Attract - Interactive Visualization of Dynamical Systems

Eduard Gröller, Herbert Oppolzer

Technical University Vienna
Institute of Computergraphics
Karlsplatz 13/186/2
A-1040 Vienna, Austria
Tel.: +43(1)58801-4582
FAX: +43(1)5874932
e-mail: groeller@cg.tuwien.ac.at
herbert@cybercafe.co.at

Abstract

An interactive program for visualizing the long term behavior of dynamical systems, e.g., attractors and bifurcation diagrams, is presented. The program allows an easy specification of a set of formulas and constants which describe a dynamical system. This set of equations is then used for displaying the geometrical structure of the long term development of the dynamical system. The user defines the assignment of variables of the dynamical system to at most three spatial axes and one color axis. Viewing parameters, e.g., point of view, zoom, rotation angle, can again be changed interactively. The program is intended to provide researchers working on dynamical systems with a fast visual analysis and experimentation tool.

Keywords: visualization, dynamical system, interactivity, attractor, bifurcation diagram

1. Introduction

Visualization comprises methods and techniques to generate images out of numbers. While computers are well suited to manipulate and process large amounts of data, the results of such calculations often have to be transformed into visual information to be easily accessible to the human analyzer. Visualization techniques are, e.g., hidden line/ hidden surface methods, projections, volume visualization, pseudo surface generation, shading methods, color coding, animation methods [BrCa92], [NiSh89]. Visualization techniques are used in a variety of different application areas, e.g., CAD, architecture, medicine, biology, chemistry, physics. The primary goal of visualization is to increase insight. This can be achieved to an even greater degree by allowing the user to influence and steer a visual analysis interactively.

Dynamical systems are either conservative (energy preserving) or dissipative (energy consuming). Chaos theory investigates dissipative, nonlinear, deterministic dynamical systems [AbSh84], [Deva89], [Tson92]. Nonlinear feedback phenomena often produce chaotic behavior in these deterministic systems. Sensitivity to errors in the initial conditions, thereby, greatly reduces the predictability of such systems. The behavior of dynamical systems can be investigated in phase space, where each degree of freedom (state variable) corresponds to a coordinate axis. A point in phase space completely describes the state of the system at one point in time. The temporal evolution of a dynamical system produces a trajectory in phase space. The long term behavior is characterized by a geometric object in phase space called attractor, e.g., limit point or limit cycle. The long term behavior of chaotic dynamical systems produces objects in phase space which are, due to their geometrical complexity, called strange attractors [Chao89].

Properties of strange attractors are: basin of attraction (each point of the basin of attraction is converging to the attractor), sensitive dependence on initial conditions (points close together follow vastly differing paths in phase space), fractal dimension, mixing (a typical point on the attractor will visit any other point on the attractor with arbitrary precision, the attractor cannot be split into two different attractors). Often a stretch-and-fold mechanism is at work in chaotic dynamical systems. Portions of phase space are stretched (points close together are moved far apart, error sensitivity) and are later on folded together (mixing).

Continuous dynamical systems are usually described by a set of differential equations (vector field) $\vec{x}' = V(\vec{x})$ with initial value $\vec{x}_0$. $V(\vec{x})$ defines the tangential direction of the flow at point $\vec{x}$. Typically the exact solution $\vec{x} = \vec{x}_0 + \int V(\vec{u})d\vec{u}$ of such a system of differential equations can not be determined analytically. Therefore numerical techniques like Euler's method or more elaborate Runge-Kutta methods are taken to calculate approximate solutions to the above system of differential equations [StBu83]. Discrete dynamical systems are described by a set of difference equations $\vec{x}_{n+1} = V(\vec{x}_n)$ with initial value $\vec{x}_0$.

Visualizing the behavior of dynamical systems is crucial for a deeper understanding of the underlying dynamics. Different techniques are already known in the field of fluid dynamics to visualize 2D and 3D vector fields [Hans93].

The most simple approach, the hedgehog method, displays the tangential directions of the flow field as vectors at some selected positions of phase space. The length of the vectors may encode flow velocity. Such plots give information about the vector field but do not show important structures, e.g., vortices, of the underlying dynamical system. Although quite feasible for 2D dynamical systems, the images for 3D dynamical systems tend to become quite intricate and difficult to analyze.

