UDC 681.51:622.788 https://doi.org/10.31721/2414-9055.2016.2.2.4

LEARNING SYSTEM OF THE LIBRARY OPENGL IN 3D-GRAPHICS USAGE

Zelensky A.S., PhD, Professor, Lysenko V.S., PhD (Econ), Associated Professor.
Kryvyi Rih National University

Abstract. The results and brief description of the developed software "Graf10", which is used in the educational process, are presented. This software package is implemented in the programming language Visual C + + 2010 with the help of the open graphics library OpenGL. Special attention is given to perspective and orthographic projection, matrix projection and models, depth buffer management and overlay, lighting and shadows, textures, anti-aliasing and surface modeling. The article concentrates on mathematical basis of the tasks: multiplication, matrix transposition, the transition from one coordinate system to another, defining plane-given projection, etc. When working with curves and surfaces, authoring constructing Bezier curves and surfaces, polynomials in 2D and 3D graphics, and interpolating splines and Hermite splines, curves and surfaces based on B- spline are presented.
Keywords:

Introduction. A lot of attention is paid to 3D-graphics in today's world of information technology. It takes place in solving various problems, where the use of 3D- graphics is necessary or appropriate.

To study 3D- graphics without practical examples is rather time-consuming. It is necessary to develop the software for reflecting the main features of 3D- graphics library using OpenGL. OpenGL and DirectX are two libraries mainly used to work with 3D-graphics. The advantage of the OpenGL library is its independence from the operating system. Thus, applications for the operating systems Windows, MacOS, IOS, Android, etc. can be developed with OpenGL, while DirectX only works for Windows.

The purpose of the article is to study the OpenGL library for reporting results of solving problems in 2D and 3D.

Materials and methods. OpenGL is one of the most popular interfaces for application development in the field of two-dimensional and three-dimensional graphics [1-6].

The software is developed in the programming language Visual C + + 2010 with the help of the OpenGL library. It is used by the Department of Computer Science and Applied Software in the educational process for subjects related to computer graphics.

The developed software package "Graf10" consists of 6 core modules, each of which contains separate task. Module selection and then its tasks are performed through the menu. The structure of the user menu "Graf10" is shown in Fig. 1, where the following modules are identified:

- working with orthographic projection.
- working with graphics primitives and buffers.
- geometric transformations.
- working with lighting and shadows.
- working with pixels and textures.
- work with curves and surfaces.

A universal structure that allows expanding the possibilities of software complex on new tasks and improving the developed ones were worked out to perform each of the presented tasks. While training, students receive the source code of programs as modules, so they can use it efficiently performing the tasks.

Some examples from the book "OpenGL. Superbook" [1] and authoring are included in this software package. Examples of this book are converted from a console-mode C + + language to Visual C + + 2010 under the MFC library and adapted to the structure proposed by the authors.

Macro flowchart of the task solution included into the package "Graf10" is shown in Fig. 2. While application "Graf10" is being downloaded, 3D- graphics (function Init_3D) is being initialized. The task is performed through the user menu.

Results. Sequence of the selected task:

When you click a menu option, response function OnCommand() is connected. There, depending on the requirements of the selected task, the following features can be added: creation of a context menu, texture initialization, timer setting, and the choice of the projection, setting of light and shadow parameters, working with the arrow keys.

