UDK 004.94

S. Vyatkin, PhD.,
S. Romaniuk,
O. Romaniuk, PhD.

VISUALIZATION OF 3D AMORPHOUS OBJECTS USING FREE FORMS

Abstract: *A technique for specifying 3D amorphous objects by free forms is considered. A fast semi-transparent object rendering algorithm is proposed. A method for simulating inhomogenety is presented.*

Keywords: *free forms, amorphous objects, voxels, multilevel ray casting*

С. И. Вяткин, канд. техн. наук,
С. А. Романюк,
О. В. Романюк, канд. техн. наук

ВИЗУАЛИЗАЦИЯ 3D-АМОРФНЫХ ОБЪЕКТОВ С ПОМОЩЬЮ СВОБОДНЫХ ФОРМ

Аннотация: *Представлена методика для формирования 3D-аморфных объектов с помощью свободных форм. быстрый алгоритм рендеринга полупрозрачных объектов. Предложен метод моделирования неоднородностей.*

Ключевые слова: *свободные формы, аморфные объекты, воксели, многоуровневый рейкастинг*

С. І. Вяткін, канд. техн. наук,
С. О. Романюк,
О. В. Романюк, канд. техн. наук

ВІЗУАЛІЗАЦІЯ 3D-АМОРФНИХ ОБ'ЄКТІВ ЗА ДОПОМОГОЮ ВІЛЬНИХ ФОРМ

Анотація: *Наведено методику для формування 3D-аморфних об'єктів за допомогю вільних форм. Пропонується швидкий алгоритм рендеринга напівпрозорих об'єктів. Запропоновано метод моделювання неоднорідності*

Ключові слова: *вільні форми, аморфні об'єкти, вокселі, багаторівневий рейкастинг*

Introduction. In work [1] was described method to generation a 3D clouds and fog of company Evans & Sutherland. In this project 3D clouds presented in the manner of layers of ellipsoids, brought evaluation of necessary amount of ellipsoids for more or less realistic scene. Described model, on which is calculated color of object observed through clouds. Since clouds divided into layers, in each layer be its color ci and fading constant pi. Brought formula, on which is calculated color of object observed through clouds.

Apparent color = (Actual Color)*exp [-0.5 (a0 + a1) d] + 0.5 (c0 + c1)*{1 – exp[-0.5(a0 + a1)d] },
where *Actual Color* is a real color of object.

d is the distance between watcher and object.

As possible understand from the brought formula, color and constant of fading is linear interpolated between two layers. If object close several layers, author resorts to such notion as a virtual color of object. For the apparent difficulty of this notion cost (stand)s enough asked thing – first Apparent color is calculated for two layers, afterwards result is substituted as Actual Color for two other layers and so on. The author notes that fading cons-tant must increase on the measure of increasing a height layer. This is done therefore that reduce an amou-nt of calculations, when is calculated color of object on the land, observed through N layers. Then if all layers sufficiently transparent, happens to conduct calculations of color of object for each layer. If upper layers suffi-ciently thick, the watcher does not see anything except the color given layer of clouds. There are different ways of modeling of mist (fog).

Brought two ways of calculation of color in the pixel with provision for the mist:

1. Color in the pixel with provision for the mist is calculated in vertex of triangles, then approximates in the pixel as of values in vertex.

2. Color in the pixel with provision for the mist undertakes from tables and does not depend on vertexes. Such way of calculation is identified tabular or pixel's by the way a calculation of mist.

In the same way distinguish two ways of calculation of fading. If be more exact, the difference is concluded in that consider a point of entry in the mist and on what distance to calculate a fading.

1. Distance is counted out from what or planes. Mist, assigning by the table, possible use with this way of calculation of fading only. On the one hand this rather quick way, but its using associate with arising number of artifacts, for instance, objects can appear from the mist when moving a watcher.

2. Distance is calculated as a length of segment from the eye of watcher before the object. This way more firm with respect to artifacts, but more expensive on computing expenses.

Formulas, on which is calculated function of fading such as follows:

Linear fading:

F = (fogend - d)/ (fogend – fogstart),

where fogstart and fogend are the points of entry and output in the fog;
where d is the distance from the watcher before the object.