Another approach introduces particles into the dynamical system [Wijk92]. The flow behavior is then investigated indirectly by analyzing the movement of the particles, which trace out trajectories in the flow field or phase space. "Streamlines" describe the path of a single particle in an autonomous (time constant) vector field. "Particle paths" describe the path of a single particle in a time varying vector field. "Streaklines" visualize the path of a sequence of particles which are introduced into a time varying vector field at a fixed spatial position but regularly distributed in time.

Sometimes the local behavior of a dynamical system is more interesting at some specific locations then at others. Objects like tufts (e.g., groups of deformable ribbons) may be fixed at those locations. The deformation on these objects visualizes the underlying flow dynamics. [LeWi93] use glyphs to visualize local characteristics of vector fields. A glyph is a geometric object whose features, e.g., shape, length, volume, color, curvature, encode interesting local properties of the vector field.

Moving the Frenet frame along a trajectory allows the specification of ribbons to visualize curling and twisting behavior. More general objects than lines, e.g., polygons and circles, may be swept along trajectories.

Blob tracing introduces higher-dimensional deformable objects into the vector field without tying the object to a specific location. This is in analogy to injecting dye into fluids. [Gröl94] gives a more detailed account of applying visualization techniques to complex and chaotic dynamical systems.

Bifurcation diagrams illustrate the long term behavior of a whole class of systems. One parameter of the set of defining equations is taken as bifurcation parameter and is varied along the horizontal axis (driving parameter). On the vertical axis the long term behavior of one of the state variables is plotted. Varying the bifurcation parameter only slightly may produce vastly differing behavior (periodic or chaotic attractor). For details on bifurcation diagrams see for example [PeJü92].

In the following we describe an interactive program for analyzing the dynamical behavior of systems defined by a set of difference or differential equations. Emphasis has been put on offering the user a high degree of interactivity. He may specify and change one of the following components: dynamical model, geometric parameters, information to be displayed, color encoding.

2. Program Functionality

In the current version the program 'Attract' visualizes the long term behavior of dissipative, dynamical systems, such as attractors and bifurcations, in two and in three dimensional phase space. Additionally color definition and viewing interaction is possible. The main components of 'Attract' are described in the following sections and include: formula editing for the specification of a dynamical system, visualization of dynamical systems, and color editing.

2.1 Formula Editor

At the beginning of a working session the user has either the choice to manipulate an already existing dynamical system or he may want to create a new one. In the following chapter we will illustrate the steps that have to be taken to define a dynamical system. In a subsequent step this dynamical system is then visualized either as a 2D bifurcation diagram, a 2D attractor or a 3D attractor.

When creating a new dynamical system the user is presented a formula editor which allows him to define functions and other important data according to his need. The dialog consists mainly of two parts, the formula definition section and a dynamic section where various data values are entered (see figure 1).

The formula definition section provides space for five formulae to be visible simultaneously, which, in case of more than five formulae, can be scrolled using the scroll bar to the right. In addition to the mathematical expression, which will be explained subsequently, every formula contains various other data. As shown in figure 1 a formula can be chosen to be active or inactive. In case a formula is selected to be inactive it will not be considered further on, so as if it were not entered at all. This allows the user to temporarily remove portions from a dynamical system from consideration without having to reenter those portions again at a later stage.

Next, the user can choose on which axis the value of the function or expression will be plotted. Possible values are *X*, *Y*, *Z*, *color*, *parameter*, and *none*. Selecting *color* induces a color encoding of the function value. The specification of a color scale can be done interactively and is explained later on. If *parameter* is chosen the formula entry is displayed on the x-axis as driving parameter of a bifurcation diagram. If *none* is chosen the corresponding function value is not displayed at all. Finally, an initial value (such as x_0) and a value used in the numerical calculations can be defined. These values are optional, if undefined default values are used.

The lower section of the formula editing window allows the definition of various other parameters which are also necessary to display a dynamical system. The functionality of this section changes dynamically by switching a radio button accordingly. In position 'Konstante' (constants), constants can be edited, i.e., they are created, deleted, or their values are changed. These constants can be part of a formula expression. Next in position 'Farben' (colors), if a formula is assigned to the color axis, the mapping of a function value to the color scale can be defined by calling one of the two available color dialogs. Position 'Darstellung' (display) allows to adjust iteration parameters, i.e., the total number of iteration steps and the number of preliminary iteration steps during which no drawing is done. This is useful as the attractor of a dissipative, dynamical system is approximated by a trajectory, and the initial portion of such a trajectory may still be far away from the attractor itself. Finally position 'Bereich' (range) allows the user to choose the range of phase space he wants to have displayed. Error messages which might occur during formula parsing are displayed at the bottom of the window.

