UDC 004.925.83

S.I. VYATKIN
Institute of Automation and Electrometry SB RAS, Novosibirsk, Russia
S.A. ROMANYUK, O.O. DUDNYK
Vinnitsa National Technical University, Ukraine

GEOMETRIC MODELING WITH SCALAR PERTURBATION FUNCTIONS

For deciding the problems, in accordance with the automation of process of preparing the databases from CAD models, with provision for specifics different (CAM/CAD) systems is offered way of preparing the databases for modeling of surfaces, assigned by scalar perturbation functions and software system concept to automation of preparing the databases. Presented method of creation height maps from databases of programs 3D- modeling and 3D- designing (CAM/CAD), such as 3d-studio and bCAD.

Additional Keywords: scalar perturbation functions, height maps, OpenGL, CAM/CAE/CAD, bCAD.

С.И. ВЯТКИН
Институт автоматики и электрометрии СО РАН, Новосибирск, Россия
С.А. РОМАНЮК, А.А. ДУДНИК
Винницкий национальный технический университет, Украина

ГЕОМЕТРИЧЕСКОЕ МОДЕЛИРОВАНИЕ НА ОСНОВЕ СКАЛЯРНЫХ ФУНКЦИЙ ВОЗМУЩЕНИЯ

Для решения проблемы, в соответствии с автоматизацией процесса подготовки базы данных на основе моделей CAD с обеспечением специфики различных (CAM/CAD) систем предлагается способ получения баз данных для моделирования поверхностей, на основе скалярных функций возмущения и концепции системы программного обеспечения для автоматизации подготовки базы данных. Представлен метод создания карты из баз данных программного обеспечения для 3D-моделирования и 3D-проектирования (CAM / CAD), таких как 3d-studio и bCAD.

Ключевые слова: функции скалярного возмущения, карты высот, OpenGL, CAM/CAE/CAD, bCAD.

Introduction

As a high-volume applied programming interface 3D graphics in the project VxVl can emerge a library Microsoft XSG (eXtensible Scene Graph), which is developed by companies Microsoft, SGI, HP within the framework of Fahrenheit project. XSG allows describing 3D scene in terms so called scene graph. Library XSG supports a mechanism of expansion (possibility of accompaniment of new types of nodes in the scene graph). Nodes of scene graph can be in particular surfaces of second order (quadrics), scalar field (regular net of heights on the base surface), three-dimensional texture - that is to say as once that objects which specific for VxVl project.

In XSG at first built-in kit of expansion, intended for the representation and visualization very complex (on the amount of potentially necessary polygons for the description of objects) scenes. This kit of expansion is marked by the abbreviation LMX (Large Model Extensions). Declared that LMX intended for the exhibit development, which will ensure an interaction with models by the volume in several (multi) gigabyte on the trained frequency not less than 10 Hz.

LMX designed for the support of modeling of geometry in the parametric representation, performing an accommodation/positioning and animation of solid objects. Image parametric represented geometry is realized by the way of adaptive a subdivision a surface on polygons and their following image. Are they herewith used simplification of surfaces, choice for a subdivision only potentially visible (not closed other) surfaces, semantic elimination for displaying (elimination by assigning user).

For designing scenes in the project VxVl we used both interactive and packet applications. Example of interactive application can be an ambience with the graphic user interface (3D Studio MAX), which allows by means of the choice of variants of actions through the menu and minimum keyboard input to create three-dimensional objects in polygonal representation. Hereinafter these objects with the help of special algorithms are translated from polygonal representation in one of the representation of project VxVl. Hereinafter, objects in the new representation are saved on the disk in the manner of the module of the database in the format, defined XSG/LMX. For visualization of saved objects must be written (programmed) extension XSG, capable to read saved objects, create for them necessary nodes of scene graph and realize their visualization in LMX paradigm.

