Bull. Nov. Comp. Center, Comp. Science, 26 (2007), 65–73

Correct visualization of solution spaces in the UniCalc system

E. Yu. Botoeva, E. S. Petrov

Abstract. The paper describes an approach to visualization of solution spaces in computer modeling problems. The advantage of the approach is that it is applicable to visualization of solution spaces of arbitrary spatial structure (e.g. consisting of multiple components, containing cuts, accumulation points, etc.) An implementation of the approach is built into the UniCalc system.

1. Introduction

Computer modeling problems often arise in knowledge-intensive areas of industry and design. The purpose of those problems is to imitate real objects and processes which are too expensive, laborious or dangerous to carry out or construct in reality. Such are explosions, natural phenomena, large engineering structures and many other things.

In computer modeling, spatial structure of solution spaces is the key to understanding critical behavior, critical states, etc. in the modeled processes and objects. Since modeling problems usually involve many parameters and constraints (equations and inequalities), convenient tools for visualization of solution spaces of computer modeling problems are indispensable in knowledge-intensive areas of industry and design.

Different visualization methods may produce different images for the same solution space; even images of some 2-dimensional solution spaces may be incorrect. Incorrect visualization is usually caused by discontinuities and uncontrolled accumulation of rounding errors.

Correct visualization is a process that creates 2d "visually explicit" images of solution spaces in such a way that each point of the solution space is represented by some point of the 2d image. The price of correctness is the fact that the 2d image usually contains extra points which do not represent any point from the true solution space. Visual explicitness of the 2d image is achieved by assigning appropriate visual characteristics (color, brightness, etc.) to its points.

The UniCalc system is positioned as a convenient tool for reliable solution of computer modeling problems. The above considerations inspired development of the graphic module for correct visualization of solution spaces in the UniCalc system.

Let us consider an example that explains operation of the UniCalc system and illustrates the need in a tool for correct visualization of solution spaces.

Assume that the constraints are $y \leqslant x$; $y \geqslant \frac{x}{2}$; $y \leqslant 3 - x$. The corresponding solution space is shown by the dark triangle in Figure 1. Without a visualization tool, the best the users of the UniCalc system can get is the interval solution $y = [0, 1.5]$, $x = [0, 2.0000000000000045]$. This interval solution is shown by the light rectangle in Figure 1. It is a correct (contains all solutions), but rough (contains many additional points), representation of the solution space.

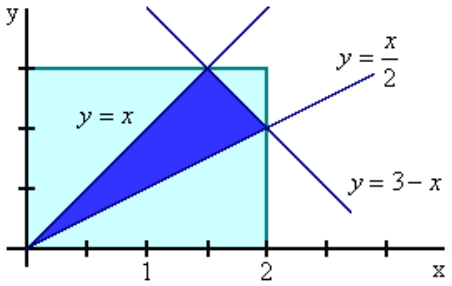


Figure 1. The interval solution and the solution space

To understand the structure of solution spaces, we can enumerate individual interval solutions of a preset width. Thus we get a set of small rectangles covering the solution space. Such representation of the solution space is correct and more exact than the interval solution but still is not visually explicit. To make it visually explicit and suitable for qualitative analysis of spatial structure of solution spaces, we need a specialized visualization tool.

The paper is structured as follows. Section 2 contains a short overview of the visualization methods in scientific computing. Section 3 gives an overview of visualization of solution spaces in the UniCalc system. Sections 4, 5, and 6 cover the major issues of visualization in the UniCalc system. Section 7 gives concluding remarks.

2. Related work

Plotting graphs of real valued functions of 1 or 2 parameters is an example of visualization of solution spaces for constraint systems of the form $y = f(x)$ or $y = f(x, z)$. This kind of visualization is most commonly performed with the help of so called mesh methods. The plotted function is evaluated at the points of some mesh. This operation produces a set of 2d or 3d points ($\{x, y = f\ (x)\}$ or $\{x, z, y = f\ (x, z)\}$). These points are connected by line segments (or spanned by simple surfaces and projected onto a 2d space) to form the image that visualizes the graph of the function.

The mesh methods are built into many systems of computer algebra such as MathCad, Mathematica, MathLab, etc. [5, 6, 7, 8]. The mesh

methods may visualize solution spaces (plot graphs) incorrectly even for simple univariate functions. Figure 2 shows plots of the graph of the function $g(y) = y \times \frac{y-1.5-0.000001}{y-1.5}$ constructed for different meshes in the MathCad system. All the plots are incorrect near the discontinuity at $y = 1.5$.