2.2 Types of formulae

The formula editor is used to enter the functions and mathematical expressions that make up a specific dynamical system. The formula parser handles various kinds of expressions (see table 1).

Type	Syntax	Example
difference equation	var=expr	x=x*k*(1-x)
differential equation	var'=expr	x'=a*y
expression	expr	b*sin(y)

Table 1: types of formulae handled by the formula parser

'expr' can be any mathematical expression with the usual operators, where additionally trigonometric and logarithmic functions, e.g., sin, cos, variables and constants can be evaluated.

For difference and differential equations an initial value (e.g., x_0) is specified. For differential equations the step size of the Euler integration, which is currently used for efficiency reasons, can be defined. For bifurcations a driving parameter has to be defined, including the iteration range. The driving parameter is automatically assigned to the horizontal axis.

Furthermore if an equation or expression is assigned to an axis, their values will be plotted on that axis, otherwise the value is just used in the computation but not displayed. If the assigned axis is the color axis then the color of the plotted point depends on the chosen color assignment.

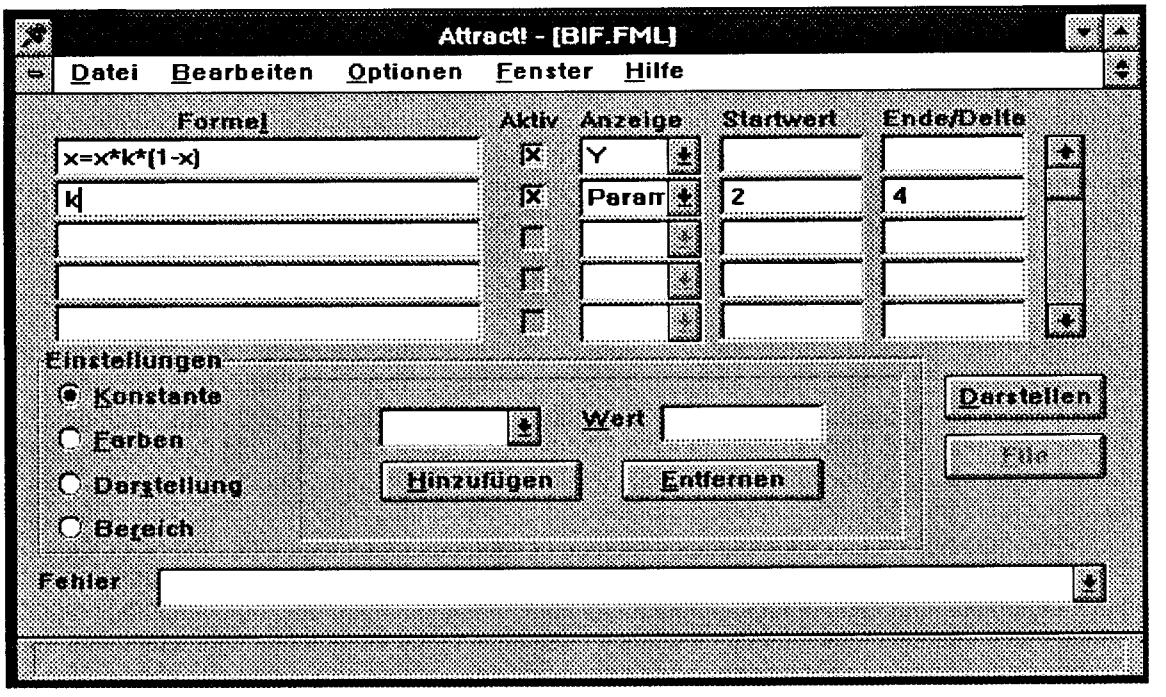

Figure 1: window for formula editing (bifurcation)