Example of packet application can be an environment with the programming possibility VxVl objects/scenes on what or programming language, for instance, on languages Hyperfun, April, C++. Programming environment must ensure a necessary kit a libraries, on the one hand maintaining semantics VxVl (functions for designing surfaces-quadrics, representation of perturbation functions and so on), but other hand maintaining classical notions 3D graphics - polygonal representation of surfaces with the sources of illuminating, textures, material, parameters of observing and so on. Programmed thereby VxVl a scene is saved on the disk in the manner of the module of the database in the format defined XSG/LMX. For visualization of saved objects in the same way either as in the event of interactive approach must be written (programmed) extension XSG, capable to read saved objects, create for them necessary elements a scene graph and realize their visualization in LMX paradigm. In both approaches - interactive and packet, this can be one and ditto extension XSG.

Given article introduces a reader with packet application method for preparing the databases, on the base of models CAD/CAE/CAM systems. Which ensure a high level of interaction with the possibility of use of already ready models. As well as relieving a process of building of new models for the system Voxel-Volumes with using a wide spectrum of instruments of modern prototyping program complexes of class 3D-Studio, bCAD, ensuring user high level of interaction at the building models.

Analysis of research and publications

Within the framework of the project Voxel-Volumes [1-11], the next objects/surfaces visualization is possible. The order of all main types:

- composition of quadrics,
- real perturbation functions,
- scalar fields (regular net of heights on the base surface),
- three-dimensional texture,
- classical polygonal representation.

One of the variants of low-level applied programming interface 3D graphics in the project Voxel-Volumes can emerge OpenGL library. This library is a standard de facto for programming 3D graphics on many platforms - and is to advantage used both in professional and in gaming applications. Function OpenGL supports sufficiently developed kit of notions 3D graphics:

- surfaces constructed from polygons ,
- surface representation by polynoms of degree N,
- sources of light,
- texture (including and three-dimensional),
- mist,
- etc.

Besides, OpenGL affords (for developers of equipment and drivers) mechanism of expansion that allows to add in it that functionality, which specific for algorithms of visualization of project Voxel-Volumes (VxVl).

Problems of preparing the databases in system Voxel-Volumes

From geometric model of the objects of the system Voxel-Volumes [1-11] in which is used by object representation in the manner of implicit perturbation functions and on the base scalar perturbation functions, obviously that process of preparing the databases for making the real object models, without facilities of automation of modeling, will very laborious. It is complicated that in the process of modeling, person has a deal not with the object, which he can see, but with its functional representation. As a rule, for surfaces, presented by quadrics and analytical perturbation functions process of modeling goes on the scheme:

1. changing a function,
2. generation of the scene,
3. the result is a satisfactory?

- if yes, the modeling stops.
- if no, are brought back into 1.

Given process is complicated that on the usual personal computer a process to generation an image occupies rather long time insufficient for ensuring interaction process of modeling.

Preparing the databases for surfaces, using representation of objects on the base scalar perturbation functions, in addition, complicated that scalar functions will be assigned by the two-dimensional table of numbers, in the textual file for instance. Accordingly, edit a file, which consists from several groups of ten of lines of type even:

< 10 10 10 20 30 30 30 20 20 30 40 30 10 10 0 0 0 >

< 10 22 32 42 52 60 58 50 52 44 50 38 12 0 10 10 0 >

much difficult. But at the amount them in several hundreds, this becomes practically impossible, but after all such data present very small table.

Problem partly leaves that as databases for terrain possible use DEM (digital elevation model) models. This standard is designed by U.S. Geological Survey and, on essences, is a table of heights terrain with counting out through 7.5 or 15 minutes. DEM model consists of two files, binary file of data in which recorded heights in the manner of 16- bit fixed numbers, and head file which describes a format of record of numbers used in the file of data (BigEndian or SmallEndian), but in the same way area on terrestrial surface which describe heights in the file of data. In more detail format DEM file is described in. But for making the surfaces, different from landscape, much difficult find a way to acceptable automation of process of preparing the databases.

