brains, minds media
ISSN: 1861-1680

tkCybernetics

Thorsten Roggendorf

Department of Biological Cybernetics, Bielefeld University, Universitaetsstraße 25, 33615 Bielefeld, Germany

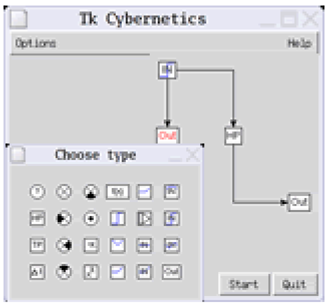
thorsten.roggendorf@uni-bielefeld.de

Abstract. tkCybernetics is a simulation software that can be used to create and evaluate simple cybernetic circuits as a combination of lowpass and highpass filters and several nonlinear elements. tkCybernetics has been developed for and successfully applied to introductory courses on biological cybernetics at Bielefeld University. Therefore, it is recommended for students who start learning about cybernetics. Students can quickly grasp the concepts used in the graphical user interface and are able to create their own circuits after a short introduction to the program. Its usefulness for advanced students and researchers is limited to quickly testing circuits. This article provides detailed tutorial information on how to install and work with tkCybernetics for self-learning and for higher education.

Keywords: cybernetics, system theory, circuits, software tool, simulation, lab course, tutorial

Citation: Roggendorf T (2006). tkCybernetics. Brains, Minds, and Media, Vol.2, bmm327 (urn:nbn:de:0009-3-3275).

Supplementary Material



Article Resources

Additional Resources

Datasheet

Related Publication

Neural Networks as Cybernetic Systems, Part I (Cruse 2006a)

1 Introduction

tkCybernetics is a simulation software that can be used to create and evaluate simple cybernetic circuits as a combination of lowpass- and highpass filters and several nonlinear elements. This stand alone application provides a graphical user interface that allows beginners (concerning computers as well as cybernetic systems) and advanced users alike to create simple cybernetic circuits. The circuits can be tested with various input functions. The output can be sampled from any element of the circuit and is visualized onscreen as a line plot. Due to its implementation in the TCL language, the program will run on almost any Unix and Linux variant, on Macintosh and even on Windows.

tkCybernetics was specifically developed for a lab course in biological cybernetics and has been successfully used in that course for several years. Students do in general quickly grasp the concepts used in the graphical user interface and are able to create their own circuits after a short introduction to the program. However, experience shows that it is advisable to have an experienced user available whom students can ask questions that will almost inevitably arise.

The program is also useful for other applications in the larger field of cybernetics. It was for example also used by various researchers to quickly test simple cybernetic circuits.

It is possible but not advisable to use tkCybernetics for the implementation of large cybernetic systems. The program is therefore recommended for students who start learning about cybernetics. Its usefulness for advanced students and researchers is limited to quickly testing circuits.

2 Installing and starting the program

2.1 Requirements

You need a Tcl/Tk interpreter to run this program. This interpreter is available for many platforms including Unix and Windows. Get it at http://www.tcl.tk, download it and follow the installation instructions. If you use a Linux distribution it very likely includes the interpreter. Since this is a graphical application, what you want is not just the normal Tcl interpreter, but the tk extension to Tcl. The normal Tcl shell is called tclsh, what you want is the tk shell called wish.

2.2 Installation

You need the files *language_module.tcl* and *tkCybernetics.tcl*. The latter is the program, the former is needed by the program – it contains all texts that appear in the program.

If you use Windows, the Tcl Installer will hopefully have registered all files ending on **.tcl* to be executed by the interpreter. If not, you have to do that manually. Put the two files in the same directory where ever you want and double click tkCybernetics.tcl.