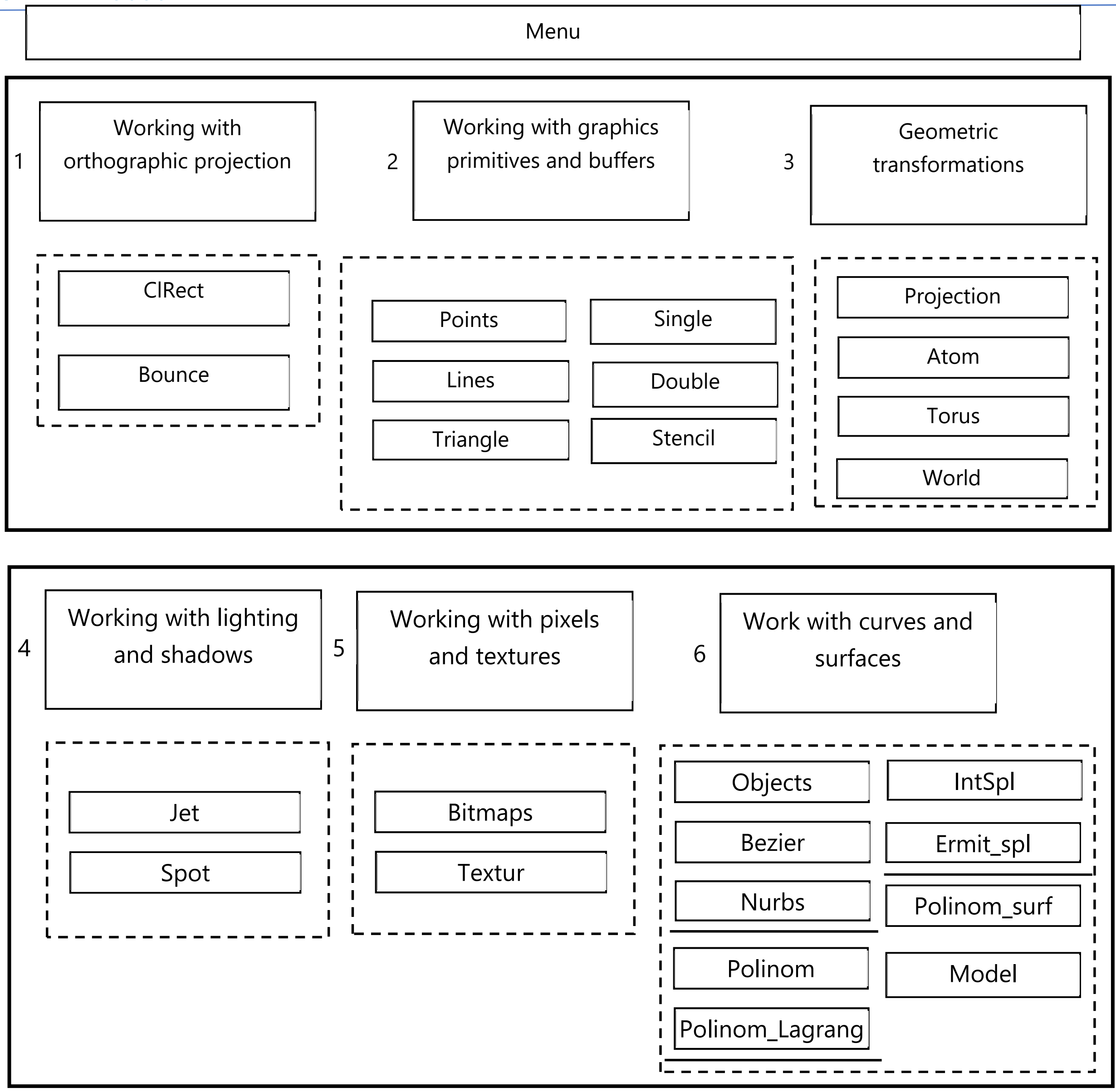

Figure 1. Structure of the user menu "Graf10"

Response function OnKeyDown() is connected with the processing of messages from the keyboard. Depending on the key arrows, object rotating relatively the coordinate axes, as well as surveillance cameras is done.

Response function OnPaint() is connected with explicit call Invalidate() or when it is necessary to redraw the window. The necessary function of task implementation connects through the code to it.

Brief characteristic of complex software tasks "Graf10" (see Figure 1), which consists of six modules:

1. Simple tasks of figure output in Ortho-projection are implemented during the first module.

2. Simple primitive tasks, such as points (Points), lines (Lines), triangles (Triangle), etc. are realized in the second module. Working with single-, double-buffering, and stencil buffer are shown (Single, Double, Stencil).

3. The third module covers the following tasks:

3.1. Projection is an example how to work with perspective and orthographic projections. The study is illustrated by the example of a cube. When orthographic projection is used, cube sides look all the same. When perspective projection is used, the back face of the cube appears smaller than the front, as it is the farthest from the observer.

In the latter case, the cube looks like a truncated pyramid. Different angles and distance to the observer are simulated during training.

These lead to decrease or increase of the cube size, as well as a change in relations front and back face of the cube.