We suggest a way of preparing the databases for modeling of surfaces, represented scalar perturbation functions designed method of creation height maps from databases of programs 3D- modeling and 3D- designing (CAM/CAD), such as 3d-studio and bCAD. These programs possess a sufficient degree of interactivity due to the fact that use wire models for visualization of objects in step of designing, as ensures a high velocity rendering expressing models. In the same way, such programs afford most broad spectrum of tools for making and editing the complex surfaces. Besides, these programs, afford a program interface (API), for writing connected modules, by means of which possible automate a process of modeling. In the same way by means of this interface possible to get a description of geometric characteristics of objects, for instance, such as coordinates vertex triangles of triangulated surfaces. As a rule, for writing by these modules are used languages of the high level. In AutoCAD system for writing the modules, is used language AutoLisp, but in bCAD for this is used language Java.

Geometric model of surfaces of the CAD/CAE/CAM systems

Objects in designing systems, are presented by triangulated surfaces, or allow getting such representation. As a rule, for complex surfaces is used irregular - triangulated representation, which shall hereinafter name triangulation. In this case surface is presented in the manner of vertex list and reference list to vertexes. Reference list to vertexes is used for the determination of elements triangulation through the vertex list. Writing a surface in such case has following type:

$$V=\{x_1,y_1,z_1, \dots x_i, y_i ,z_{i,\dots} x_n ,y_n ,z_n \}, \quad F=\{0 ,1 ,2 \dots k,m,l,\}, \tag{1}$$

where V presents itself an array of the coordinates of the vertexes of the elements of triangulation, but F presents triangles list. Accordingly, amount vertexes this surface is n, but an amount of elements triangulation is LF/3, where LF is an amount of elements in F. Each an element of triangulation is defined, as a three-tuple vertexes Tv1,Tv2,Tv3 , but vertex is defined by the three-tuple of coordinates x, y, z. Accordingly first element of triangulation is defined through arrays V and F as follows:

$$\begin{gathered} T_{v0x} =V[F [0]*3] , T_{v0y}= V[F [0]*3+1] , T_{v0z} =V [F [0]*3+2], \\ T_{v1x} =V[F [1]*3] , T_{v1y}= V[F [1]*3+1] , T_{v1z} =V [F [1]*3+2], \\ T_{v2x} =V[F [2]*3] , T_{v2y}= V[F [2]*3+1] , T_{v2z} =V [F [2]*3+2], \end{gathered} \tag{2}$$

where *i*-element of triangulation (j= i*3) is:

$$\begin{gathered} T_{vix} = V [F [j]*3] , T_{vy\,i} = V [F [j]*3 +1], T_{vz\,i} = V [F [j]*3 +2], \\ T_{vy\,i+1} = V [F [j+1]*3] , T_{vy\,i+1} = V [F [j+1]*3+1], T_{z\,i+1} = V [F [j+1]*3+2], \\ T_{vx\,i+2} = V [F [j+2] *3] , T_{vy\,i+2} = V [F [j+2]*3+1], T_{vz\,i+2} = V [F [j+2]*3+2], \end{gathered} \tag{3}$$

where *F*[*j*] is a j–element in the array of triangles F, but V[F[j]*3] is a three-tuple in the array of coordinates V corresponding this element.

Representation of surfaces in such type helps to reduce an amount duplicating information, to the account that element of triangulation will be assigned through references to three-tuples of coordinates of vertexes, which are fixed numbers, while direct representation of one vertex (representation of its coordinates) requires three-tuples real numbers. Thereby, direct representation of a triangle (through coordinates of vertexes), which is used by in real-time visualization systems, requires nine real numbers, while representation through references requires the whole only three-tuples fixed numbers. And accordingly size of data, getting under the direct representation of a regular triangulation is $S_1= 9*(2*n^2)*8$ bytes. And size of database in the second case is $S_2= (n^2)*8 +3*(n-1)^2*4$ bytes. Where size a real number prescribed equal eight bytes, but size fixed number equal four bytes. Accordingly ratio of sizes of databases will $k= S_2 /S_1= 0.05 +0.08(n-1)^2/n^2 \approx 0.13$.