Linux/Unix users will have to make the file executable. Change into the directory, where the file lies and type `chmod +x tkCybernetics.tcl`. Now you can start the program by typing `tkCybernetics.tcl`. If that does not work, you must edit the first line of the file to point out the fully qualified path of where your wish (the tk shell) lies. Don't erase the leading two characters: `#!`

2.3 Other files

If you got tkCybernetics in the official tarball, you should have found it to include some more files:

- **english.tcl~german.tcl:** These are the language modules. *language_module.tcl* is the one in use. If you want to change the language, replace *language_module.tcl* by the module of your choice (e.g. rename *english.tcl* to *language_module.tcl* - or copy it to that name).
- **bmm_tkC_english.pdf:** This article in pdf-format. Windows users can use an Acrobat^{TM} reader to read it. Linux users can use xpdf or gpdf.
- **tkC-icon.xpm:** This graphic file can be used as an icon for tkCybernetics if you want to create a clickable starter on the desktop environment of your choice.
- **GPL:** The gnu general public license - the license terms for all files of this package including this file. Read it if you don't know it already. Usage of any kind of any of the files included with this package implies acceptance of the terms stated in the GPL!

3 Manual

The following tutorial explains how to create circuits. tkCybernetic's design recognizes the two major populations of computer users: the mouse (wo)men and the key (wo)men. You have to decide what type you are and should then follow the according instructions. Note that learning the keyboard shortcuts takes a bit longer than using the pure mouse interface. However, if you are planning to create more than a handful of circuits you'll be much faster with the keyboard.

Task	**Key**	**Left mouse button**
create element	c	default
choose element type	space	X
parametrize element	p	X
remove element/connection	del	X
Connect	right mouse[1]	
flip connection	space	X
move elements	middle mouse dragging	
debug[2] element	i	
debug all	I	

Table 1: How to achieve common tasks. Hover the mouse arrow over the according element or connection and press the according key. Alternatively the left mouse button can be reconfigured to achieve most of the tasks.

[1] Right click and hold on origin, move to target and release.
[2] Debugging output is send to `stdout`. To see it you have to start tkcybernetics from a shell.

If you are a mouse (wo)man ignore the keyboard shortcuts, otherwise you can ignore all instructions about the mouse configuration window (Fig. 2). Just hover the mouse over the according element or connection and press the according key (Tab. 1)).

The following sub section is written in the spirit of a step by step tutorial. You do not have to follow the instructions verbatim if you don't feel like it. But if you do, the state of your tkCybernetics will resemble the figures in the tutorial and hopefully give you some confidence that you got it right.

3.1 Tutorial

Start the program. You'll see a scarcely populated menu bar at the top, a grey area in the middle (this is where you create circuits) and two buttons at the bottom (Fig. 1).

A circuit is generated by creating elements and connecting them with arrows. Arrows reflect the direction of information flow, while predefined calculations are performed in the elements. The number of possible/required inputs to an element depend on its type.