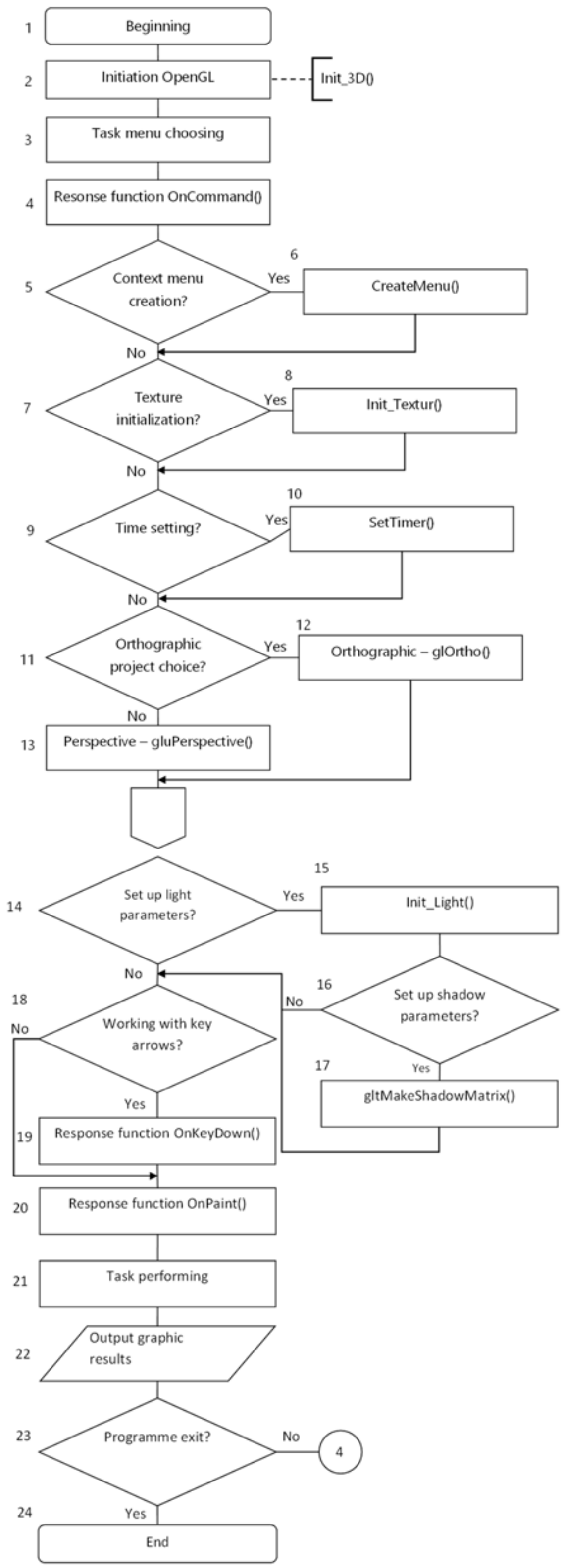

Figure2. Block diagram of task solving

3.2. Atom is an example of a dynamic model of the atom. Three electrons move in their own orbits around the atom's nucleus here. Electrons and nucleus of an atom are presented in the form of balls. Each electron orbit is formed by a rotational angle relative to the core and the subsequent displacement. On the basis of this task, students simulate various orbits, and set the rotation speed of electrons due to the timer and angle of rotation.

3.3. Torus is the construction of a torus with the help of linked triangles. The output is shown in Fig. 3.

3.4. World is an example of the work with camera and actors.

Actors are objects that can move and have their own frame of reference or a local coordinate system. To represent the reference system, data structure that contains a point in space (vLocation), vector pointing forward (vForward) and vector pointing up are used (vUp). Camera is the position of an observer. It is necessary to modify these three vectors to convert the camera. Thus, the vector vForward changes when the camera turns, and the position of the observer vLocation changes when the camera moves. The study of this task is based on the construction of its own world of object- actors (World) and installing surveillance camera. Then, the change of the position of surveillance camera with matrix OpenGL changing is performed with the help of arrow keys.

4. The fourth module deals with lighting and shadows of objects.

The task Jet shows drawing a plane using light and shadow (Fig. 4) [1]. The aircraft is casted in a form of triangles. When the plane is turned with the help of keys cursor its parts (triangles) are lit, as well as object shadows are changed. Shadow of the object is its projection on the horizontal plane.

Task Spot shows the work with a floodlight. The moving of the light source, which illuminates the scene objects from different sides, is done with the help of cursor keys.

5. The fifth module shows working with textures and pixel operations.

The example of Bitmaps shows the ability to download and display raster images with the use of pixel operations.

Studying this example, the student opens a random file in *bmp* or *tga* and selects pixel operations using the context menu: image rotation, scaling, inverting, black and white mode, red, green and blue colour channels, etc.