Coming from that, what is a, or him like, representation of surfaces much wide-spread and is supported by many designing systems, therefore was designed following software system model of preparing the databases for the system Voxel-Volumes. Software system structure has a following type:

- Kit connected by the modules for different designing systems.
- Program of preparing the databases.

At primary task, entrusted on connected module, is a transformation of geometric representation of surface of given modeling system. Moreover, with provision for its specifics, occurs a transformation in the above-mentioned format and record it's in the file. Tinned thereby files, containing triangulated surfaces of objects created in the modeling system, are processed by the program of preparing the databases. Such concept was offered coming from that different prototyping systems use different API and programming languages, used for writing connected modules. Under such approach of functions, realized by the module, are concluded only in ensuring compatibility of geometric representations of surfaces. Thereby, module, itself, is a small program, and its program realization does not require the greater temporary expenses. The main part is stood in the separate program, which takes at the input file of source data of standard format, and creates on its base a file of database for the system Voxel-Volumes.

Algorithm description of preparing the databases

Let we have an object *O*, and triangulation *T*, presenting surface of object *O* in the manner of (1). Coming from this, we needed to build a representation of object in the manner of two height maps, in the manner of two *oriented* planes P_U and P_D and corresponding it two two-dimensional tables of numbers t_u and t_d. Where t_u characterizes a deflection of source surface from the plane P_U toward of normal P_U. But td characterizes a deflection of source surface from the plane P_D toward of normal P_D. Accordingly surface of object *O* must meet the requirements, which are presented to surfaces prototyped with its help. Will hereinafter be described method of building model, using partitioning an object by *one* plane. Using a partition by only one plane was choose therefore that realization algorithm for the greater amount of splitting planes has no a particular sense. Since in the case of not satisfaction by the model to conditions of used method, interface of CAM systems allows to split a model on forming its group of surfaces, each of which will already satisfy these conditions, and build separately models of these groups by proposed method. While realization of suitable interface for ensuring such building in the case of using a partition by more then one plane seems meaningless. The plane $B_{p,}$ splitting object on two parts shall hereinafter name *a plane of partition* a triangulation *T*. Two triangles of triangulation *T*, will name *to be bound on the edge* or *on the vertex*, if they have, at least once, one general edge, or one general vertex accordingly. Triangles, which bound on two vertexes, are bound on edge. Triangle will name *to be oriented* toward the vector *v*, or simply *oriented*, if angle between normal of triangle *n* and vector *v* less or is $\pi/2$. In other words if scalar product $(\boldsymbol{n} \bullet \boldsymbol{v}) \geq 0$. In analogy, *oriented* will name a plane with the narrowly defined vector of normal. Accordingly, a plane P_U is oriented toward the vector of normal $B_{p.}$ Plane P_D is oriented in the opposite direction. Hereinafter, height map for the plane P_U shall name *upper* height map, but height map corresponding plane P_D shall name *lower* height map. Not losing generalities, possible consider that plane of partition B_p of triangulation *T*, will be plane XOY, oriented toward axis Z. Process of building consists of four stages:

1. Finding of square-wave parallelepiped, containing triangulation.
2. External orientation of triangulation.
3. Partition of triangulation.
4. Height map building.

Finding of square-wave parallelepiped, which is kept a source object, occurs by samples of maximum and minimum values of coordinates from the array vertexes of triangulation T. Coordinates of vertexes of a parallelepiped are defined as:

$$P.=\{x_{min}\, y_{min}\, z_{min}\,,\ x_{min}\, y_{max}\, z_{min}\,,\ x_{max}\, y_{min}\, z_{min}\,,\ x_{max}\, y_{max}\, z_{min}\,,\ x_{min}\, y_{min}\, z_{max}\,,\ x_{min}\, y_{max}\, z_{max}\,,\ x_{max}\, y_{min}\, z_{max}\,,\ x_{max}\, y_{max}\, z_{max}\},$$

where x_{min} ,y_{min} ,z_{min} is a minimum coordinates of vertexes, x_{max} ,y_{max} , z_{max} is a maximum coordinates of vertexes.