Exponential fading:

F = 1 / (d * fogdensity),

where fogdensity is the density of fog.

Quadratic exponential fading:

F = 1 / (d * fogdensity)2.

As well was described a strategy Elevation Maps, [2], which main idea is concluded in use alpha-channel for keeping of some values (conditionally heights), which are used for the choice necessary texture layer. Within the framework of this strategies offered new way of displaying the three-dimensional atmospheric effects (3D clouds, smoke). To value of given method should refer a using a small quantity of primitives, such as triangles and in too time of the rich texture. However in this method is a restriction on the orientation of object. That is to say impossible observe a silhouette of object under the certain angle.

In the flight simulator system MaxView from Canadian company CAE Electronics clouds marketed in the manner of kits of such primitives, as raster light points. To defects of given approach should refer a greater amount of primitives for modeling more or less realistic cloud. So in the system MaxView, in first, it is impossible simultaneously render greater groups of clouds, and in secondly, in such clouds are absent heterogeneity. Model of calculation of color of object observed through such clouds does not correspond physical.

Known in the same way one more way of modeling of atmospheric effects, under which form clouds is built from base primitives, color and transparency are taken from texture map. Also known way to visualizations of atmospheric spottiness is POV-Ray. Leading total, should note that in all mentioned articles possible to observe that three-dimensional atmospheric effects are prototyped basically to the account of rich textures with the small amount of primitives. Absence of forms beside such objects brings about undesirable artifacts under their visualization. So in given work is offered to use alongside with the rich texture and complex form of objects.

An approach to particle system processing on a GPU is discussed in [3]. Balancing of CPU and GPU loads is described in detail. Original approaches aimed at reducing the data flow from the system memory to the video memory are proposed.

Method displaying the translucent objects formed by free form surfaces

In the visualization system Voxel Volumes designed method to visualization of uniform translucent objects limited by free form surfaces [4].

The freeform is a composition of the base surface and the perturbation functions

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

where the perturbation function R(x, y, z) is found as follows

$$R_i(x,y,z) = \begin{cases} Q_i^3(x,y,z), if\ Q_i(x,y,z) \ge 0 \\ 0, if\ Q_i(x,y,z) < 0 \end{cases}.$$

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

Since max[Q + R] ≤max[Q] + max[R], for estimating the maximum Q on some interval we have to calculate the maximum perturbation function on the same interval.

We propose an optimized version of algorithm, in which leaped through uniform area. In given work marketed mechanism of displaying the objects with pseudo-heterogeneous sharing density, in which is used such type of functions a fading, for which is easy calculated integral value of transparency for lengths along Z coordinate.

We use a texture [5 – 6] for the calculation of function density. In this case color on surfaces of object undertakes from texture map, but function of transparen-cy is calculated on this color (or in general undertakes from the separate map), for instance, possible take density to the proportional color, or inversely proportional.

Model description

Difference of algorithm ray casting from ray tracing is that in first from them in purposes of increasing a velocity of calculations are not traced secondary rays. At modeling of passing of light through translucent ambiences with this will take aim neglect a refraction and fading the secondary rays. In given models stays only reflection and fading a light on way of following from the object to the eye of watcher.

Algorithm of color accumulation

Formula, on which is calculated color of pixel possible to express as follows [7]:

$$P_\lambda = \sum_{n=0}^{N} I_{\lambda n} \Omega_n \prod_{m=0}^{n-1} (1 - \Omega_m).$$

The final color of pixel, but $\tilde{\lambda}$ may be r, where P_λ g or b (i.e. red, green or blue, accordingly). $I_{\lambda n}$ is the intensity of n-voxel, computable by Phong illumination model, Ω_n opacity of n-voxel.

$I_{\lambda 0}$ is the reflected light from first point on ray of scan, $I_{\lambda N}$ back ground color and $\Omega_N = 1$. Achievement of density threshold is possible to trace as follows. If on k-step general transparency $(1-\Omega_0)(1-\Omega_1)...(1-\Omega_{k-1})$ becomes less certain ζ. This signifies that contribution all following for k-voxel will be small. And, so, scan possible to stop.