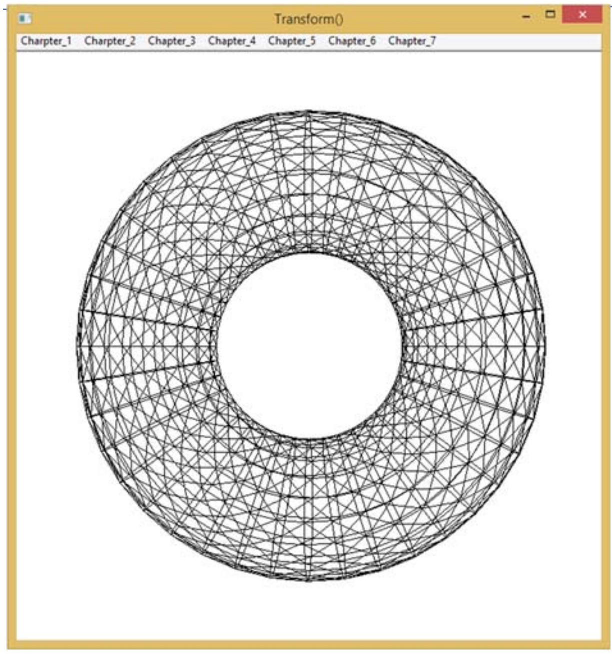

Figure 3. The result of the work of Torus Programme

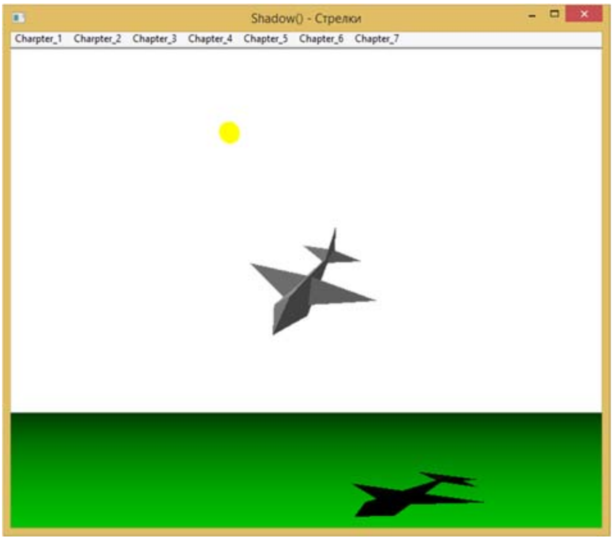

Figure 4. The result of the work of Jet programme which uses light and shadow

The example of Textur demonstrates different features of the textures (Fig. 5).

The first way is the binding of the texture coordinates to the coordinates of the object. The coordinates (0, 0) characterize the lower left corner of the object, (1, 1) characterize the upper right corner. These are lower-left and upper-left squares in the figure. If it is necessary to bind multiple textures to an object, i.e. two textures horizontally and three vertically (top box on the right), the following coordinates are set (Fig. 6).

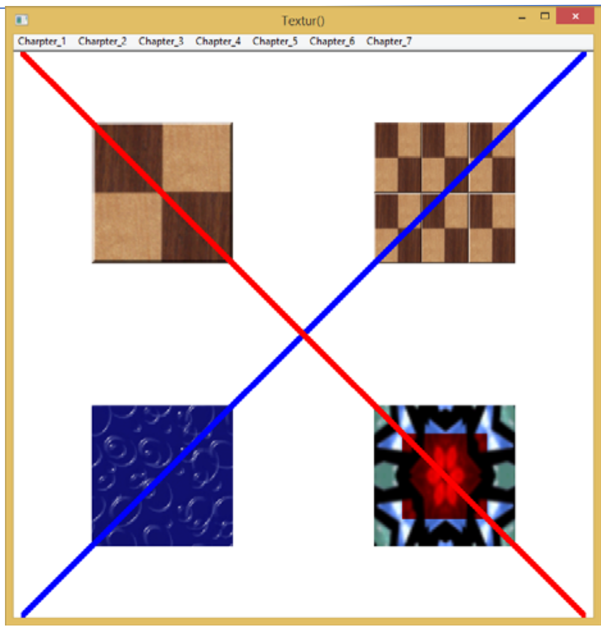

Figure 5. Result of the work of the programme of output of texture (Textur)

The second way is the automatic generation of texture coordinates (lower right square) in which the texture is fully integrated into an object.

Thus, in the process of training the student binds textures to various objects, and performs automatic generation of textures.

6. The sixth module deals with the output curves and surfaces. The first example of Objects demonstrates the work with standard Glu- and Glut- objects. Authoring constructing Bezier curves and surfaces, polynomials in 2D and 3D graphics, and spline interpolation and Hermite splines, curves and surfaces based on B- spline are presented. Moreover, the comparative analysis of the carcass, the bar models and a model which uses B- spline are done.

Task characteristic of the 6th module:

6.1. Objects is an example of standard Glu- and Glut- objects: sphere, cylinder, disc with a hole, part of a disk with a hole, kettle, cone, torus, tetrahedron, hexahedron (cube), octahedron, icosahedron, dodecahedron. The choice of an object through the context menu is allowed, as well as the way of taking the object (in the form of lines GL_LINE or filled with GL_FILL).