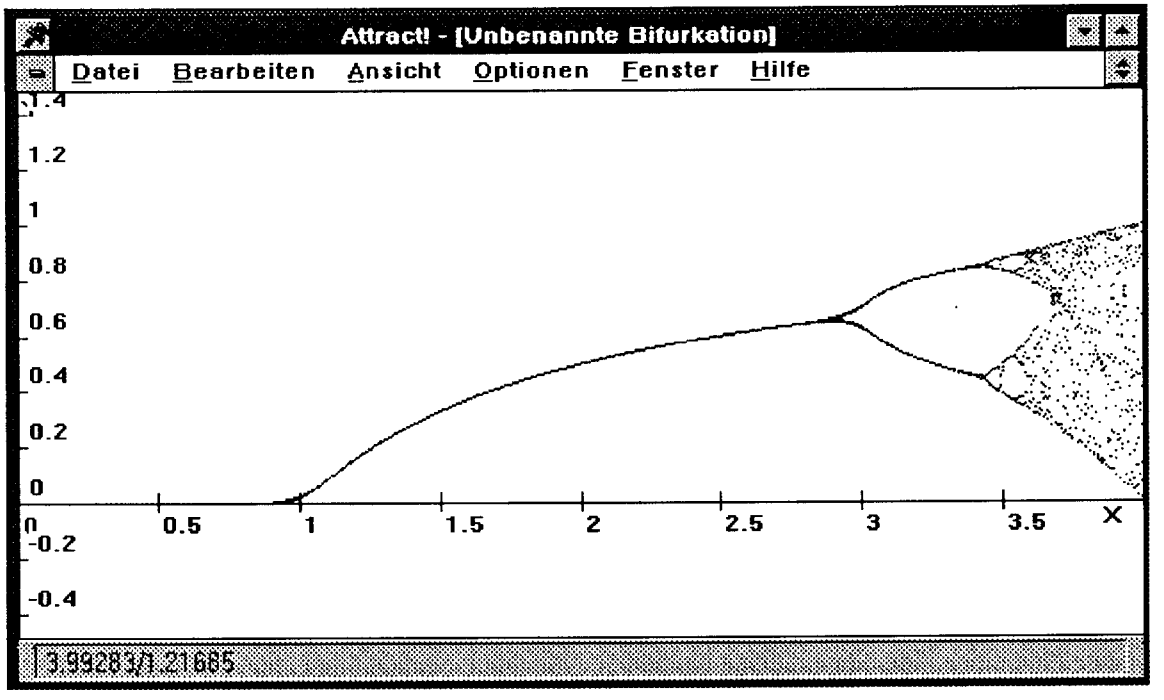

Figure 2: display window of a bifurcation diagram

2.3 Bifurcation diagram (example)

As our first example we want to display a bifurcation diagram which is due to the system $x_n=x_{n-1}*k*(1-x_{n-1})$. Thereby k is the driving parameter of the bifurcation and is varied along the

x-axis. Figure 1 shows the representation of this bifurcation in the formula editor. As indicated above, the choice *parameter* as axis is used only for bifurcations to determine the driving parameter. In this case the two numerical values in the same row as the driving parameter provide the initial and final values of this parameter.

This example also shows that the user does not have to bother with function or variable indexes. In a function definition the index of the assigned variable is always assumed to be *n* , and the indices on the right side of the assignment symbol are taken to be *n-1*. If components of a higher level of recursion are required they have to be defined explicitly as, e.g., x[n-2] to represent x_{n-2} or y[a-3] to represent y_{a-3}.

Pressing the button 'Darstellen' (display) opens the window to display the bifurcation diagram. (see figure 2)