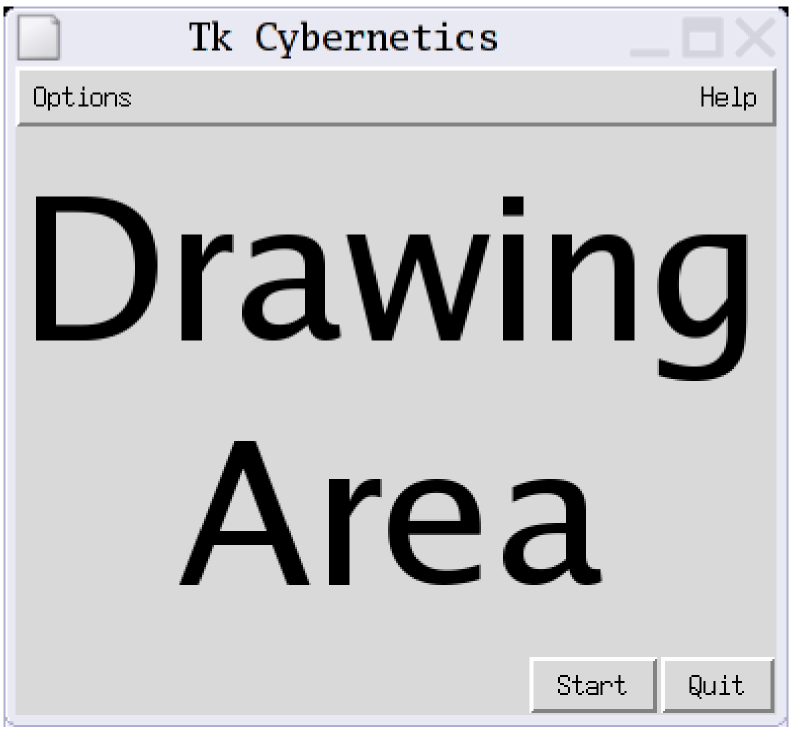


Figure 1: tkCybernetics main window. The text “drawing area” was added for clarification and is not part of the program.

Click on the *options* menu and choose *mouse*. A small window will pop up that lets you determine the function of the left mouse key (Fig. 2). Move that window to a convenient position beside the grey drawing area. Since you just started tkCybernetics, the active selection in the mouse window will be *Create element*. If you click with the left mouse button anywhere on the drawing area an element will be created at the cursor position. If you want to create more elements later after you used the left mouse button for other purposes you'll have to first select *Create element* in the mouse configuration window.

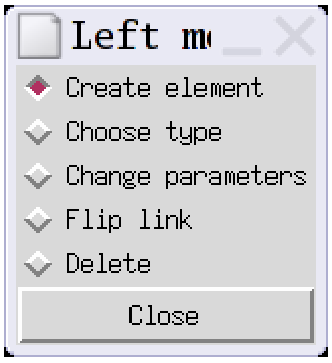


Figure 2: Mouse configuration window.

Freshly created elements are no more than placeholders. You have to tell them what kind of cybernetic element they are supposed to simulate. To do this choose *Choose type* in the mouse configuration window and then left click that element for which you want to assign a new type. Another window will pop up that lets you choose the type (Fig. 3).

When you hover the cursor over the elements a mouse over text will tell you what kind of element that is. Only the names are given. Covering the theory of cybernetic systems is out of the scope of this manual. You'll have to consult other sources to learn what such elements do. A good introduction can be found in Holk Cruse's freely available book “Neural Networks as Cybernetic Systems” (Cruse 2006a, 2006b). Please visit http://www.brains-minds-media.org to learn more about this.

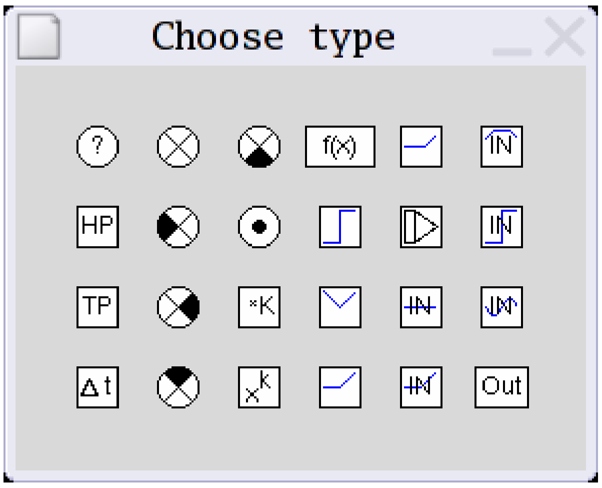


Figure 3: Window for choosing element types.

Depending on which type you chose you might want to set the parameters of the element. Some elements like sums do not require this step. But lets assume you chose a high pass filter, the first element of the second row in the type choosing window. Then you'll have to set the time constant of the filter. After choosing a type that requires additional parameters (like the high pass filter) a window will automatically pop up that lets you enter the required parameters (Fig. 4).