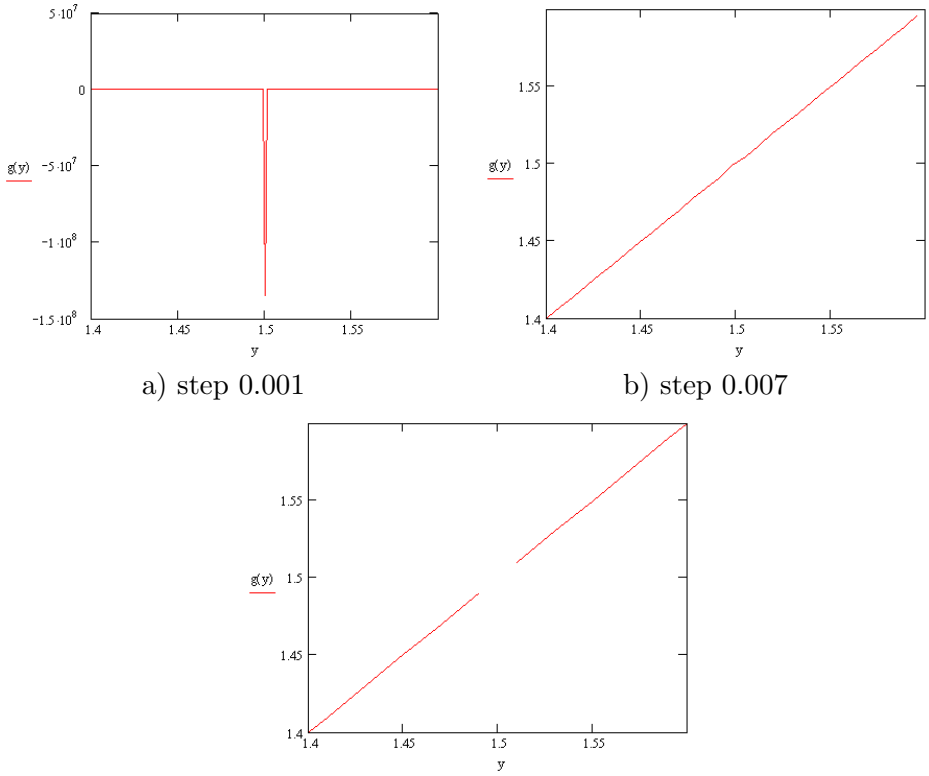


Figure 2. Incorrect plots of the graph of the function g(y) constructed for regular meshes with steps 0.001, 0.007 and 0.01 in the interval [1.4, 1.6]

The root cause for incorrectness of the mesh methods is that these methods assume that the plotted functions are continuous.

Authors of [2, 3, 4] handle this drawback with the help of interval analysis. Consider, for example, the method proposed in [2] to visualize solution spaces for inequalities of the form $f(x, y) \leqslant 0$ specified by mathematical formulas containing variables x and y.

The plotting algorithm is a loop that processes a list of rectangles. Initially, the list of rectangles contains the entire plotting area. The loop terminates, if there are no more rectangles or all the rectangles are smaller than the resolution required.

For each rectangle in this list, the range of $f(x, y)$ is evaluated by means of interval analysis (which handles both continuous and discontinuous functions). If this range is located to the left of 0, the corresponding rectangle is a part of the solution space and is plotted with black. If this range is located to the right of 0 and does not contain 0, the corresponding rectangle is not a part of the solution space and is plotted with white. If this range contains 0 in its interiority, the corresponding rectangle is plotted with red. It is then divided into smaller rectangles and these rectangles are inserted into the list of rectangles.

In the following sections we generalize this approach for the case of arbitrary sets of relations specified by mathematical formulas containing an arbitrary number of variables.

3. Overview of visualization in the UniCalc system

Visualization of a solution space in the UniCalc system is divided into two steps:

1. Constructing a set of rectangles or parallelepipeds which together cover the solution space;
2. Plotting these rectangles or parallelepipeds with the help of external graphics tools.

If the solution space has more than 3 dimensions, it is projected onto 2 or 3 coordinate axes chosen by the user and it is this projection that is actually visualized. Thus we assume that (1) the dimension of the solution space is at most 3 and that (2) the axes of the user's choice correspond to the parameters x, y, and z.

Rectangles and parallelepipeds whose sides (edges) are parallel to the coordinate axes are a special case of *boxes*. Rectangles are 2-dimensional boxes; parallelepipeds are 3-dimensional boxes.

The sets of 2d or 3d boxes which together cover the solution space are called *coverings* of the solution space.