Hereinafter produced external orientation of triangles of source surface. It is concluded in normal calculation to the triangle so that this normal was external to surface. For this occurs a sample of the most high triangle, the triangle, one of vertexes which has the most greater z -coordinate in the array of the vertexes of triangulation *T.* If z - coordinate of vertex this triangle inheres above plane of partition B_p, it is orientated toward z, otherwise toward -z. After orientating this triangle, *connected* with him triangles, are orientated toward his normal, and are marked as processed. Whereupon, are orientated all adjacent triangles of these triangles and so on. Criterion of coherence (on edge or on vertex) is installed by the user. If after orientating in *T* have stayed untilled triangles, from they are chosen uppermost and process is repeated until will stay nor one not oriented triangle. Process of orientating a triangle *t,* given by three vertexes v_1, v_2, v_3, toward the vector *n,* is concluded in the calculation normal triangle n_{tr} on the following formula (4):

$$n= [(v1- v2) \otimes (v2- v3)](sign((v1- v2) \otimes (v2- v3)) \bullet n), \tag{4}$$

where $\otimes$ is a mark a vector product, $\bullet$ is a scalar *sign* mark a function of sign, but a *v1, v2, v3* mark a vertexes of triangle. After orientating a triangulation, on the criterion with - directivities normal of a triangle and splitting plane B_p, is produced division of source array of triangles of triangulation *T* on two arrays, "upper" T_U and "lower" T_D.

That is to say

$$\begin{aligned} &T_i \subset T_u \text{ , if } (N_{Ti} \bullet N_{Bp}) \geq 0 \,, \\ &t_n \subset D,\ if\ \ (N_{Ti} \bullet N_{Bp}) < 0 \,, \end{aligned} \tag{5}$$

where N_{Bp} is a normal of splitting plane, N_{Ti} is a normal of i-element of triangulation *T.* Hereon they are built upper and lower maps of heights from groups of triangles T_U and T_D oriented toward N_{Bp} vector and $-N_{Bp}$ accordingly.

For this is created height map of given permit, in the suggestion that plane of partition is a plane XOY, it presents itself a rectangle being projection of enclosing parallelepiped on XOY plane. Nodes of height map situated on distance $dl=max(x_{max} - x_{min},\ y_{max} - y_{min})/map\_dim$, where x_{max} , x_{min} , y_{max} , y_{min} accordingly maximum and minimum x and y coordinates of projection of enclosing parallelepiped, but *map_dim* dimensionality of height map specified by the user. Ray is extend from each node of net toward the vector N_p for the upper group and toward $-N_p$ for lower. Ray presents itself two vectors: a vector beginning and unit normalized direction vector, *ray={pos,dir}.* Thereby, ray this ensemble of points in $\Re^3$ defined by following vector equation:

$$\mathbf{V} = pos + dir*t;\ t\in R,\ t\geq 0 \tag{6}$$

Ray coming out of [i,j]-node of upper height map is calculated as follows:

$$ray_{ijU} =\{pos_{ij},\ n\},\ \ pos_{ij}=(x_{min}+dl*i\ ,\ y_{min} +dl*j, z_{min}),\ n=N_{Bp}.\ , \tag{7}$$

ray coming out of [i,j]-node of lower height map is calculated as follows:

$$ray_{ijD} =\{pos_{ij},\ n\},\ \ pos_{ij}=(x_{min}+dl*i\ ,\ y_{min} +dl*j, z_{min}),\ n=-N_{Bp}.. \tag{8}$$

For each ray extended from upper height map is produced test on intersection with each triangle from the upper group. If ray crosses a triangle, is calculated distance from the node height map, from which it was extended, before point its intersection with the triangle. Hereinafter mod this distance is compared to number basing in the given node of height map and in the node is written maximum from these two numbers. Since ray perpendicular plane height map, x and y coordinates of cross point are x and y coordinates of vector *pos* given ray, consequently, distance from cross point of ray with triangle before node of height map is mod difference z coordinates of cross point and height map node:

$$t_u[i][j]=max(\ |z_{inter} - z_{pos}\ |\ ), \tag{9}$$

where z_{inter} is a z-coordinate of cross point of ray with triangle.

