Multi-Level Ray Casting of Function-Based Surfaces

S I Vyatkin¹, A N Romanyuk², L A Savytska ²

¹Institute of Automation and Electrometry, Siberian Branch of the Russian Academy of Sciences, Academician Koptug Ave. 1, Novosibirsk, 630090, Russia,
²Vinnytsia National Technical University, 95, Khmelnytske shoes, Vinnytsia, 21021, Ukraine

E-mail: sivser@mail.ru

Abstract. The paper deals with the multi-level ray casting of function-based surfaces, a method of defining free forms without approximating them with polygons or patches, issues of using perturbation functions for rendering the surfaces of 3D objects. Free forms based on the perturbation functions have an advantage of spline representation of surfaces, that is, a high degree of smoothness, and an advantage of an arbitrary form for a small number of perturbation functions. Multi-level ray casting allows us to provide both the interactivity and any required level of detail leading to a photo-realistic appearance of the shapes.

1. Introduction
Nowadays, the real-time computer graphics oriented to 3-D scene visualization has attained considerable success. However, even though a sufficiently high realism of real-time scene imaging has been attained, some problems are still present, for instance, it is necessary to store and visualize scenes containing a greater number of polygons than it is implemented in the present-day systems. Analysis of possible directions of evolution of a real-time visualization system shows that the easiest way to improve picture quality, i.e., to increase the number of polygons rendered per frame, is not the most effective one. Along these lines, the qualitative changes are difficult to achieve.

Several representations of geometric objects are currently used in computer graphics. Each of the objects, according to its properties, is used in different fields, beginning from 3-D simulation and CAD systems up to real-time visualization systems.

The functional representation describes most accurately the object geometry and has the smallest size of the required data. Procedures of functional representation demonstrate compact and flexible representation of surfaces and objects that are the results of logical operations on volumes. Its disadvantage is complicated geometrical processing and visualization in real-time.

The paper describes free forms based on the analytical perturbation functions. The aim of this paper is the visualization of objects formed by free-form surfaces, with quick calculation of the ray-surface intersection.

2. Previous Work
One of the main disadvantages of the known visualization methods is complexity of the calculation of points on the surface. Thus, the ray marching method does not guarantee detecting the surface, and, in addition, it is slow [1, 2]. The method of determining the intersection of a ray with an implicitly defined surface is too complex to calculate the L- and G-parameters [3].

In the tracing method, finding the maximum radius when no point of the volume lies within the
sphere is a nontrivial task [4, 9]. Ray tracing with analysis of the interval for complex functions requires individual calculations for each ray and each interval along this ray [5]. In fast tracing, search for the rays intersecting the surface requires a lot of calculations and is not efficient enough as the clustering procedures of this method do not solve this problem completely [6]. A ray tracing method for imaging surfaces defined by algebraic polynomials of high degree is described in [7]. However, it is not easy to model real objects using polynomials. Nor is the accuracy of approximation of the initial function with a Bezier curve guaranteed. Another disadvantage of this method is that transformation of objects to another coordinate system is a complex task. Therefore, the creation of dynamic scenes is problematic.

3. Geometric Objects

It is possible to describe complex geometry forms by specifying the surface deviation function (of the second order) in addition to the surface basic function of the second order [8]. Generally function \( F(x,y,z) \) specifies the surface of the second order that is quadric (see Fig. 1):

\[
F(x,y,z) = A_{11}x^2 + A_{22}y^2 + A_{33}z^2 + A_{12}xy + A_{13}xz + A_{23}yz + A_{44} \geq 0,
\]

where \( x, y \) and \( z \) are spatial variables.