Coverings of the solution space are constructed with the help of a so-called method of *subdefinite calculations* [1]. Given a computer modeling problem, that is a set of constraints, subdefinite calculations produce a bounding box for its solution space. A box $I \times J$ (respectively, $I \times J \times K$) is *subdefinitely consistent* with a modeling problem C, if subdefinite calculations produce a non-empty bounding box for the solution space of $C \cup \{x \in I, y \in J\}$ (respectively, $C \cup \{x \in I, y \in J, z \in K\}$).

Now let us consider in detail construction of coverings for solution spaces in the UniCalc system.

4. Construction of coverings of solution spaces

Coverings of a solution space are constructed repeatedly until all the boxes are smaller than the resolution required. The initial covering consists of the bounding box obtained by subdefinite calculations for the solution space. Subsequent coverings are obtained one from another by a series of *refinements*.

Refinement of a covering of the solution space consists of the following steps:

1. Each box is replaced with 2^n smaller boxes of equal size (n is 2 for rectangles and 3 for parallelepipeds);
2. Those smaller boxes that are not subdefinitely consistent with the modeling problem are deleted from the covering.

Figure 3 shows the first 4 coverings of the solution space of the modeling problem $\{x^2 + y^2 = 1\}$.

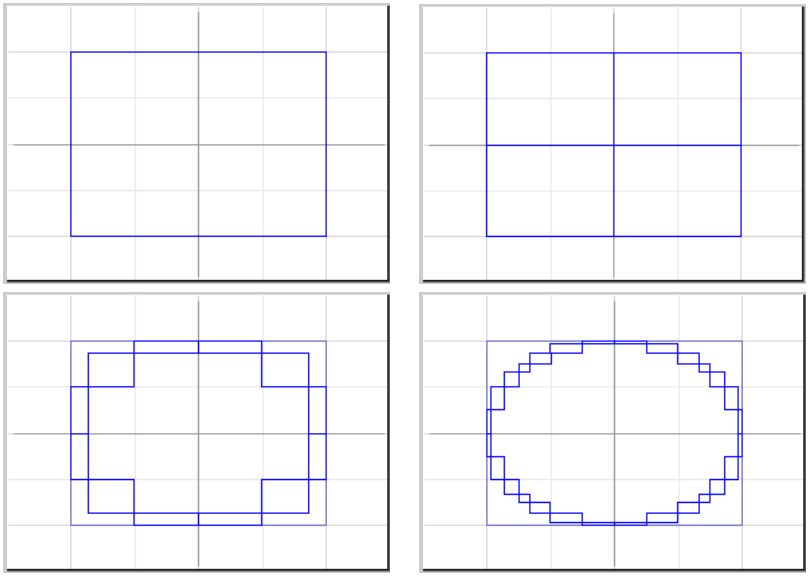


Figure 3. The first 4 coverings for the equation $x^2 + y^2 = 1$

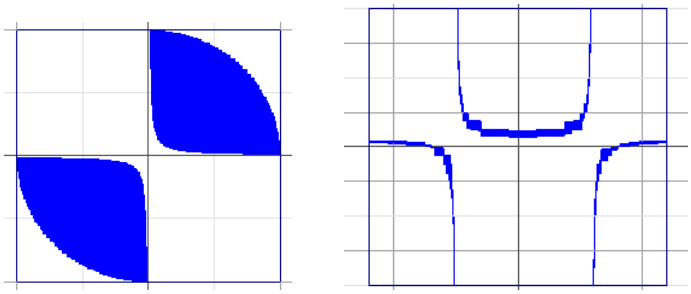
The main advantage of this covering algorithm is that there is no need to analyze the spatial structure of the solution space (search for isolated components, cuts, accumulation points, etc.)

A correct visualization of any covering generated by the covering algorithm is a correct visualization of the solution space because all coverings indeed cover the solution space. Consider in detail visualization of coverings in the UniCalc system.

5. Visualization of 2d coverings

Coverings of 2d solution spaces are 2d objects and can be directly plotted on the screen. To this end, the UniCalc system uses the standard Windows Graphics API. The covering is scaled to fit into the window opened by the UniCalc system for plotting. In addition, the coordinate grid and coordinate axes are plotted.

