Journal of Neuroscience Methods 175 (2008) 104–107

Contents lists available at ScienceDirect

Journal of Neuroscience Methods

journal homepage: www.elsevier.com/locate/jneumeth

The Biopsychology-Toolbox: A free, open-source Matlab-toolbox for the control of behavioral experiments

Jonas Rose[a,*], Tobias Otto[a], Lars Dittrich[a,b]

[a] Institute of Cognitive Neuroscience, Biopsychology, Department of Psychology, Ruhr-University Bochum, GAFO 05/618, D-44780 Bochum, Germany
[b] International Graduate School of Neuroscience, Ruhr-University Bochum, FNO 01/114, D-44801 Bochum, Germany

ARTICLE INFO

Article history:
Received 17 June 2008
Received in revised form 5 August 2008
Accepted 6 August 2008

Keywords:
Behavior
Matlab
Operant conditioning
Programming

ABSTRACT

The Biopsychology-Toolbox is a free, open-source Matlab-toolbox for the control of behavioral experiments. The major aim of the project was to provide a set of basic tools that allow programming novices to control basic hardware used for behavioral experimentation without limiting the power and flexibility of the underlying programming language. The modular design of the toolbox allows portation of parts as well as entire paradigms between different types of hardware. In addition to the toolbox, this project offers a platform for the exchange of functions, hardware solutions and complete behavioral paradigms.

1. Introduction

Biopsychology is the application of principles of psychology to the study of mental processes. In order to conduct such studies it is of crucial importance to have powerful, flexible and easy-to-use tools to control the experimental environment of subjects. One of the most abundant and significant of such experimental environments is the operant conditioning chamber (Skinner box), a simple box in which animals can be stimulated visually, auditory and electrically, rewarded or punished (Skinner, 1956). While originally controlled manually, the development of the SKED system (Dingler et al., 1976) initiated the use of computer-controlled Skinner boxes. This automation boosted not only work-efficiency but also the available complexity of stimuli, behavioral protocols and data-analysis. In addition to the traditional use of Skinner boxes in purely behavioral experimentation, the range of experiments increased over the last decades and placed new demands on experimental hardware. Finding a reliable, easy-to-use and flexible solution for the control of Skinner boxes in particular and for the control of biopsychological experiments in general is therefore of utmost importance to every lab in this field of research. Unfortunately, to date there is no widely accepted standard for data-acquisition hardware and software.

Matlab is a programming language for technical computing, originally aimed at applied mathematics. In the past years the applications for Matlab have grown enormously and along came numerous commercially available Matlab-toolboxes that alleviate the analysis of experimental data (the Statistics and Curve Fitting Toolboxes to name a few). For this reason, Matlab has become the analysis software of choice for most labs, especially in computationally extensive fields of neuroscience, electrophysiology for instance. This wide use of Matlab in turn inspired several community-based toolboxes (Genovesio and Mitz, 2007; Meyer and Constantinidis, 2005; Brainard, 1997) for the analysis of experimental data. While Matlab has become the gold-standard for data-analysis, the picture is less clear when it comes to data-acquisition. Most laboratories use self-built setups, where the control relies on custom software, programmed by each researcher individually. Commercial systems are also available; however they provide a pricey and often inflexible solution with very limited compatibility. The vivid development of tools for the analysis of experimental data is pitted against a very small number of tools available for the control of experiments. The Biopsychology-Toolbox aims to fill this gap.

Here we introduce a Matlab-toolbox that allows a direct approach to the control of Skinner boxes in particular, but that is not bound to the use with Skinner boxes. In fact, one of the major goals of this project was to provide a set of tools capable of driving different types of hardware. The toolbox is designed in a modular fashion and is therefore highly flexible in terms of portability, extendibility and supported hardware. In addition to the Matlab control-software, this project offers suggestions of the

* Corresponding author. Tel.: +49 234 32 26845; fax: +49 324 32 14377.
E-mail address: Jonas.Rose@Ruhr-Uni-Bochum.de (J. Rose).

0165-0270/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.jneumeth.2008.08.006

basic hardware required to set up biopsychological experiments, particularly we describe several interfaces and a complete setup with Skinner box, touchscreen and interface for electrophysiology. All hardware components are modular and can be combined and replaced according to the requirements of a given experiment.

The Biopsychology-Toolbox is a free, open-source Matlab-toolbox that is aimed at the experienced programmer as well as the novice. The project as it is presented here reflects the current state of the toolbox, different implementations of software and hardware that are thoroughly tested and have successfully been used in our laboratory. However, we like to see the toolbox as work in progress. We will continue to expand it and hope for rich participation of the biopsychology-community. To grant full communal use and flexibility, all code, documentation of code and documentation of hardware are published under a creative commons license (CC GNU GPL by attribution, http://creativecommons.org/licenses/GPL/2.0/). This license allows free use, modification and redistribution of the Biopsychology-Toolbox or documentation thereof as long as the authors are cited (cite this article) wherever the toolbox or documentation thereof is used or results that were generated in benefit of the toolbox are published.

In the following we will give an overview of the general idea of the Biopsychology-Toolbox, its organization, features and limitations. The toolbox provides a set of basic functions that enable easy implementation of behavioral paradigms, additionally a set of portable and fully functional paradigms are offered for download. As stated above, one major aim of the project was developing a toolbox suitable for a broad range of experimental hardware. Therefore we will introduce different examples of interfaces and experimental setups that can be controlled by the toolbox. Presently, the toolbox supports different USB-based interfaces, a parallel-port based interface, a PCI-card and a touchscreen. These interfaces are fully supported by the toolbox, however, the scope of the project is by no means limited to these interfaces, and any other interface with Matlab-support can be used along the toolbox. Mathworks, for example, offers a toolbox for hardware-interaction (Data Acquisition Toolbox, see http://www.mathworks.com/products/daq; for a list of supported hardware, see http://www.mathworks.com/products/daq/description2.html) and most third-party companies offer Matlab-drivers for their hardware interfaces.