The free form is a composition of the base surface and the perturbation functions:

\[
F'(x,y,z) = F(x,y,z) + \sum_{i=1}^{N} f_i R_i(x,y,z),
\]

where \( f_i \) is the form-factor; perturbation function \( R(x, y, z) \) is found as follows:

\[
R_i(x,y,z) = \begin{cases} 
Q_i(x,y,z), & \text{if } Q_i(x,y,z) \geq 0 \\
0, & \text{if } Q_i(x,y,z) < 0
\end{cases}
\]

Herein, \( Q(x, y, z) \) is the perturbing quadric.

gometric operations

4. Geometric Operations

A special kind of object is defined in the visualization system; the object is used to perform logical operations on objects. Hence, the whole scene is a kind of a tree. Each node of the tree is an object-constructor performing logical operations on its descendants, and vertices of the tree are primitives used by the system. When the object-constructor is queried while rendering, the object addresses its descendants, transforms the obtained results, and gives the answer to the query. The descendant may be a primitive or another object-constructor. While applying the geometric operations, rotations, displacements, and scaling to the object-constructor, it performs all these operations on its descendants, and in addition changes its Boolean function in case of inversion.

Two major types of elements of the set of geometric objects are simple geometric objects and complex geometric objects. A complex geometric object is a result of operations on simple geometric objects [8]. Let objects \( G_1 \) and \( G_2 \) be defined as \( f_1(X) \geq 0 \) and \( f_2(X) \geq 0 \). The binary operation of objects \( G_1 \) and \( G_2 \) means operation \( G_3 = \Phi_1(G_1, G_2) \) with definition

\[
f_3 = \psi(f_1(x,y,z), f_2(x,y,z)) \geq 0,
\]

where \( \psi \) is the continuous real function of two variables.

5. Multi-Level Ray Casting

We used the multilevel ray casting algorithm, which performs an efficient search for volume elements - voxels which participate in image generation. At the first stage of recursion, the initial viewing pyramid is divided into four smaller sub pyramids in the screen plane. At the stage of division of space along the quaternary tree, 2-time compression and transfer by \( \pm 1 \) along two coordinates are performed:
\[ A1' = A11/4; \]
\[ A22' = A22/4; \]
\[ A33' = A33; \]
\[ A12' = A12/4; \]
\[ A13' = A13/2; \]
\[ A23' = A23/2; \]
\[ A14' = A14/2 + i A11/2 + j A12/4; \]
\[ A24' = A24/2 + i A12/4 + j A22/2; \]
\[ A34' = A34 + i A13/2 + j A23/2; \]
\[ A44' = A44 + i A14/4 + j A24/4; \]
\[ A44' = A44 + i A14'/2 + j A24'/2. \]

If in the equation of quadric \( Q(x, y, z) = 0 \), the values of variables \( x, y, z \) vary within length \([-1, 1]\), then

\[ \max \left[ |Q(x, y, z) - A44| \right] \leq \max F = |A11| + |A22| + |A33| + |A12| + |A13| + |A23| + |A14| + |A24| + |A34| \tag{6} \]

We should note that if \(|A44| \geq 0\), probably, point \( M_0 = (x_0, y_0, z_0) \) \((-1 < x_0, y_0, z_0 < 1)\) exists such that \( Q(x_0, y_0, z_0) = 0 \). If \( \max F < |A44| \), then such points do not knowingly exist, and the sign of coefficient \( A44 \) distinguishes location of the pyramid inside or outside with respect to quadric surface \( Q=0 \) (if \( A44 \geq 0 \), then the sub pyramid is inside the quadric). Using results of this test, we perform division of sub pyramids that fall within the quadric completely or, probably, partially, and the knowingly external sub pyramids are eliminated from processing. A test for intersection of sub pyramids with a freeform is somewhat different. For the basic quadric, the test for intersection looks as follows:

\[
\text{if } ( ( A44+R ) < 0 ) \&\& ( |A11|+|A22|+|A33|+|A12|+|A13|+|A23|+|A14|+|A24|+|A34| < -(A44+R ) ), \text{ then the sub pyramid is outside.}
\]