Figure 4 shows two examples of visualization of 2d solution spaces of different structure.



a) $x^2 + y^2 = 1; 100xy > 1;$ b) $y = \frac{(x+3.5)(x-4)}{(x+2.5)(x-3)};$

Figure 4. Visualization of 2d solution spaces of different structure

6. Visualization of 3d coverings

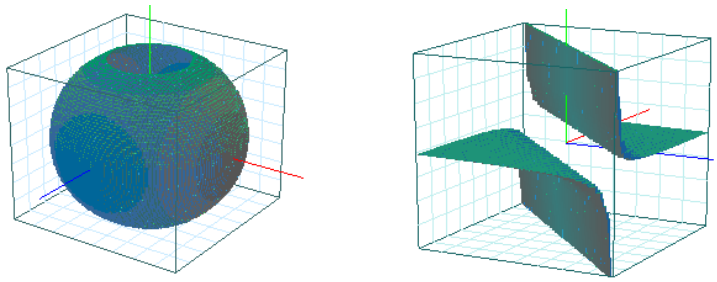
Coverings of 3d solution spaces are 3d objects. To plot 3d objects on the 2d screen the UniCalc system uses the Open Graphics Library (OpenGL) [9]. The coverings are transformed into OpenGL data and passed to OpenGL. Then OpenGL removes all invisible lines and faces and plots the resulting 2d images on the screen.

The UniCalc plots coverings of 3d solution spaces in two modes: "stepped" and "smoothed".

The stepped mode is the simplest. All the boxes from the covering are passed to OpenGL individually. The plot consists of the plots of individual boxes and looks like a set of small steps. To make the image more explicit visually, the faces of the boxes can be colored in 3 slightly different colors depending on their orientation with respect to coordinate axes or their distance from the center of the covering.

Figure 5 shows two examples of visualization of solution spaces of different spatial structure in the stepped mode.

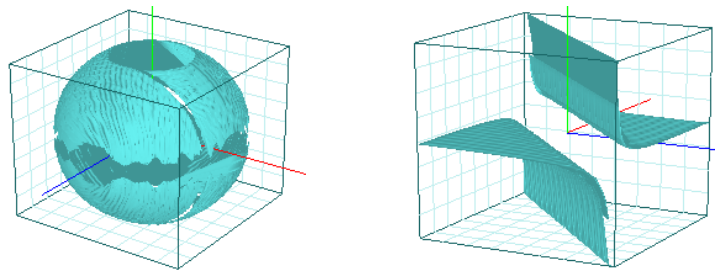
In the smoothed mode, the UniCalc system tries to produce a "visually smooth" image of the solution space without guaranteeing the correctness of visualization. We continue implementation of this mode and report our preliminary results here.



a) $x^2 + y^2 + z^2 = 1; y = [-0.9, 0.9]$; b) $y = \frac{1}{x+z}$;

Figure 5. Visualization of 3d solution spaces of different spatial structure in the stepped mode

The UniCalc system passes to OpenGL the data which describe a set of Bezier surfaces. These surfaces span the exterior (with respect to the solution space) vertexes of the boxes which form the top view and the bottom view of the covering. The plot consists of a set of small smooth "patches". Such an approach works well for "thin" solution spaces like the one shown in Figure 6b and not so well for "voluminous" solution spaces like the one in Figure 6a.



a) $x^2 + y^2 + z^2 = 1; y = [-0.9, 0.9]$; b) $y = \frac{1}{x+z}$;

Figure 6. Visualization of 3d solution spaces of different spatial structure in the smoothed mode

7. Conclusion

In this paper the authors describe an approach to visualization of solution spaces of computer modeling problems. The advantage of the approach is that it is applicable to visualization of solution spaces of arbitrary spatial structure (e.g. consisting of multiple components, containing cuts, accumulation points, etc.) Moreover, this approach is correct in the natural sense.

An implementation of the approach is built into the UniCalc system. Also see Section 8 for the GUI of the graphics module of the UniCalc system.

In this paper the authors were able to solve the problem of smooth and correct visualization *only partially*. In the future they will continue to work on a solution to this problem.

References

[1] Narin'yani A. Subdefinite models: a big jump in knowledge // Proc. the East-West Conf. on Artificial Intelligence (EWAIC93). — Moscow, Russia, Dec 1993. — P. 227–231.

[2] Tupper J. Reliable Two-Dimensional graphing methods for mathematical formulae with two free variables // Proc. SIGGRAPH 2001. — ACM, 2001. — P. 77–86.

[3] Shou H., Shen J., Yoon D. Robust plotting of polar algebraic curves, space algebraic curves and offsets of planar algebraic curves // J. Reliable Computing. — 2006. — Vol. 12, N 4. — P. 323–335.

[4] Hickey T., Qiu Z., Emden M. Interval constraint plotting for interactive visual exploration of implicitly defined relations // J. Reliable Computing. — 2000. — Vol. 6, N 1. — P. 81–92.