Such image is produced building of lower height map. Cross point of ray with the triangle is defined on the following algorithm. First inheres a cross point of ray with the plane of triangle, if ray crosses a plane of triangle, searches a cross point of ray with the triangle. Angle is first calculated on first stage between normal to triangle and directing ray vector. If scalar product $n_{tr} \bullet dir_{ray} =0$, the ray is parallel to plane of triangle. In this case it is considered that triangle by the ray is not crossed. Hereinafter calculated location of cross point of ray with plane of triangle comparatively beginning of ray:

$b=(n_{tr}\bullet pos_{ray}) -d$, where *d-* is a distance from origin of coordinates before given triangle. Then calculated parameter t corresponding to t in (6):

$$t= b/a, \tag{10}$$

if t<0, cross point with the plane of triangle does not belong a ray (6).

Then calculated vector of cross point of ray with the plane of triangle I_p.

$$I_p = pos_{ray} +dir_{ray} *t \tag{11}$$

Then coordinates of vertexes of triangle, transformation of shift on the vector $-I_p$ are translated in the coordinate system, which origin bases in the point I_p. Hereinafter, all vertexes project on the plane XOY, and checking for intersection goes already with vertexes projections only. It is produced in two stages:

1. If all xs or all y coordinates of projection of vertexes one sign, the ray does not cross a triangle.

2. If 1 is not executed, from the point 0 toward axis x is extend the ray and is produced test on intersection by it with edges of triangle, if amount of crossing a ray with edges is 1, then ray crosses a triangle and in this case cross point vector the ray and a triangle is I_p, otherwise ray does not cross a triangle.

Software realization of preparing the databases from geometric models CAD

On the given moment program complex of preparing the databases presents itself main a module, implementing building of models for the system Voxel -Volumes from vdm (voxel digital model file) file, written on language C++ and connected module for bCAD written on Java language. Connected module for bCAD in accordance with the above-mentioned software system concept realizes a process of getting a geometric object description in bCAD and data conservation in vdm format (voxel digital model file) structure which described below. Given file is a standard file for the main module and is used for the building of models for the system Voxel-Volumes.

Format of command line for starting main module of program.
input_file_name output_file_name [options] [cdegree] [dimx] [dimy],
where input_file_name is the name of model file, output_file_name is the name of model file for system Voxel-Volumes, options is the options of building, degree is a degree of coherence - 1 on vertex, 2 on edge, dimx and dimy is the resolution of height map.

Options of building -si show information on model written in vdm file, -gi generate additional information about geometry model in vdm file.

Example: tvm.exe ex.vdmf exout -gi 1 128 128

Format description vdm file

Indication integer means fixed number of 4 bytes size, float means real number of 4 bytes size, and char means a symbol of 2 bytes size. The structure of the voxel digital model file (vdmf, vdm) shown oh figure 1. File presents itself a structure which consists from the file header and data kit describing triangulation of surfaces. Format of file header vdm file format means format of record of numbers in file, 0 is the SmallEndian 1 is the BigEndian, surface_cnt is an amount of triangulation basing in the file, m_name_len is the length of name models in symbols, name is the name of model, baze_z is the z –coordinate of splitting plane.

Format of writing of triangulation, model_name_len is the length of symbol name of triangulation in symbol, vertices_data_len is_an amount of vertexes of triangulation, faces_data_len is an amount of elements of triangulation, model_name is the name of surface, vertices- coordinates of vertexes of an element of triangulation, faces is the edges of triangles.