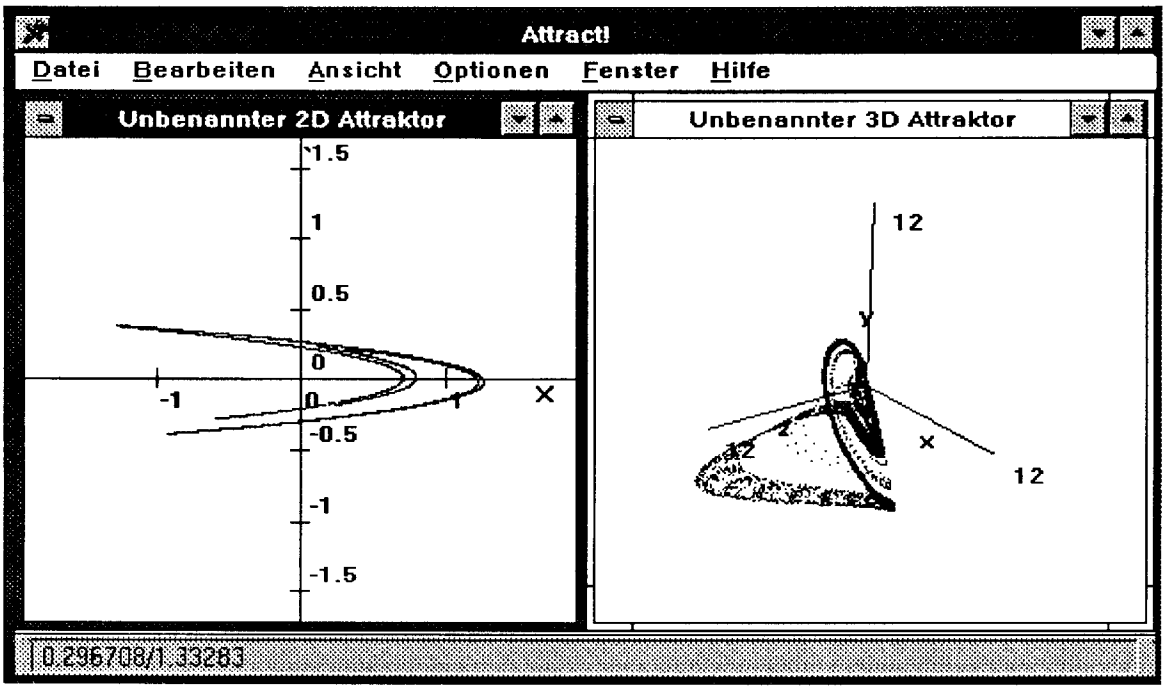

Figure 3: display window of 2D and 3D attractors

2.4 2D and 3D Attractor (example)

We take the following dynamical system for an example of a 2D attractor: $x_n = y_{n-1} + 1 - a * x_{n-1} * x_{n-1}$ and $y_n = x_{n-1} * b$ Constants a and b are chosen as a=1.4 and b=0.3. Figure 3 on the left shows the corresponding 2D attractor.

Figure 3 to the right illustrates a 3D attractor. The underlying dynamical system is given as: x'=-y-z, y'=x+0.4*y, and z'=2+z*(x-4). In this case a set of differential equations is used as opposed to the 2D example where the system was defined by a set of difference equations.

2.5 Color Definition

When defining a color axis, the user can choose between value ranges that are assigned colors and a continuous color axis. In both cases, the respective colors can be adjusted by scroll bars associated with the components of the HLS color model. In the HLS model a given color is defined by three values, hue, lightness, and saturation. Hue is measured in degrees, where 0 is blue, 90 magenta, 180 yellow, and 270 green, lightness gives the position between white and black, and saturation gives the amount of color, where 0 is gray and 1 is full color. For user feedback the color spectrum with medium lightness and full saturation is shown, as well as the color resulting from the current settings. The color dialogs also allow file operations such as saving and loading defined color schemes.

In the range color dialog (see figure 4), the value ranges can be defined by the upper boundaries, either in absolute values or in percentages of the overall value range of the formula or function. For each value range, a color can be defined in the described manner.

In the continuous color dialog (see figure 5), two colors and the number of intermediate colors are defined. For each function value the corresponding color is determined by interpolation of the two initial colors.