6.2. Bezier is an example of output of Bezier curve and surface.

6.3. Nurbs is an example of output of Nurbs- surface with the help of weight coefficients.

6.4. Polinom is an example of the construction of a polynomial in 2D-graphics.

Polynomial model in a general form is presented: $f(x) = A_0 + A_1 x + A_2 x^2 + ... + A_M^M$, or $f(x) = \sum_{i=0}^{M} A_i x^i$,where M is the degree of the polynomial.

Fig.7. presents 8th degree polynomial for 9 control points which are defined by their coordinates. Studies have shown that whether the degree of the polynomial is close to the number of control points, despite an increase in the coefficient of determination R^2, the result of curve building gets worse [8].

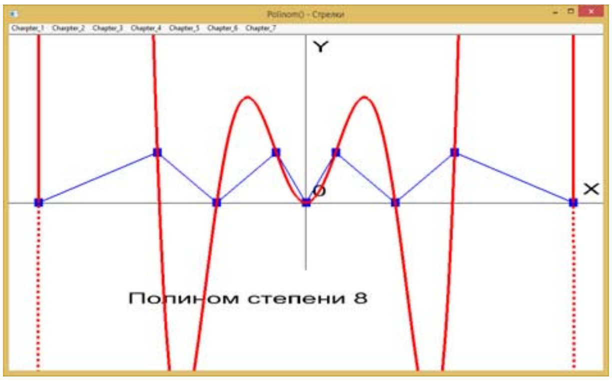

Figure7. Interpolation done by 8th degree polynomial

6.5. Polinom_Lagrange is an example of constructing polynomial interpolation with the help of Lagrange's formula. The following interaction which is used to calculate the points of coordinates:

$$y(x) = \sum_{i=0}^{n} y_i \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j}.$$

6.6. IntSpl is an example of the construction of the natural cubic spline [8], in which the first and second derivatives are continuous. This spline is global, that is, the change of one control point affects the entire behavior of the curve. Fig.8. shows the result of the implementation of the natural interpolation cubic spline.

6.7. Ermit_Spl is an example of the construction of Hermitian spline, which, unlike the natural cubic spline, is local. It is necessary to set the corresponding values of the first derivatives, except the values of the control points, to build it. The change in derivative control point affects only the neighboring points. Thus, it is possible to control the first derivative of the starting points in order to change the behaviour of the curve.

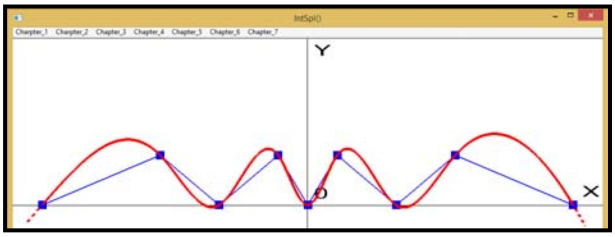

Figure 8. Interpolation cubic spline

Figure 9 represents Hermitian interpolation cubic spline.

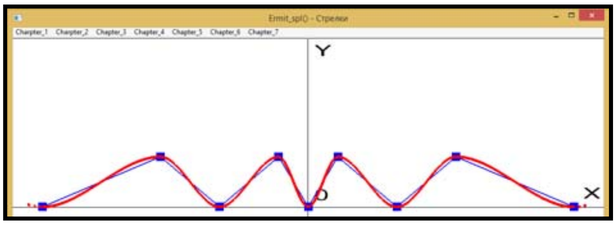

Figure 9. Hermitian interpolation spline

6.8. Polinom_Surf is an example of the construction of a polynomial surface $z = f(x, y)$.

Fig.10. represents degree 3 polynomial in space.

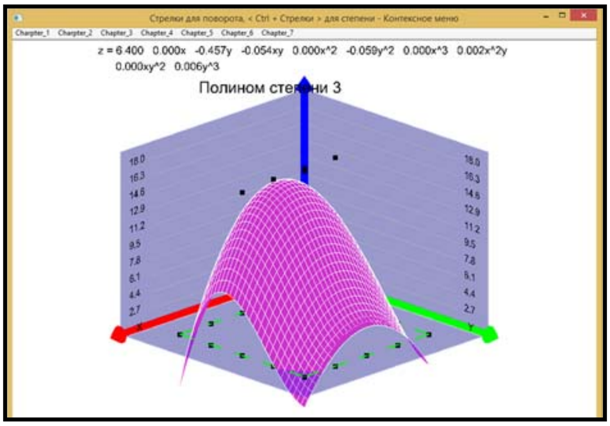

Figure 10. Smoothing the surface of the 3rd degree polynomial

To determine the polynomial according to the three-dimensional space $z = f(x, y)$ the algorithm proposed by the authors is used [8].