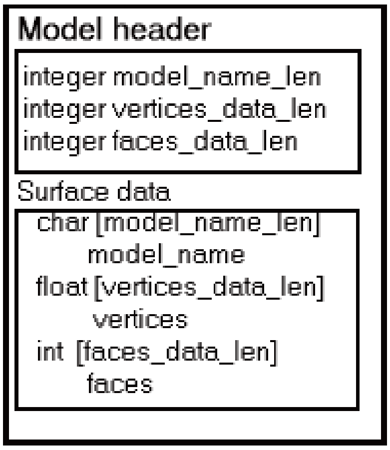

Fig. 1. Surface CAM model

Using restriction

Since model is built by the separate modeling method, which described above, the type of surface CAM model must satisfy all restrictions superimposed by this method on the class of prototyping surfaces. On the other hand, algorithm of building itself superimposes their own restrictions already not on the type of surfaces, but on their internal structure. First, CAM model must be presented by triangles. Secondly, for the correct building of database it is necessary correctly orientate triangles of surface, for this all triangles, within the framework of one surface model, must be *bound*. But for more correct building triangles must be bound on the *edge*. Needed to note that two last restrictions superimposed by the algorithm of building, permissive, they affect only on accuracy of building models.

So, models for the system Voxel-Volumes possible build from surfaces satisfying following requirements:

1. The type of surfaces must satisfy to criterions of the method of separate modeling.

2. Surface must be presented by triangles.

```
inter{     /* announcement that object is operation of crossing the volumes */
  qplane{ /* define the type of base surface as plane */
height("crown_up.tga", defval 3)/* define a type of perturbation a base surface as
scalar, and indicate its parameters: file, describing table of heights and threshold
of perturbation*/
        {
           use(int) /* define a type of interpolation as bicubic */
           clip 1 /* announcement of given surface as surfaces of the free form */
           factor 1.1 /* indicate degree of perturbation */
        }
          }
qplane /* clipping plane */
  {
        rotate(0,0,pi) / * operation of rotation on angle pi around axis z */
   }
}
```

Conclusion

For deciding the problems, in accordance with the automation of process of preparing the databases from CAD models, with provision for specifics different system CAD is offered algorithm of preparing the databases and software system concept to automation of preparing the databases. Realized program complex representing itself main module, realizing building models for the system Voxel-Volumes, and connected module for bCAD designing system. The main module was realized on language C++ and presents itself console application. Module for bCAD realized on Java language and presents itself application, started in bCAD. Object models were built from models 3D-Studio, which were imported in bCAD and were processed by above- mentioned programs of preparing of the databases from models CAD/CAE/CAM systems.

References

1. Sergey I. Vyatkin, Boris S. Dolgovesov, Valerie V. Ovechkin, Sergei E. Chizhik, Nail R. Kaipov “Photorealistic imaging of digital terrains, freeforms and thematic textures in realtime visualization system Voxel-Volumes”, GraphiCon ’97, Moscow.

2. Savchenko V.V., Pasko A.A., Vyatkin S.I. et al. New approach in geometric modeling: distributed and hardware implementation perspectives.// International Conference on Computers and Devices for Communication CODEC-98. Calcutta, India, 1998, pp. 285-290.

3. S.I. Vyatkin, B.S. Dolgovesov, S.E. Chizhik, “Synthesis of virtual environment with object-space recursive subdivision” - Graphicon’98 Proceedings, S. Klimenko et al. (Eds). 1998.

4. S.I. Vyatkin, B.S. Dolgovesov, A.V. Yesin et al. "The Real-Time System of Volume Visualization VOXEL-VOLUMES", 4 International Conference - Recognition of Objects and Analysis of Images, Novosibirsk-1998.

5. S.I. Vyatkin, B.S. Dolgovesov, A.V. Yesin et al. “The Visualization of Non-Homogeneous Structures in Real-Time with Multi-Level Recursive Ray Casting”, 4 International Conference - Recognition of Objects and Analysis of Images, Novosibirsk-1998.

6. S.I. Vyatkin, B.S. Dolgovesov, A.V. Yesin et al. Voxel Volumes volume-oriented visualization system, International Conference on Shape Modeling and Applications (March 1-4, 1999, Aizu-Wakamatsu, Japan) IEEE Computer Society, Los Alamitos, California, 1999, pp. 234-241.