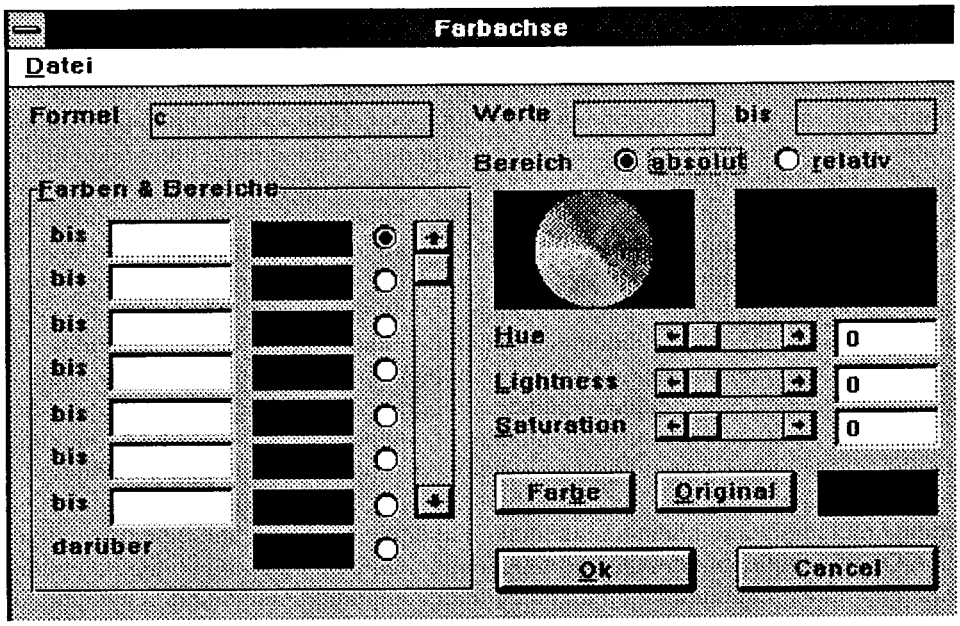

Figure 4: range color dialog

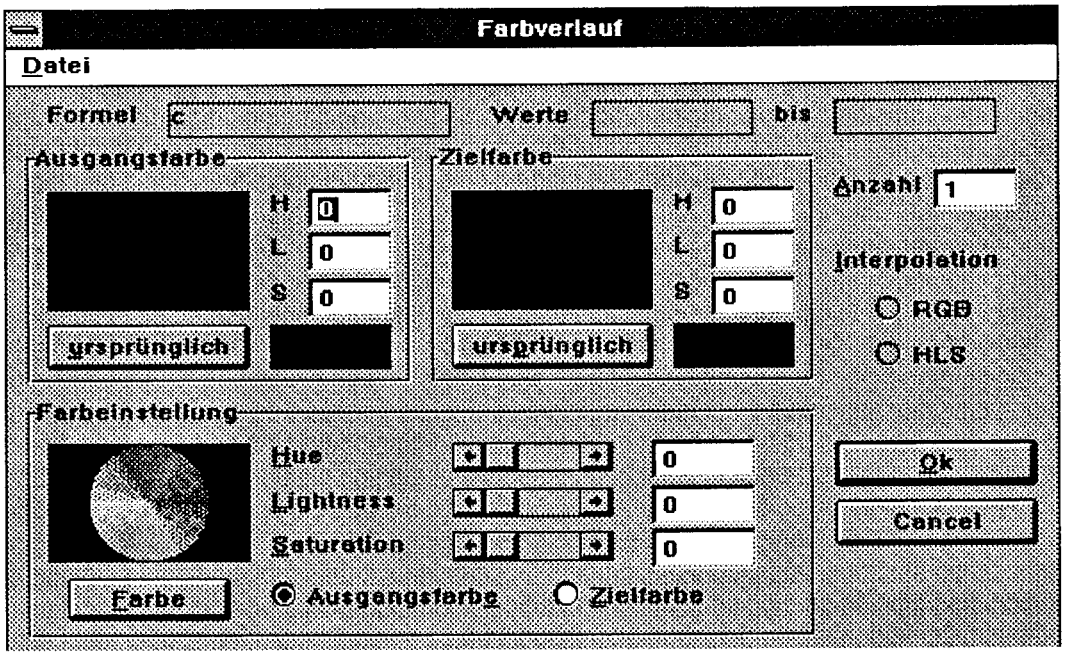

Figure 5: continuous color dialog

2.6 Visualization of Dynamical Systems

As described in the introduction there are various ways to visualize dynamical systems. In the current program only the most basic type is implemented, i.e., only point sets and polylines are drawn. This choice was taken due to the graphics and computational restrictions of the target hardware platform. The point sets and trajectories are colored according to the chosen color assignment.

Currently, there are three graphic views in use: 2D attractors, 2D bifurcations, and 3D attractors. The difference between attractor and bifurcation views is that the computation of attractors starts at an initial point and performs a given number of iterations, whereas the

computation of bifurcations iterates the bifurcation parameter in an outer loop, and for every value of this parameter iterates the remaining formulae.

In the 2D views the user may zoom-in and zoom-out to adjust the visible range. Zooming in is accomplished by drawing a rectangle, i.e., pointing to one corner, keeping the mouse button pressed and releasing the mouse button at the opposite corner. Zooming out is done by either undoing the zoom-in operations, or by doubling the visible range. By resizing the window a higher or lower resolution for the same range can by achieved, thus changing the applied scale. Furthermore by moving the mouse across a 2D view, the coordinates of the current mouse cursor position are automatically shown in the status bar.

3D views allow zoom-in and zoom-out operations similar to the method described for the 2D case. The repositioning of the origin of the coordinate system may be done as well. This is simply achieved by switching to the 'move' mode and dragging the origin to the desired position. Finally, we implemented an operation to rotate the coordinate system using a hemisphere metaphor: A circle centered at the origin of the coordinate system is shown to represent a hemisphere. A point within the circle unambiguously identifies a point on the hemisphere. Therefore by clicking on a point within the circle and dragging it, we can calculate the respective point on the hemisphere, simulate the rotation of a sphere, and change the settings of the camera position accordingly.