Here, \( R \) is the maximum perturbation function on the current interval; \( Aij \) are the coefficients of the quadratic function. The following test is performed for the perturbation function:

\[
\text{if } ( |A11|+|A22|+|A33|+|A12|+|A13|+|A23|+|A14|+|A24|+|A34| < |A44| ), \text{ then the sub pyramid is outside of the range of definition of perturbation, where } Aij \text{ are the coefficients of the quadratic perturbation function, and a value of } R \text{ is additionally calculated and added to the basic function. If the intersection is determined, then the sub pyramid is subjected to the next recursion level. Sub pyramids that do not intersect with the object are not subjected to further immersion into recursion. This corresponds to elimination of square screen spaces from consideration, to which the sub pyramid (and, consequently, the object surface) is not mapped. The viewing pyramid is subdivided until reaching the maximal set level of recursion. The technique has an advantage as it allows discarding of large parts of empty space at an early stage. While searching for voxels containing the imaging object surface areas, the pyramidal space is traversed along the quaternary tree which leaves are roots of binary trees. Masking is used during traverse of the tree in case of opaque objects. The multilevel ray casting technique allows us to determine effectively and quickly belonging of rays of different levels (pyramids) to surfaces, and discard space regions outside the objects.}

Application of projective transformation extrapolates the rendering algorithm to pyramidal volumes and thereby allows us to generate images with perspective. In 3D space, the point with the Cartesian coordinates \((x, y, z)\) is associated with an infinite set of homogeneous coordinates \((x', y', z', a)\) such that \(x=x'/a, y=y'/a, z=z'/a\) i.e., the homogeneous coordinates are determined within a common nonzero factor. Special interest presents the transformation matrix affecting the homogeneous coordinates in the following manner:
\[
\begin{pmatrix}
C_{11} & C_{12} & C_{13} & C_{14} \\
C_{21} & C_{22} & C_{23} & C_{24} \\
C_{31} & C_{32} & C_{33} & C_{34} \\
C_{41} & C_{42} & C_{43} & C_{44}
\end{pmatrix}
\begin{pmatrix}
X_m \\
Y_m \\
Z_m \\
1
\end{pmatrix}
=
\begin{pmatrix}
x_p \\
y_p \\
z_p \\
a_p
\end{pmatrix}
\text{or } (C)(M) = (P),
\] (7)

where \((C)\) is the transformation matrix; \((M)\) are the homogeneous coordinates of the point of space \(M\); \((P)\) are the coordinates at \(P\) corresponding by the mapping. Within the scope of projective geometry, a theorem is proved that the projective mapping of space \(M\) to space \(P\) is unambiguously defined by specifying five pairs of points corresponding by the mapping provided that from five points specified in space \(M\), none of four points are in the same plane. Let us choose five pairs of such reference points \((M_i)\) and \((P_i)\) (the upper index corresponds to the number of pair) and compose the set of equations:

\[
(C)(M_i^*) = \rho^i(P_i^*),
\] (8)

where \(i = [1,...,5]\), \(\rho^1, \rho^2, \rho^3\) and \(\rho^4\) are the unknown factors; \(\rho^5 = 1\). Solving these equations, find the coefficients of projective transformation matrix \((C)\) used further to transform the geometric primitives.

To determine the elements of \(C\), we set up a system of matrix equations

\[
CM = P \quad (9)
\]

\[
CM_{ext} = P_{ext} \quad (10)
\]

Next, we solve the system, where \(c_{ij}\) are the elements of matrix \(C\); \(m_{ij}\) is the \(j\)-th homogeneous coordinates of the \(i\) is the \(th\) reference point; \(q_i\) are unknown multipliers; \(p_{ij}\) is the \(j\)-th homogeneous coordinate of the \(i\)-th transformed point.

Thus, we have twenty unknowns and as many equations.