Just enter the desired value there (choose 10 for now). If you want to change the parameters later, you will have to choose *Change parameters* in the mouse configuration window and then left click the according element. The same dialog will pop up retaining your first parameter choice and allowing you to change the parameters as desired.

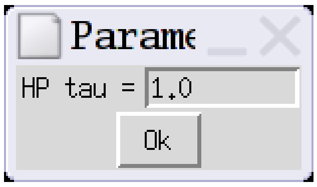


Figure 4: Window for setting the parameters of a high pass filter.

Lets create three more elements now, two *outputs* (leave default parameters for the first and change the second to Color = red, Y-offset = 100) and an input element *step-in*. For the latter set parameter *Start(t)* (i.e. the step onset) to 50 and *Stop(t)* (i.e. the step end) to 100. Set *Amplitude 1* (the value before and after the step) to 0 and *Amplitude 2* (the value during the step) to 50. You can edit the parameters faster by using the tab key after entering a value to jump to the next parameter and pressing return when you are finished. If you hover the mouse over the elements a mouse over text will display the current parameters.

Now you should have four elements. But to do anything interesting with them, you will have to connect them. Hover the mouse pointer over the *Step-in*, press and hold the right mouse button, move to the *high pass filter* and release the right mouse button. That should create an arrow from the *Step-in* to the *high pass filter*. Create another arrow from the *input* to the *red output* and one from the *high pass filter* to the *unchanged output*.

Congratulations! You have just created your first circuit with tkCybernetics. If you have followed this tutorial it should look somewhat like Figure 5.

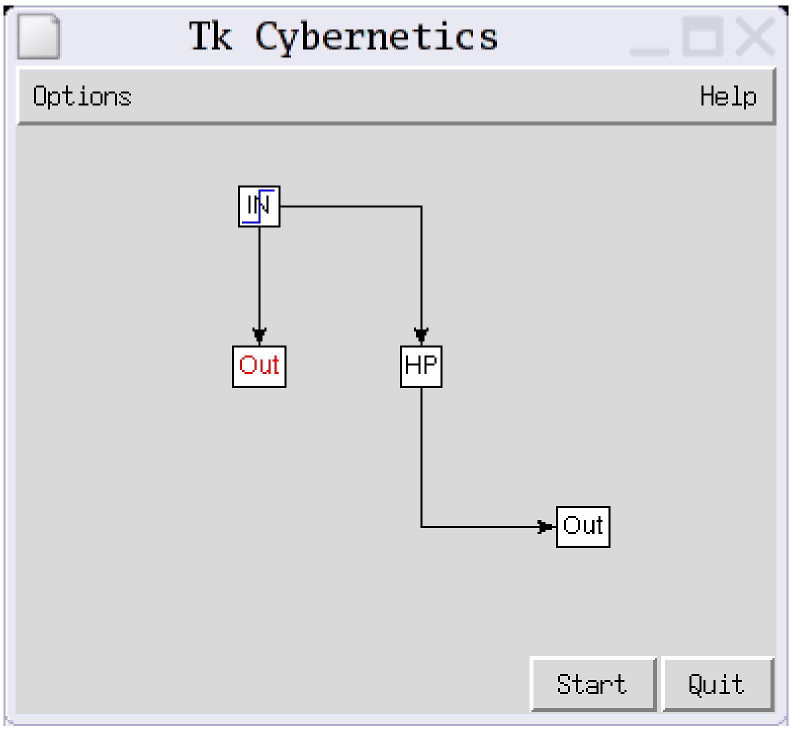


Figure 5: A simple circuit created in tkCybernetics

If you want to have your circuit look exactly like Figure 5 you can use the middle mouse button (press, hold, and drag) to move your elements to the according positions. Still the connection from the high pass filter to the output will look different in your circuit. Choose *Flip link* in the mouse configuration window and click that connection to flip it to the other position. You will need to move elements and flip links when you are creating more complex circuits. Note that elements can have only one input from one direction. Otherwise printouts of the circuits could be ambiguous, negatively impeding students' protocols.