Functioning of algorithm possible to express as follows:

```
{
if (Transparency > $\zeta$) and (The last level of recursion) and
(Inquire for intersection= 1), then
{
    $P_{\lambda i} += I_{\lambda i}\, \Omega_i$ Transparency
    Transparency $= 1 - \Omega_i$
}
```

For the first step $P_\lambda = 0$, for $\lambda = r, g, b$; Transparency = 1.

If it is used perspective transformation, should contribute an adjustment in the algorithm of accumulation of color in the pixel. This is because size of voxel, as a result geometric primitive transformations, becomes dependent from the coordinate. That is to say with increase Z size of voxel increases and so at the recalculation of color in the pixel opacity of voxel should also recalculate with the correction for changing its length.

In general event, if change sum an integral, intensity [9] will is:

$$I = \int_0^D color(x(s)) \exp\{-\int_0^s opacity(x(t))dt\} ds$$

where *D* is a distance on *Z* coordinate, on which occurs a calculation of intensity; *x*(*s*) is a point on the length along the ray of observation.

Color, opacity is the color and value of opacity in given point.

This formula more suits for our event in contrast with the formula (1), since in she is taken extent of length account into z coordinate. However, then happen to calculate an integral on each length or draw near its or expression. So we happen to use an admission, for instance that on length of segment a value of transparency constantly. In this case for N elements possible to change:

$$P = 1-(1-\Omega)^N$$

Ω – it is opacity in point.
P is the total opacity for N elements.

If consider N at all amount of elementary lengths on ray, but as its length, definitively:

$$P = 1-(1-\Omega)^{\Delta Z}$$

where ΔZ is a length of segment.

As was it already noted in consequence of garbling the geometric primitives a length of voxel to become to be hung from Z coordinates. This dependency possible to express as follows:

$$\Delta Z = M_{zz}\left(\frac{Z_{\max}}{M_{tz}Z_{\max}+M_{tt}} - \frac{Z_{\min}}{M_{tz}Z_{\min}+M_{tt}}\right)$$

That is to say conduct transformation for coordinates beginning and end of segment.

M_{zz}, M_{tz}, M_{tt} are the elements of perspective transformation matrix, Z_{max}, Z_{min} is a beginning and end of segment in the coordinate system of object.

Definitively this part of algorithm possible to express as follows:

```
if (Transparency $>\zeta$) and (Inquire for intersection = 1), then
{
    Accomulated $Color$ += $Color\ P$ Transparency
    Transparency = 1 − $P$
}
```

Accomulated Color is the final color of pixel.

Color is the color computable in point, for instance, on Phong illumination model.

Transparency is the value marking general transparency from the beginning of scan of ray before the given point.

P is the opacity in the point.

In the same way should point to the fact, in check is not met condition (Last recursion level), this as once corresponds that segments on the ray of scan can be a different length.

Texture

In the visualization system Voxel-Volumes designed method of imposition of texture on freeform surfaces [3]. In hierarchies of classes a class of texture VxTexture is inherited from VxObject base class. So exists a possibility to conduct manipulates with the texture to similar manipulates with rest objects, for instance, tumbling, shift, are realized by the cognate image as this is done with free form surfaces.

In consequence of which, exists a possibility to use such objects as the most textures, i.e. object itself, used as textures is not displayed, but limits texture map, inflicted on other object. It is necessary to note an advantage that texture is stood in the separate class, rather then is a floor of already existing classes of objects. This does a process of displaying a texture independent from objects to which it is superimposed. For instance, us have only such classes of displayed objects as quadrics and quadrics with analytical perturbations, and are required to add scalar perturbations in the system. In the first event we happen to except realization of methods of class of scalar perturbations, realize a method of displaying a texture in this class, but on the strength of that that texture does not depend on the object to which it is superimposed, second to do already there is no need to. For the texture also marketed several ways of parameterization. Is it by default considered that texture not parameterized – is located inwardly cube by layers to perpendicular axis Z, in this case for the calculation of texture coordinates, coordinates X,Y are simply scaled on the size of texture, if texture is turned, multiplied by corresponding rotation matrix [10]. In the event of parameterization, for instance, spherical, for the calculation of texture coordinates is conducted separate transformation, corresponding to projective on the sphere. In the same way designed a method to filtration a texture. We use MIP-map and conduct bilinear interpolation of texture. In general, event voxel gets between four texels from the given level of detail, and color in given voxel is from amounts these four values, taken with certain weights. Within the framework of work on the project was marketed trilinear interpolation of texture.