[5] Larsen, R. Introduction to MathCAD. — Prentice-Hall, 1999. — ISBN 0139374930.

[6] Wolfram S. Mathematica: A System for Doing Mathematics by Computer. — Addison-Wesley, 1991. — ISBN 0201515024.

[7] Etter D., Kuncicky D. Introduction to MATLAB. — Prentice-Hall, 1999. — ISBN 0130131490.

[8] Heck A. Introduction to Maple. — Springer, 2003. — ISBN 0387002308.

[9] Shreiner D., Rost R. OpenGL(R) Library. — Addison-Wesley, 2007. — ISBN 0321514327.

8. Appendix

Figure 7 and Figure 5 show the GUI of the graphics module of the UniCalc system in the 2d and 3d cases.

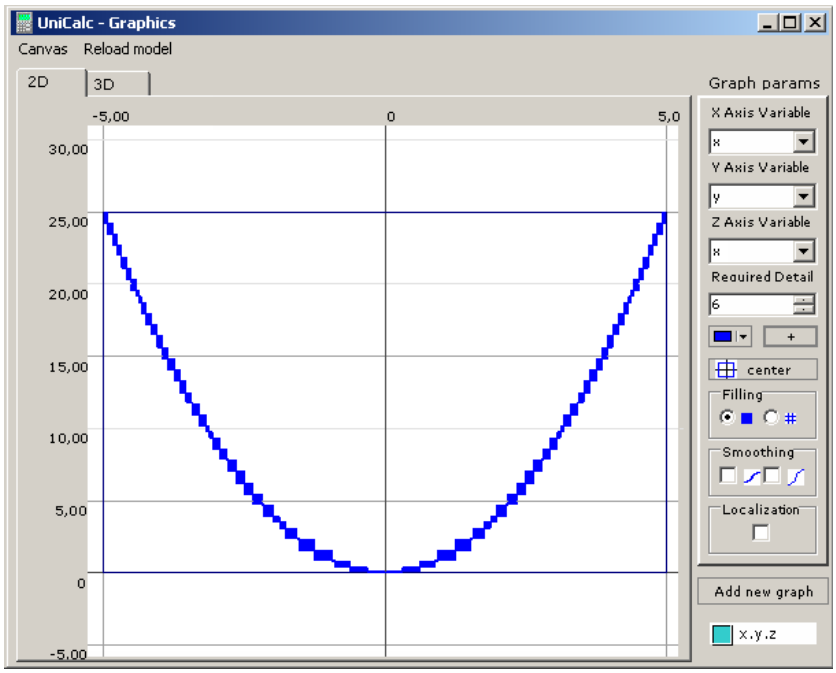


Figure 7. The GUI of the graphics module of the UniCalc system in the 2d case

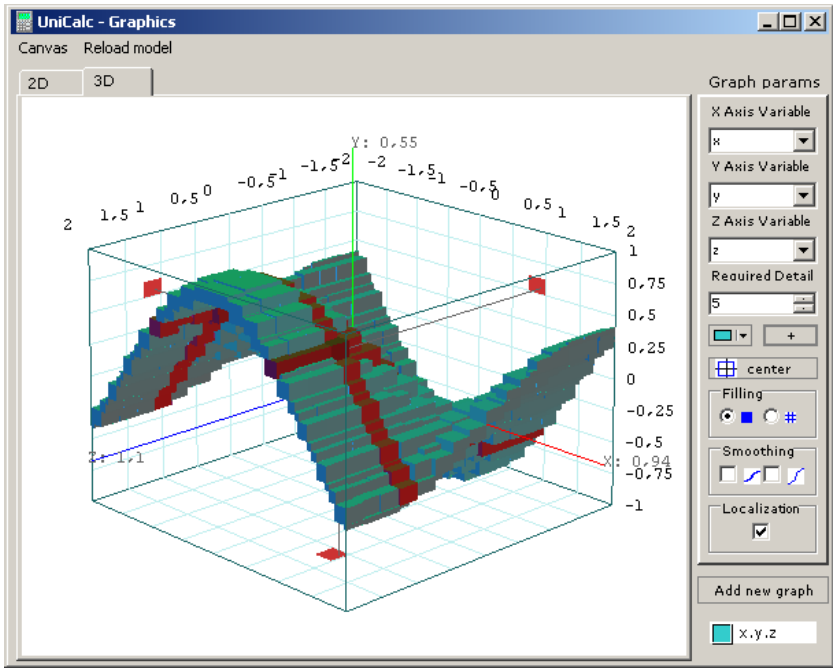


Figure 8. The GUI of the graphics module of the UniCalc system in the 3d case