7. S.I. Vyatkin, B.S. Dolgovesov, A.V. Yesin et al. "Geometric Modeling and Visualizatiom of Free Forms" // Autometry - 1999.-N6.

8. S.I. Vyatkin, B.S. Dolgovesov, A.V. Yesin et al. “The Visualization of Free Forms and Voxel Volumes in Real Time” // - Graphicon’99 Proceedings, S. Klimenko et al. (Eds). 1999.

9. S.I. Vyatkin, B.S. Dolgovesov, A.V. Yesin et al. "Algorithm of Visualization of Free Form Surfaces" // Software and Systems - Moscow, 1999.

10. S.I. Vyatkin, B.S. Dolgovesov, A.V. Yesin et al. Geometric Modeling and Visualization of Functionally Defined Objects // Optoelectronics, Instrumentation and Data Processing. – 2000 -N6.

11. S.I. Vyatkin, B.S. Dolgovesov, A.V. Yesin et al. Voxel Volumes Volume-Oriented Visualization System // Computer Graphics and Geometry, Internet Journal http://cg-g1.newmail.ru/smi99_1/VYATKIN.htm

Литература

1. Вяткин С.И, Фотореалистичная визуализация цифровой местности, свободных форм и тематических текстур в системе визуализации в реальном времени Voxel-Volumes / Долговесов Б.С., Овечкин В.В., Чижик С.Е., Каипов Н.Р. - GraphiCon 97, Москва.

2. Савченко В.В. Новый подход в геометрическом моделированиии: распределение и аппаратная реализация перспективы / Пасько А.А., Вяткин С.И. - Международная конференция по компьютерах и устройствах связи CODEC-98. Калькутта, Индия, 1998, стр. 285-290.

3. Вяткин С.И. Синтез виртуальной среды с объектно-пространственным рекурсивным подразделением / Долговесов Б.С., Чижик С.Е.-Graphicon'98 Материалы, С. Клименко и др. (Ред). 1998.

4. Вяткин С.И. Система объемной визуализации в реальном времени Voxel-Volumes./ Долговесов Б.С., Есин А.В. - Международная конференция - распознавание объектов и анализа изображений, Новосибирск-1998.

5. Вяткин С.И. Визуализация неоднородных структур в режиме реального времени с Multi-Level Рекурсивные Ray Casting / Долговесов Б.С., Есин А.В. - Международная конференции - распознавание объектов и анализ изображений, Новосибирск-1998.

6. Вяткин С.И. Объемно-ориентированная система визуализации Voxel-Volumes / Долговесов Б.С., Есин А.В. - Международная конференция по Shape моделированию и приложениях (1-4 марта 1999 года, Айдзу-Вакамацу, Япония) Компьютерное общество IEEE, Лос-Аламитос, Калифорния, 1999, стр. 234-241.

7. Вяткин С.И. Геометрическое моделирование и визуализация свободных форм" / Долговесов Б.С., Есин А.В. - Autometry - 1999.-N6.

8. Вяткин С.И Визуализация свободных форм и воксельных объемов в реальном времени" / Долговесов Б.С., Есин А.В. - Graphicon'99, С. Клименко и др. (Ред). 1999.

9. Вяткин С.И., Алгоритм визуализации формы поверхности без программного обеспечения и систем / Долговесов Б.С., Есин А.В. - Москва, 1999.

10. Вяткин С.И. Геометрическое моделирование и визуализация функционально заданных объектов / Долговесов Б.С., Есин А.В. - Оптоэлектроника, аппаратура и обработка данных. - 2000 -N6.

11. Вяткин С.И. Объемно-ориентированная система визуализации Voxel-Volumes / Долговесов Б.С., Есин А.В. - Компьютерная графика и геометрия, Интернет-журнал *http://cg-g1.newmail.ru/smi99_1/VYATKIN.htm*

Рецензія/Peer review : 22.12.2014 р. Надрукована/Printed :14.1.2015 р.