For 5th degree polynomial of this dependence can be expressed as

$$\begin{aligned} z = {} & A_0 + A_1 x + A_2 y + A_3 x^2 + A_4 xy + A_5 y^2 + \\ & + A_6 x^3 + A_7 x^2 y + A_8 xy^2 + A_9 y^3 + A_{10} x^4 + \\ & + A_{11} x^3 y + A_{12} x^2 y^2 + A_{13} xy^3 + A_{14} y^4 + A_{15} x^5 + \\ & + A_{16} x^4 y + A_{17} x^3 y^2 + A_{18} x^2 y^3 + A_{19} xy^4 + A_{20} y^5, \end{aligned}$$

where A_0, A_1, ..., A_{20} are polynomial coefficients to be determined.

The number of coefficients of the indeterminate (linear equations) in dependence on the degree of α polynomial is defined by the formula

$$K = \frac{(\alpha+1)(\alpha+2)}{2}.$$

For 5th degree polynomial $K = 21$.

Having calculated the partial derivatives of the parameters A_i, a system of 21th linear equations is obtained.

$$\sum_{s=0}^{20} a_{\ell s} A_s = b_\ell, \quad \ell = 0,1,\dots,20 \text{ i.e.}$$

$$\begin{cases} a_{00}A_0 + a_{01}A_1 + \dots + a_{0\,20}A_{20} = b_0 \\ a_{10}A_0 + a_{11}A_1 + \dots + a_{1\,20}A_{20} = b_1 \\ \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \\ a_{20\,0}A_0 + a_{20\,1}A_1 + \dots + a_{20\,20}A_{20} = b_{20} \end{cases},$$

where $a_{\ell s}$ is the coefficients of linear equations; b_ℓ is constant term of the system of equations; ℓ is the line number (equation); S is the column number.

Simple relations for determining the coefficients $a_{\ell s}$ with indeterminate A0, A1, ..., A20 and constant term b_ℓ is:

$$a_{\ell s} = \sum_{i=1}^{n} x_i^{k_\ell + k_s} y_i^{p_\ell + p_s}; \qquad b_\ell = \sum_{i=1}^{n} z_i x_i^{k_\ell} y_i^{p_\ell},$$

where k_ℓ, p_ℓ are exponents which depend on the line number; ℓ; k_S, p_S are exponents which depend on the column number S:

$$k_\ell = c_1(c_1+1)/2 - (\ell+1);$$

$$k_s = c_2(c_2+1)/2 - (S+1);$$

$$p_\ell = \ell - c_1(c_1-1)/2;$$

$$p_s = S - c_2(c_2-1)/2;$$

$$c_1 = \text{int}\left(\frac{1+\sqrt{8(\ell+1)-7}}{2}\right);$$

$$c_2 = \text{int}\left(\frac{1+\sqrt{8(S+1)-7}}{2}\right).$$

6.9. Model is an example of carcass constructing, the bar surface models, as well as models with the help of B- spline.

A closed loop, which covers a given area, is formed by the user with a mouse at the first stage of the surface construction. Then, the closed loop is covered with the correct regular uniform grid with a given step. Index values are researched by interpolation methods in the nodes of the network relatively to the control points. Fig. 11 presents only calculated values of indicators in the network nodes. Checkpoints are not shown to avoid cluttering the figure.

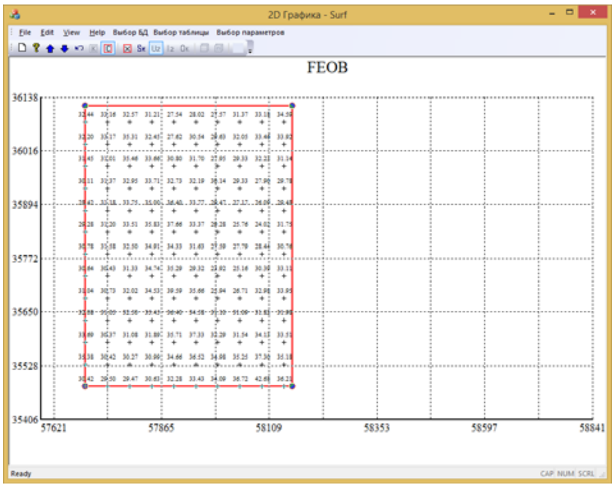

Figure 11. Interpolation network obtained according to the selected indicator

Interpolation nodes can be done in different ways in the network: the inverse distance, polynomial, point Kriging [7].

