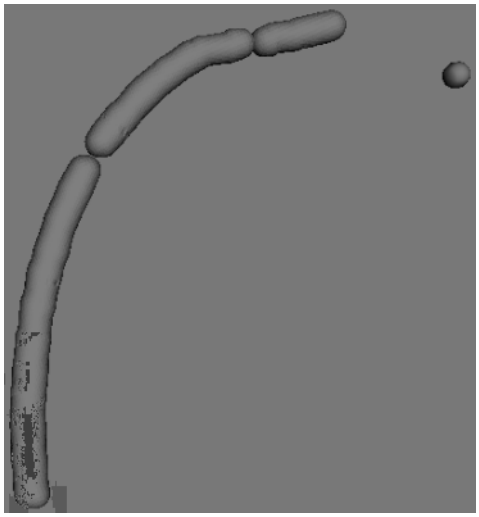

Fig. 13: Example of visualization with real input data obtained on the basis of a physical engine

based on the idea of iteration. The existing set of points at each step is supplemented by another one lying on the tangent plane to the specified function. It is the so-called predicted position. Then, its position is moved to the position to visualized surface, the corrected one. During its use it is necessary to know the value of the function in each point of the desired area and have at least one point belonging to the desired surface at the initial stage. This method allows to obtain a good detailed surface but it is noticeably inferior in speed and is complex for programming and optimization. For example such algorithms are very difficult to parallelize in a classical sense. Of course, you may perform iterations simultaneously in different directions from a point but in this case, we need to communicate with other flows and coordinate their actions. This and many other aspects do not allow the use of this class of algorithms in the applications running in real time.

Another class of techniques is called a mosaic one. The point is to break a required surface in primitives to simplify further triangulation. But in this case, there is quite an obvious problem of such parts connection which certainly is not an advantage.

**Summary:** The program was successfully tested and used in physics engine visualization as well as in a special medical simulation software.

## CONCLUSION

According to the stated above one may conclude that the use of this algorithm allows one to solve the problem of fluid behavior visualization effectively and simply in real time.

**ACKNOWLEDGEMENT**

The research is performed according to the Russian Government Program of Competitive Growth of Kazan Federal University.

**REFERENCES**

Bugrov, N.V., V.I. Golubev, A.Y. Dizhevsky, D.G. Kakauridze, A.S. Klimenko, A.S. Oboymov and P.V. Frolov, 2012. "Review of triangulation algorithms for an implicit surface". International conference MEDIAS2012, Cyprus, Limassol.

Delone, B.I., 1934. About spehere emptiness. Bul. USSR Academy of Sciences. OMEN 4, pp: 793-800.

Dizhevsky, A.Y., 2011. Visualization of three-dimensional objects and geometric aspects of peculiarities detection. Dis. of Tehn. Sciences Candidate.

Dizhevsky, A.Y., 2007. The general approach to the implementation of triangulation methods development for implicit surfaces using the space partitioning into cells. Computational Methods and Programming, 8: 286-296.

Evruhimov, V.L., A.D. Kapustin and A.V. Malashkina, 1999. Implementation of terrain visualization algorithm in real time. Conference materials for computer graphics and visualization "GrafiCon'99". Moscow.

Gueziec, A., 1995. IEEE Transactions on Visualization and Computer Graphics, 1 (4): 328-342.

Kozlov, D. and V. Turlapov, 2010. Surface reconstruction algorithm from the cloud of points by the graphic processor. GrafiCon'2010: 20th International Conference on Computer Graphics and Vision, September, 20-24, 2010, St. Petersburg, Russia. http://www.graphicon.ru/proceedings/2010/conference/RU/Se5/44.pdf.

Lorensen, W.E., 1987. Harvey E. Cline, CG, Vol. 21, No. 4.

Semenihin, A., 2004. Comparative analysis of interactive triangulation methods for grid functions. Computer graphics and multimedia. Issue No. 2 (2). cgm.computergraphics.ru/content/view/63.

Skala, V., 2000. Conference on Scientific Computing 2000, pp: 368-378. http://www.emis.de/journals/AMUC/_contributed/algo2000/skala.pdf.