Time to test your shiny new circuit. Press the *Start* button. Another window will pop up that shows the values as they arrive in the output element. For a qualitative impression of what the circuit does, this is perfect. If you want to make more precise measurements though, you should activate the coordinate system: From the menu choose *Options/Output/Draw coordinate system* and press the *Start* button again.

If you followed this tutorial, your results should look like those in Figure 6. The circuit created in this tutorial can be used to explore the available elements in tkCybernetics. Just change the type or the parameters of the high pass filter element and see how elements respond to the input step function. Step functions are very good for getting an impression of what cybernetic systems do.

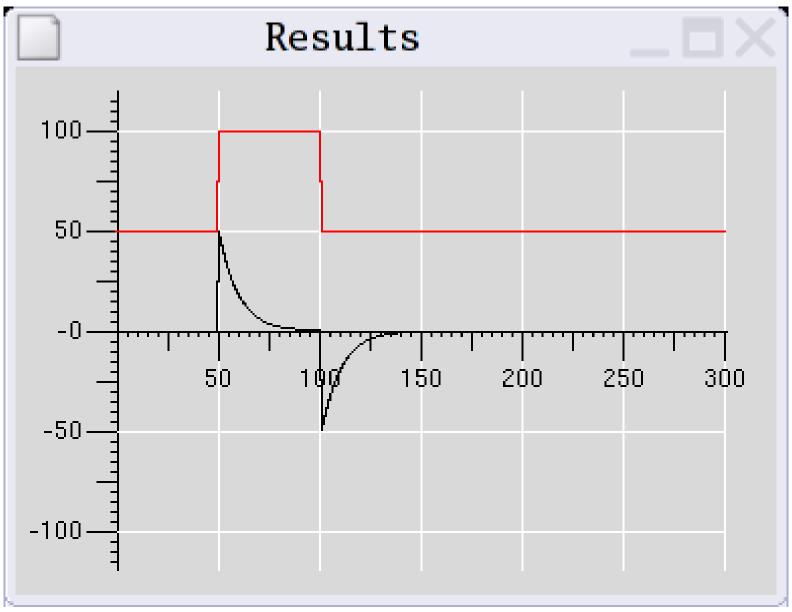


Figure 6: Output of the circuit in Figure 5.

3.2 More options and functions

The number of simulated time steps depends on the width of the result window. You will get one time step per pixel (minus a small margin of the plot). Thus you have to resize the results window to change the number of simulation steps.

Error messages often give the position of the faulty element. To use this information the cursor position can be displayed by clicking *Options/Cursor position*. (Usually you will not need this because faulty elements are also highlighted. Elements may be hidden under other elements though).

The diagram can be saved as postscript (*Options/Save/Diagram*). By activating the check button, the parameters of the elements will also be saved (this may not look nice, because parameters can overlap if elements are close to each other). The results can also be saved as postscript (*Options/Save/Results*). Note that circuits cannot be loaded, what you save is only a graphic representation of your circuit!

The time step of the simulation can be adjusted through *Options/Output/Time Step*. Raising that value will make the simulation rather imprecise but can be useful for getting a rough impression of what a complex circuit does.

The coordinate system in the results window can be configured through *Options/Output/Options coordinate system*. The pop up dialog lets you set tick sizes and distances and determine the grid color.