The step of the network, the number of nearest points to a network node is selected for all methods of interpolation. Step network is determined by taking into account the variability of indicators based on the normalized autocorrelation function.

Exponent is introduced additionally to above mentioned parameters at the interpolation by the method of inverse distance. The index value in the node A is defined as the weighted value:

$$C_A = \sum_{i=1}^{N} C_i\, d_i^{-\alpha} \Big/ \sum_{i=1}^{N} d_i^{-\alpha},$$

where C_A is interpolated value of the index at the node A; N is the number of nearest points; α is the exponent; d_i is the distance between interpolation node; C_i is the value of the index at the point nearest to the node interpolation.

Interpolated values array is used to visualize the spatial variability of parameters in three-dimensional space and is the starting point for constructing surfaces. Comparing the vertical bar (Fig. 12) and wire-frame (Fig. 13) models, a model based on B- spline (Fig. 14), the advantage of spline method is visually obvious over the others methods of variability of estimates. Fig. 12-14 presents $Fe_{general}$, a studied indicator, which is determined according to well testing on the ore mines. The building of wireframes is done through connecting the lines between the interpolation points in two orthogonally related directions. Adjoining rectangular prisms are formed in vertical bar graph models. There are squares with the center in interpolation points and sides equal to the interpolation step of the network in bisections of these prisms. The height of the prisms determines the values in the nodes.