2.7 Configuration and program setup

Configuration of the system is done interactively as well. The program allows the specification of all default values as well as of some display options (see figure 6).

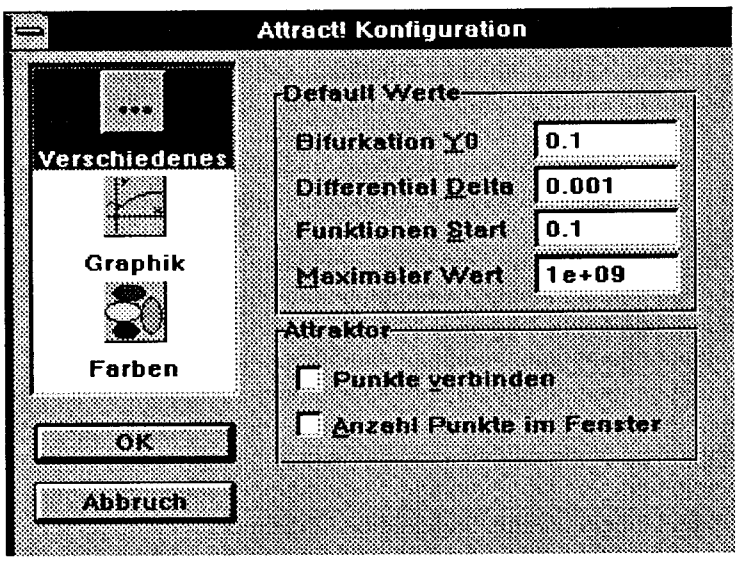

Figure 6: Configuration of default values

The default values are: initial value for equations assigned to the y-axis in bifurcation diagrams, step size for the Euler integration method, initial value for each equation (e.g., x_0), and maximum value allowed for computations. A checkbox in the lower half of the window determines if an attractor is approximated by a point set or polyline. With another checkbox the user specifies the termination of calculation: processing stops either after a certain number of points have been calculated or after a certain number of points visible in the current window have been calculated. With the second option the density of the displayed point set will not change after, e.g., zoom-in operations, although the computation time will increase.

The graphics sub dialog (see figure 7) defines whether display update after exposure or resizing will occur always, only when the window is active, or on command. The lower part defines the display mode of axes, i.e., whether to show them at all, and whether to display legends and units along the axes.

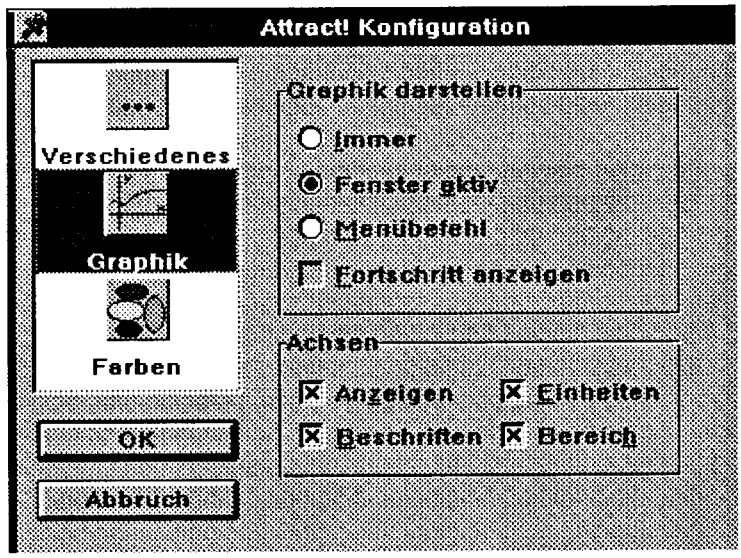

Figure 7: display options

3. Implementation

The current version of 'Attract' was implemented with Borland's C++ 4.0 compiler on Windows 3.1. We chose this environment because it was the one the prospective users had at hand, and we could build upon the object library and tools provided with the development system. These considerations had priority over handicaps such as computing speed.

The central data structures used are the formula and the so-called formula system. A formula system consists of a set of formulae, a set of constants, which can be referred to within a formula, and a set of variables which stores the current and, if needed, previous values of the functions and expressions. In the current implementation the attributes for a formula are as shown in table 2.