3.3 Element types

The following elements are available. *x* is always the input value, if more than one input is possible, inputs are indexed (x_i). *t* is current time step (i.e. iteration number). All other *emphasized* words refer to parameters that can be set through the parameter dialog (e.g. Figure 4):

? **no type** assigned

HP **temporal highpass filter**

TP **temporal lowpass filter**

Δt **time delay** $x_{t\text{-}delay}$

⊗ **sum** $\sum_0^i x_i$

⊗, ⊗, ⊗, ⊗ **difference**, like sum, but inputs from black sectors are subtracted. Output must leave from white sector

⊙ **product** $\prod_0^i x_i$

*K **scale** K_x

power x^k

function *f(x)* **–** scriptable *function* of single input (see section 3.4 or online help)

relay $\begin{cases} amplitude1 & \text{if } x<threshold \\ amplitude2 & \text{else} \end{cases}$

absolute $|x|$

threshold $\begin{cases} 0 & \text{if } x<0 \\ 1 & \text{else} \end{cases}$

variable threshold $\begin{cases} 0 & \text{if } x<threshold \\ x-threshold & \text{else} \end{cases}$

integrator $\int_0^t x_t$ May have one or two inputs. If it has only one, this may come from an arbitrary direction. If it has two, one must come from the direction of the funnel opening (left). This input will be integrated. The other input may come from an arbitrary direction (not left). If the latter input has a nonzero value it will reset the integrator (*empty it*).

constant-in *amplitude*

ramp-in $\begin{cases} 0 & \text{if } t<start \\ t-start & \text{else} \end{cases}$

ramp-and-hold-in $\begin{cases} amplitude1 & \text{if } t < start \\ \max(amplitude2, \{(t-start)slope + amplitude1\}) & \text{if } start < t < stop \\ \min(amplitude1, \{amplitude2 - (t-stop)slope\}) & \text{else} \end{cases}$

step-in $\begin{cases} amplitude2 & \text{if } start<t<stop \\ amplitude1 & \text{else} \end{cases}$

sine-in $amplitude \sin\left(\frac{2\pi t}{period}\right)$

out The single input value will be visualized over time in the result plot. Out-elements may have any color supported by your system (usually at least the usual English names like red, green ...). Y-offset will move the output up or down in the result plot as according to the given value.

3.4 Functions

Elements of type function (represented as *f(x)* in Fig. 3) expect a function as their parameter. This function must follow a certain syntax. First the available built in functions are listed, then the syntax is explained.

3.4.1 Available built in functions

The following math functions are available:

abs(w) absolute

sin(w), cos(w), tan(w), asin(w), acos(w), atan(w), atan2(w,v), cosh(w), sinh(w) trigonometric functions

ceil(w), floor(w) well, guess ;)

round(w) ...

fmod(w,v) remainder of division *w/v*

pow(w,v) *w* to the power of *v*

sqrt(w) pow(*w*,0.5) -- square root of *w*

log(w), log10(w) base *e* and base *ten* logarithm

hypot(w,v) sqrt(pow(w,2)+pow(v,2))

exp(w) *e* to the power of *w*

3.4.2 Syntax

w and *v* can be arbitrary fixed values or *x*, the input of the unit. Functions and values can by linked by the usual mathematical operators (+ - * /). The brackets have to be set correctly (take special care of complex expressions). Only round brackets *( )* are valid.

3.4.3 Example

Suppose you want to have the function

$$\frac{\sqrt{a^2+b^2}}{i}+j$$

in a function element, where *a* is the input value of the element and the other values are constants. Then the following line could be the parameter of your function module:

(sqrt (x * x + pow(7.9,2)) / 5.1) + 3.7

4 Copyright

5. References

Cruse, Holk (2006a). Neural Networks as Cybernetic Systems - Part I (2nd and revised edition). Brains, Minds and Media, Vol. 2, bmm289 (urn:nbn:de:0009-3-2891).

Cruse, Holk (2006b). Neural Networks as Cybernetic Systems - Part II (2nd and revised edition). Brains, Minds and Media, Vol. 2, bmm290 (urn:nbn:de:0009-3-2906).

GPL Homepage – http://www.gnu.org/copyleft/gpl.html

tcl/tk Homepage – http://www.tcl.tk.

tkCybernetics Homepage – http://tkCybernetics.sourceforge.net/