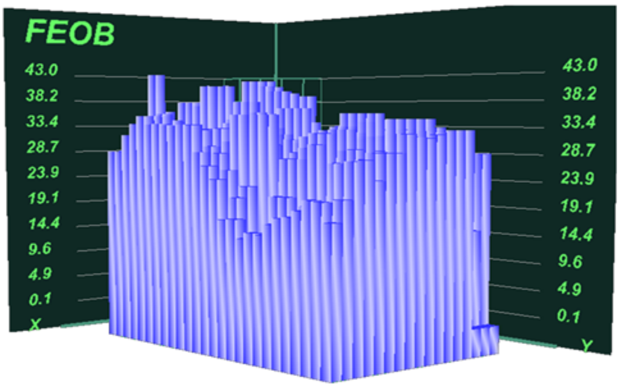

Figure 12. Vertical bar graph of estimates variability

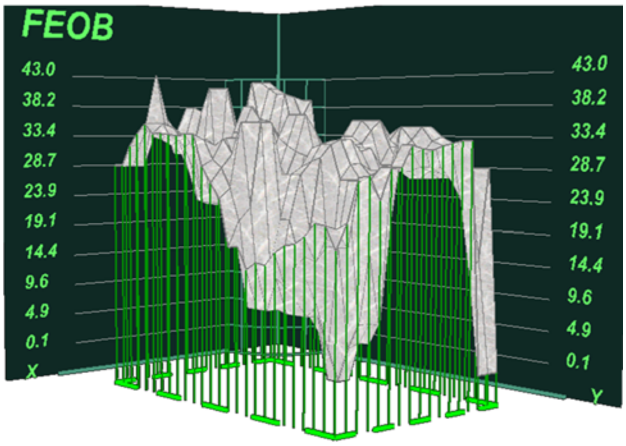

Figure 13. Wireframe of estimates variability

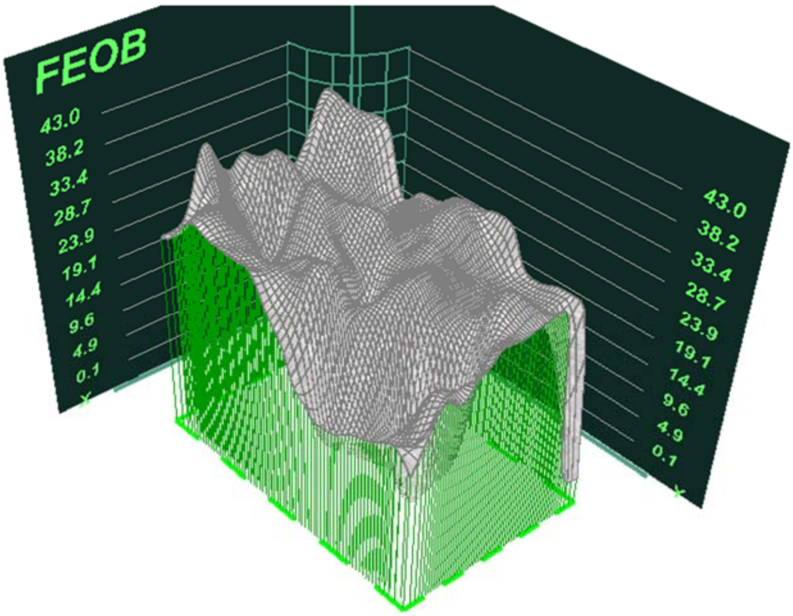

Figure 14. Displaying variability of estimate index with the help of B- spline

The building of topofunctions as NURBS-surfaces (B- spline) is the most effective on the basis of the interpolation network. There are advantages of B-spline usage. Only area of the surface, which is the closest to the data, changes when the source data adjusts. NURBS- surface is controllable, the character of the passage of the surface with respect to the control points, changes their weight influence. Construction of such a surface does not require high valence of studied parameters, high efficiency implementation is provided. [7-11].

NURBS- surfaces constructing using standard OpenGL does not allow managing the surface flexibly, there is no possibility of obtaining the surface coordinates of nodes to perform various calculations. Furthermore, one can select the colour of the surface, and only the surface as a whole. In this regard, its own functions for constructing NURBS - surfaces based on B- splines has been developed.

When constructing NURBS- surface, the following formula is used:

$$P(u,v)=\frac{\sum_{i=0}^{M}\sum_{k=0}^{L}w_{i,k}P_{i,k}N_{i,m}(u)N_{k,n}(v)}{\sum_{i=0}^{M}\sum_{k=0}^{L}w_{i,k}N_{i,m}(u)N_{k,n}(v)},$$

where (M + 1) is number of control points along the axis OX (the numbering of the points from 0 to M); (L + 1) is the number of control points along the axis OY (the numbering of the points from 0 to L); $P_{i,k}$= ($x_{i,k}$, $y_{i,k}$, $z_{i,k}$) are the coordinates of reference points; $w_{i,k}$ is the weighting coefficients of control points; $N_{i,m}(u)$ and $N_{k,n}(v)$ are respectively low B-spline function in the direction of the axes OX and OY; m and n are respectively blending B- spline functions $N_{i,m}(u)$ and $N_{k,n}(v)$; $u\in 0, \ldots, u_{max}$ (u_{max} are maximum value in the node vector u, in the direction of the axis OX, equals $M - m + 2$); $v\in 0, \ldots, v_{max}$ (v_{max} are the maximum value in the node vector v in the direction of axis OY, and equals $L - n + 2$).

Conclusions. Implementation of complex software of tasks "Graf10" is done by the authors in the language Visual C + + 2010 for Windows using tools of open graphics library OpenGL. A universal structure for performing each of the presented tasks is developed. It allows expanding the possibilities of software complex with new tasks and improving the developed ones. The developed software package is used by the Department of Computer Science and Applied Software in the educational process to teach subjects related to computer graphics.

References

1. Wright R. S., Lipchak Jr., B. OpenGL. Superbook, 3rd edition translated from English, Moscow: Williams, 2006.

2. Zelensky O.S, Lysenko V.S, Baran S.V. Operation manuals to self study of standard commands of graphic Book Cabinets OPENGL with the help of language C + +,Kryvyi Rih : KEI KNEU, 2006.

3. Zelenskiy A.S., Lysenko V.S., Churin N.A. Operation manuals to self-study of the mathematical foundations of computer graphics, Kryvyi Rih: KEI KNEU, 2007.

4. F. Hill OpenGL. Programming computer graphics. For professionals, SPb: Peter , 2002.

5. Hearn D., Baker M. P. Computer graphics and standard OpenGL, 3rd publication, Moscow Williams, 2005.

6. Rogers D., Adams J. Mathematical Foundations of Computer Graphics, Wiley, 2001.

7. Zelenskiy A.S., Baran S.V., Lysenko V.S. Automation geological survey software in the management information system ore mines: monograph. Kryvyi Rih, Publishing Center SHEE "NUC", 2012.

8. Zelenskiy A.S., Lysenko V.S. (2013). Construction of curves and surfaces in the solution mining and geometric problems, *Journal of Kryvyi Rih Technical University*, **27**: 186 –190.

9. Zelenskiy A.S. Lysenko V.S., Melnychuk V.I. (2009). Surface modeling career with B-splines */Journal of Kryvyi Rih Technical University*, **23**: 50 – 54.

10. Zelenskiy A.S., Puhanov S.S., Melnychuk V.I. (2009). Using NURBS surfaces for automated contouring variability of quality indicators ore quarries, *Scientific Bulletin NGU*, **6**: 69–73.

11. Zamytsky A.V., Shepelenko M.I., (2015). 3D modeling in thermal power engineering, *Computer Science, Information Technology, Automation*, **1**: 6-10.