Attribute	*Description*	*Example*
Expression	expression to be evaluated	y=sin(x)
Active	use formula in current system	yes/no
Axis	plot value on which axis	x, y, z, color, none
Initial value		0
step size	used for Euler integration	.001
Limit	upper limit of bifurcation parameters	4
Expression tree	built by parser for faster evaluation	
Color description	color axis information	

Table 2: formula attributes

When opening a display view the whole formula system is checked for consistency errors, such as undeclared constants or functions. After parsing the expression the formula contains an expression tree, which is evaluated for efficiency reasons, with references to storage objects of variables and constants. In addition, each formula can be assigned a color axis information. If errors have occurred they are displayed, otherwise the editor creates a copy of the formula system, opens the according display view and passes the copy. The user now has the choice of either editing the original formula system and opening new views, or editing the formula system belonging to the respective view (see figure 8).

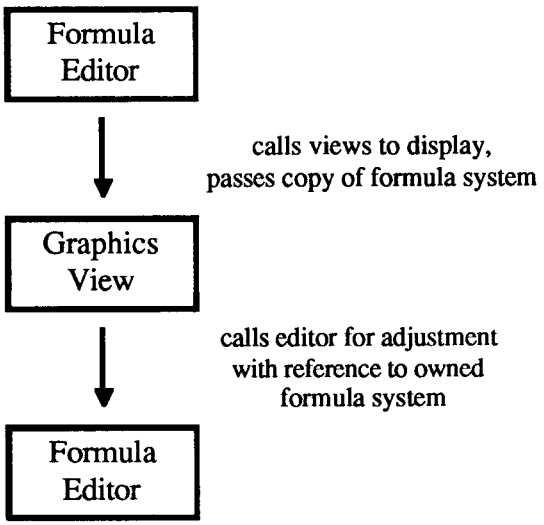

Figure 8: module interaction

4. Conclusion, future work

In this paper we present 'Attract', a program that visualizes various dynamical systems. In our prototype, we concentrated on interactivity questions. Future versions of the program will include more elaborate calculation, visualization and interaction techniques. We will especially concentrate on the implementation and definition of more complex visualization methods.

References

[AbSh84] Abraham, R.H., Shaw, Ch.D., "Dynamics - The Geometry of Behavior", Vol. 1–4, Aerial Press, 1984.

[BrCa92] Brodlie, K.W., Carpenter, L.A., et. al., "Scientific Visualization - Techniques and Applications", Springer, 1992.

[Chao89] "Chaos und Fraktale", Spektrum der Wissenschaft, 1989.

[Deva89] Devaney, R.L., "An Introduction to Chaotic Dynamical Systems", Addison-Wesley, 1989.

[Gröl94] Gröller, E., "Application of Visualization Techniques to Complex and Chaotic Dynamical Systems", Proceedings of 5th EUROGRAPHICS Workshop on Visualization in Scientific Computing, 30. May - 1. June 1994, Rostock, Germany.

[Hans93] Hansen, Ch., "Visualization of Vector fields (2D and 3D)", SIGGRAPH 93, Course notes no 2., Introduction to Scientific Visualization Tools and Techniques, 1993.

[LeWi93] Leeuw, W.C., Wijk, J.J., "A Probe for Local Flow Field Visualization", IEEE Computer Society Press, Proceedings of Visualization '93, 1993.

[NiSh89] Nielson, G.M., Shriver, B., Rosenblum, L.J., "Visualization in Scientific Computing", IEEE Computer Society Press Tutorial, 1989.

[PeJü92] Peitgen, H.-O., Jürgens, H., Saupe, D., "Chaos and Fractals - New Frontiers in Science", Springer Verlag, 1992.

[StBu83] Stoer, J., Bulirsch, R.: Introduction to Numerical Analysis (2nd ed.), Springer, 1983.

[Tson92] Tsonis, A.A., "Chaos - From theory to applications", Plenum Press, 1992.

[Wijk92] Wijk, J.J., "Rendering Surface Particles", IEEE Computer Society Press, Proceedings of Visualization '92, 1992.

Acknowledgment

The program system was developed due to suggestions of researchers at the institute of econometrics, operations research and system theory of the Technical University Vienna. The authors would like to thank the following persons for fruitful discussions: Prof. G. Feichtinger, A. Milik, A. Prskawetz.