Automated visualization of calculation results in the Algozit programming environment

V O Kaledin, A E Paulzen and A D Ulyanov
NBI of Kemerovo State University, 23 Tsiolkovsky Street, Novokuznetsk, 654000, Russia
E-mail: zbrg@mail.ru

Abstract. The functional object diagram of algorithms for constructing a graphical representation of the source and calculated data in the visual programming environment “Algozit” is considered. When defining the functional content of circuit elements (algomats), a built-in interpreted programming language is used, supplemented by graphical operators. Using the built-in language simplifies the modification and support of the developed visual algorithm. Graphic operators use the capabilities of a graphic library built into the visual programming environment. The library allows you to accumulate data on graphic primitives in three-dimensional space and build them in a graphic window in the form of a central projection or stereo image. The graphics window is equipped with display controls.

1. Introduction
Visual programming environment “Algozit” [1, 2] allows you to design computational algorithms and general purpose algorithms. Elements of the functional object network (FON) are algomats. The functions of the algomats are defined in the form of methods of functional classes in C++, which are defined in the library – “factory of objects”. The interpreted language “Jadro” [3] was integrated into the “Algozit” environment to speed up the development process and make the execution of executed algorithms more clear. This allows you to make changes to the functional content of algomats and not to resort to changing compiled program code in C++. This modification allows you to reduce the cost of working time of the developer.

In the study of material objects by numerical modeling methods [4], the problem arises of visualizing the initial and obtained data. Mathematical methods and hypotheses can have significant modifications. The format of the initial data of the model and calculation results may depend on these modifications. Often an individual solution is used to visualize data. To speed up the process of selecting or developing the necessary visualization algorithm, a universal FON was developed in the “Algozit” environment. To build images using the built-in language “Jadro” and its graphical tools.

2. Description of implementation
Figure 1 partially shows the communication structure according to the components of the “Algozit” environment: block 1 in the figure indicates the “Debugger” module with the FON loaded in it for execution; block 2 – library-factory of functional classes, in the methods of which either the compiled code is executed, or – the script code through the interpreter; block 3 – interpreter module of the embedded language “Jadro”, which is equipped with graphical operators; block 4 – graphic library, including the functions of display on the screen, viewing settings and legends.
The developed FON can be considered universal due to general algorithms for the accumulation of topological and geometric data and the availability of modification of network parts for preparing portions of graphic data of various kinds. Algorithms are implemented using the “Jadro” language and its graphical tools in algorithms handlers. These algorithms are available for editing in the runtime constructor of the FON of the “Algozit” environment.

In order to implement support for the core language, the ability to replace existing C++ handlers with scripts with predefined variables has been added to the settings of the algomats. An example of the script settings window is shown in figure 2. Each table item in this window corresponds to a specific method of the algomat. Predefined variables in scripts and their types are defined according to the selected functional class of the algomat.

![Diagram](image)

**Figure 1.** The dependency diagram of the modules of the functional-object programming environment “Algozit”.

**Figure 2.** Algorithm script settings window.

The algorithm for accumulating data for drawing was implemented in the form of FON. Its upper level is shown in figure 3. The results of the calculation in a tabular form are loaded into the memory by the network panels, processed and accumulated in the memory of the algomats “Stress”, “Movement”, “Connection of circuits” and “Form”. The names of the panels correspond to their content. In the figures, the names of the algomats are abbreviated. They serve as input data for image building. Data for construction are formed in the form of points, their attributes and lines in three-
dimensional space by the following language operators: AddContour – add contour, AddNode – add point to contour, SetValueOnTitle – adding point attribute data. Buffer management by commands:

Figure 3. FON top-level for data accumulation and output to the graphics window.

ClearImage – clear image data, CompleteImage – completion of construction.

Before you start drawing, you must reset the previously accumulated graphic information with the ClearImage command. The AddContour command is executed to create a new sequence of points (contour). Its result is a new contour number. After the contour is created, it can be replenished with dots (AddNode). When executing the CompleteImage command, contours are drawn in the order of addition with all points with the specified color and drawing style. The SetValueOnTitle command is used to add point attribute data.

Attribute data is used to construct scalar or vector fields for the selected data component. The script for graphical construction using the prepared data is implemented in the “Script Instead of Drawing” handler of the “Original form + stress” object and is executed when the build command is called during the execution of the algorithm in the “Debugger” module [1,2].

Figure 4 shows the result of constructing the initial and deformed shapes with scaling of the linear components of displacements in the window of the Graphics module. The original shape is drawn with dashed lines in gray, the shape with solid movements taken into account is constructed with black lines. The components of the source or calculated fields at the internal points of the contours can be represented by colored dots with a legend.

The window contains interactive viewing controls: rotation of the object relative to the coordinate axes, setting the color legend for scalar fields, construction of vector fields and diagrams, building a deformed shape relative to the original, display image element captions, anaglyph stereo image.

The constructed scheme (FON) refers to the finished tabular data. Determining the method of drawing and selecting points from a piece of data is carried out separately for each view, by means of switchable FON pages.
Figure 4. An example of a graphical construction of a model of a plate with stresses and displacements of the scale.

Figure 5 shows an example of one of the switchable web pages. Each switchable page (subnet) provides the correct selection of data from the same sources as in the calculations. Therefore, the network can be adapted for any algorithms implemented within the framework of the considered programming environment. The accumulation of data is carried out in the algorithms “Stress subsampling” (stress table at points of the current element) and “Stress subsampling plan” (stress table at the plan points of the current element). The accumulation algorithm is set by a script in the “Jadro” language.

Figure 5. A fragment of a nested switch network that implements data accumulation for drawing.

3. Conclusion
Thus, the developed standard FON allows visualization of calculation results automatically for many data types. If the data type is not supported by the FON, then the user has the option to enable its support. To do this, you need to add a small number of typical algomats with functional content in the “Jadro” language to a switchable page.
Acknowledgments
This work was partially supported by the grant from the NBI of KemSU (grant agreement No. 3-05 / 1GR-18 dated 10/18/2018).

References
[1] Ulyanov AD, Kaledin V O et al 2016 Proc. of XVI International Scientific and Practical Conference in 3 parts (Chita: Transbaikal State University) part 2 pp 141–145
[2] Ulyanov A D and Kaledin V O 2016 Proc. of the V Int. Sci. and Pract. Conf. in Prospects for the Innovative Development of the Coal Regions of Russia (Prokopyevsk: Branch of KuzSTU) pp 350–352
[3] Kaledin V O Interpreter “Jadro” The Certificate on Official Registration of the Computer Program No 2017612706 publ. 02.03.2017
[4] Bathe K Ju 1996 Finite Element Procedures (New Jersey: Prentice Hall) p 1050