Let us find:

\[
C = P M_{inv}, \text{ where } M_{inv} \text{ is the inverse of } M \quad (11)
\]

\[
PA = P_{ext}, \text{ where } A = M_{inv} M_{ext} = (a_i), \quad (12)
\]

The new matrix \(B = (bij = ai * pij)\):

\[
QB = P_{ext}, \text{ which is found from the last relation, we get the final expression of projective transformation matrix } C
\]

\[
\begin{pmatrix}
C_{11} & C_{12} & C_{13} & C_{14} \\
C_{21} & C_{22} & C_{23} & C_{24} \\
C_{31} & C_{32} & C_{33} & C_{34} \\
C_{41} & C_{42} & C_{43} & C_{44}
\end{pmatrix}
= \hat{N}
\] (15)

\[
\begin{pmatrix}
m_{11} & m_{21} & m_{31} & m_{41} \\
m_{12} & m_{22} & m_{32} & m_{42} \\
m_{13} & m_{23} & m_{33} & m_{43} \\
m_{14} & m_{24} & m_{34} & m_{44}
\end{pmatrix}
= M
\] (16)

\[
\begin{pmatrix}
q_1p_{11} & q_1p_{21} & q_1p_{31} & q_1p_{41} \\
q_1p_{12} & q_1p_{22} & q_1p_{32} & q_1p_{42} \\
q_1p_{13} & q_1p_{23} & q_1p_{33} & q_1p_{43} \\
q_1p_{14} & q_1p_{24} & q_1p_{34} & q_1p_{44}
\end{pmatrix}
= P
\] (17)
\[
\begin{pmatrix}
m_{11} \\
m_{22} \\
m_{33} \\
m_{44}
\end{pmatrix} = M_{ext} 
\]  
(18)

\[
\begin{pmatrix}
p_{11} \\
p_{22} \\
p_{33} \\
p_{44}
\end{pmatrix} = P_{ext} 
\]  
(19)

6. Conclusions

Our investigation in the volume-oriented visualization technology has made it possible to reveal some advantages in both the scene representation technique and the rendering algorithm oriented to real-time implementation. The changing over from rasterization in the image plane (“back-end” or the image-space end of the graphics pipeline) to volume rendering, (“fronted” or the object-space end of the graphics pipeline) in combination with the proposed object definition techniques, though increases the amount of real-time computation, as a whole, but nevertheless it results in some merits improving the scene imaging realism. The main merits of our approach are the following: efficiency of the multilevel masking rendering technique combining simple computation with fast search and discard of spaces out of the scene objects; the reduced number of surfaces for describing curvilinear objects (representation of objects by freeform surfaces reduces 100 times and more the database description compared with their representation by polygons); the reduction of the load on the geometry processor and a decrease of data flow from it to the video processor; simple animation and morphing of scenes.

References

[1] Tuy H., and Tuy L. 1984 “Direct 2-D Display of 3-D Objects,” IEEE Computer Graphics and Applications 4 (10) 29–33
[2] Perlin K., and Hoffert E. M. 1989 “Hypertexture” Computer Graphics 23 (3) 253-262
[3] Karla D., and Barr A. H. 1989 “Guaranteed Ray Intersections with Implicit Surfaces,” Computer Graphics 23 (3) 297-306
[4] Hart J. C. 1994 “Sphere Tracing: A Geometric Method for the Antialiased Ray Tracing of Implicit Surfaces,” The Visual Computer 12 (10) 527-545
[5] Mitchel D. 1990 “Robust Ray Intersection with Interval Arithmetic,” in Proc. on Graphics Interface (Toronto: Canadian Information Processing Society) pp. 68-74
[6] Sherstuyk A. 1999 “Fast Ray Tracing of Implicit Surfaces,” Computer Graphics Forum 18 (2) 139-147
[7] Reimers M., and Seland J. 2008 “Ray Casting Algebraic Surfaces using the Frustum Form,” Computer Graphics Forum 27 (2) 361-370
[8] Vyatkin S. I. 2007 “Complex Surface Modeling Using Perturbation Functions,” Avtometriya 43 (3) pp. 40–47
[9] Romanyuk O. N., Lyashenko Y. L., Melnik A. V., Goncharuk 2010 «Effective Models for the Specular Color » Constituent Computing Journal of computer science and engineering 2(2) 25-29