Animation

3D metamorphosis operation transforms the first defined object into second with obtaining multiple intermediate forms. The term originates from the word metamorphosis and refers to the animation technique in which one pattern is gradually transformed to another. During the metamorphosis (morphing), the initial pattern is gradually transformed to the final one.

A sequence of frames of transformation of one object to another is generated by means of the initial, final, and key intermediate models.

Let F1 and F2 be values of the perturbation functions of the first and second objects, respectively. Then the resulting perturbation function is calculated as follow:

$$F=\beta F1 + (1-\beta)F2,$$

where β is the positive continuous function.

For function-based objects with the use of perturbation functions, one can perform 3D morphing of nongomeomorphic objects.

Another way is global deformation. First thing that is necessary to state is that if we want to propagate the deformation it should be somehow added to all object. Actually the current scene-tree is organized so that it is no possibility to add object only by referencing i.e. without copying. This is done for avoiding situations when being changed somewhere the object unintentionally change the other part of the scene that referenced to it too. Thus the additional perturbations should have such parameters to assure the part-per-part connectivity for each pair of the object the perturbation affects. In this case it will be looked as one perturbation.

Conclusions

We have developed an algorithm for visualizing semi-transparent objects bounded by free forms. A method for fast rendering the internal inhomogeneous object via the attenuation function was proposed. We have realized a method for inhomogenety simulation. The advantage of the method is the possibility of processing inhomogeneous semi-transparent structures bounded by free forms.

References

1. Schachter B., (1980), “Computer Image Generation”, *A Wiley-Inderscience Publication John Wiley & Sons,* New York.

2. Vyatkin S.I., and Romaniuk O.V., (2010), Formation of Microrelief on the Base of Multilevel Elevation Maps, *Proceedings of 10th International Conference on Development and Application Systems,* Suceava, Romania, pp. 386 – 389.

3. Gladkiy D.A., Belago I.V., Elykov N.A., and Kuzikowski S.A., (2009), GPU Processing of Particle System Animation, *Avtometriya*, Novosibirsk, Russian Federation, Vol. 45, No. 6, pp. 22 – 28 (In Russian).

4. Vyatkin S.I., Dolgovesov B.S., Yesin A.V., [etc.], (1999), “Voxel Volumes Volume-Oriented Visualization System” *PROC. Shape Modeling and Applications*, *March 1-4, 1999,* Aizu-Wakamatsu, Japan, pp. 234 – 241.

5. Romaniuk O.V., (2010), High-productive Method of Perspective-correct Texturing, *Visnyk of Vinnytsia Politechnical Institute,* Vinnitsa, Ukraine, No. 2 (89), pp. 74 – 77 (In Russian).

6. Romaniuk S.O., Romaniuk O.N., and Piddubetska M.P., (2013), Basic Analisys of Texture Filtering Methods, *IX Miedzynarodowej Naukowi-praktycznej Konferencji “Actualne Problem Nowoczesnych Nauk”,* Przemysl, Vol. 30, pp. 29 – 34.

7. Knittel G., (1993), VERVE: “Voxel Engine for Real-time Visualization and Examination”, *Eurographics’93,* Vol. 12, No. 3, pp. 36 – 48.

8. Romaniuk O.V., (2010), Productive Methods of Quadratic Approximation of Perspective-correct Texturing, *Visnyk of Volodymyr Dahl East Ukrainian National University,* Luhansk, Ukraine, No. 10 (152), pp. 194 – 198 (In Ukrainian).

9. Nelson Max, Roger Crawfits, and Barry Becker, (1995), “Application of Texture Mapping to Volume and Flow Visualization”, *Proc. GRAPHICON’95,* Moscow, Russian Federation, 108 p. (In Russian).

10. Romaniuk S.O., Romaniuk O.V., Blagodyr D.L., (2013), Procedure of Determining Scalar Product of Two Vectors Given In Polar Coordinates For The Tasks of Computer Graphics, *MCTTP*, Khmel'nitsk, Ukraine, No. 2, pp.128 – 131 (In Ukrainian).

Received 03.06.2015

Список використаної літератури

1. Schachter B., (1980),“Computer Image Gene ration”, *A Wiley-Inderscience Publication John Wiley & Sons,* New York.

2. Vyatkin S.I., and Romaniuk O.V., (2010), Formation of Microrelief on the Base of Multilevel Elevation Maps, *Proceedings of 10th International Conference on Development and Application Systems,* Suceava, Romania, pp. 386 – 389.

3. Гладкий Д. А. Анимация систем частиц на графическом ускорителе / Д. А. Гладкий, И. В. Белаго, Н. А. Елыков, С. А. Кузиковский //Автометрия. – Новосибирск : – 2009. – Том 45. – № 6. – С. 22 – 28.

4. Vyatkin S.I., Dolgovesov B.S., Yesin A.V., [etc.], (1999), “Voxel Volumes Volume-Oriented Visualization System” *PROC. Shape Modeling and Applications*, *March 1-4, 1999,* Aizu-Wakamatsu, Japan, pp. 234 – 241.

5. Романюк О. В. Високопродуктивний метод перспективно-коректного текстурування / О. В. Романюк // Вісник Вінницького політехнічного інституту. – 2010. – № 2 (89). – С. 74–77.

6. Romaniuk S.O., Romaniuk O.N., and Piddubetska M.P., (2013), Basic Analisys of Texture Filtering Methods, *IX Miedzynarodowej Naukowi-praktycznej Konferencji “Actualne Problem Nowoczesnych Nauk”,* Przemysl, Vol. 30, pp. 29 – 34.

7. Knittel G., (1993), VERVE: “Voxel Engine fo real-time Visualization and Examination”, *Eurographics’93,* Vol. 12, No. 3, pp. 36 – 48.

8. Романюк О. В. Продуктивні методи квадратичної апроксимації перспективно-коректного текстурування / О. В. Романюк // Вісник Східноукраїнського національного університету імені Володимира Даля. – 2010. – № 10 (152). – С. 194–198.

9. Nelson Max, Roger Crawfits, and Barry Becker, (1995), “Application of Texture Mapping to Volume and Flow Visualization”, *Proc. GRAPHICON’95,* Moscow, Russian Federation, 108 p.

10. Романюк О. В. Визначення скалярного добутку двох векторів, заданих у полярних координатах, для задач комп’ютерної графіки / С. О. Романюк, О. В. Романюк, Д. Л. Благодир // Вимірювальна та обчислювальна техніка в технологічних процесах. – Хмельницький. – 2013. – № 2. – С. 128 – 131.

Vyatkin
Sergei Ivanovich, Ph.D,
Senior Scientific Researcher of Synthesizing Visualization Systems Laboratory, Institute of Automation and Electrometric SB RAS, Koptyuga avenue, 1, Novosibirsk, Russian Federation,
Tel. (383)3333630.
E-mail: sivser@mail.ru

Romaniuk
Sergii Oleksandrovych, postgraduate of the Department of General Physics and Photonics, Vinnitsa National Technical University, Khmelnitsky shoes street, 95, Vinnitsa, Ukraine, 21021,
Tel.: (0432) 598-273.
E-mail: office@hitec.com.ua

Romaniuk
Oksana Volodymyrivna, Ph.D, Senior Lecturer of the Department of Software, Vinnytsia National Technical University, Khmelnitsky shoes street, 95, Vinnitsa, Ukraine, 21021,
Tel: (0432) 598-273.
E-mail: oksana_gelika@mail.ru