2. Materials and methods

2.1. Idea and outline of the project

In order to include the contradictory goal of easy-to-use functions with maximal flexibility, we chose to provide a set of very basic but powerful functions that solve many of the recurring problems in experimental control and can easily be extended or replaced. We deliberately decided against the use of most graphical user interfaces to avoid sacrificing power and flexibility to graphical appeal, instead we chose to put great efforts on documentation; numerous example programs and code-snippets are provided documenting every function of the toolbox. The toolbox was initially developed as a loosely connected set of functions which we bundled up into a complete toolbox to initialize a community-based project.

Along the toolbox we maintain an Internet site for help, download, communal discussions and the upload of extensions or modifications (homepage of the project: http://biopsytoolbox.sourceforge.net). A community-based project has many advantages over individual solutions. First, reliability and quality of code is improved by sharing and testing through different users. Second, a community-based platform provides maximal independence from commercial offers; software-licenses, updates and extensions can be obtained free of charge. Furthermore, extensions that are not yet available can easily be implemented since the entire project is open-source. Third, a large number of contributing labs help to widen scope and functionality of the toolbox.

In order to provide easy cost-control we designed the Biopsychology-Toolbox as stand-alone, requiring the Matlab programming environment but no additional toolboxes. However, all built-in Matlab functions can be used and additional toolboxes can be incorporated in extensions of the Biopsychology-Toolbox. At the present state, the toolbox provides functions for initialization, randomization, visual and auditory stimulation, control of external hardware such as pecking-keys, feeders or light-barriers as well as basic visualization and analysis of experimental results. Additionally, it provides software- and hardware-solutions for the use in electrophysiology; this includes the transmission of behavioral event-codes to the recording setup. Currently, we are using the toolbox with head-fixated animals and in Skinner boxes for behavioral, pharmacological and single-cell recording studies.

2.2. Organization and software-components

A small set of functions lie at the core of the toolbox, these functions allow for initialization, definition of a specific setup and the interaction with a given interface. Functions can easily be ported between experimental setups; only the function *mySetup* is bound to specific equipment since it defines all hardware-specific parameters. When *mySetup* is adjusted appropriately, all other functions of the toolbox are freely portable between setups; this includes complete experiments which can be used without any adaptations, for example in a touchscreen Skinner box and in a setup for head-restrained electrophysiology. Reaching this level of compatibility between different experimental setups keeps the programming of redundant code to a minimum and, in combination with the community-platform, this allows for a vivid exchange of experimental designs within and between labs.

2.3. Core functions

2.3.1. Start

A function used to initialize the toolbox and to set the path of toolbox-functions. This function calls *mySetup* to define all hardware-parameters of the setup.

2.3.2. mySetup

Sets all basic parameters of the setup in use. All other functions implementing hardware-interactions rely on the global variable SETUP which is set by *mySetup*. This function needs to be stored on each experimental computer and in turn allows free porting of all subsequent functions implementing hardware-interaction. By setting the field ‘interface’ to the value ‘keyboard’ it is possible to test functions without experimental hardware. Behavioral events can be simulated with keyboard or mouse; outputs are redirected to the command-window.

2.3.3. bIO

The core function for all interactions with external hardware other than monitors and speakers. This function depends on the interface used. So far it supports four types of interface described below in more detail.

2.4. Basic functions by category

In addition to these core-functions, the toolbox provides a set of basic-functions, making the programming of most behavioral

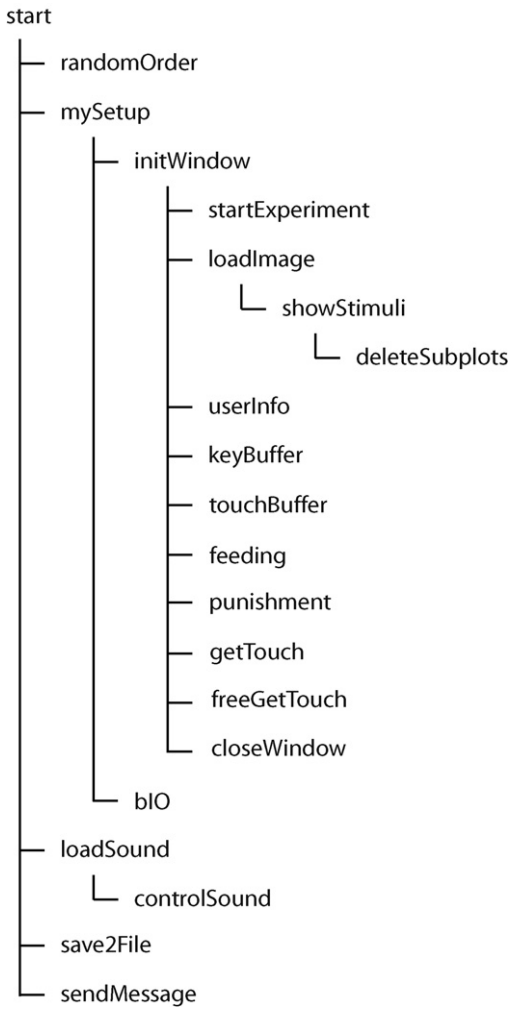

Fig. 1. The logical dependencies of the most important toolbox-functions.

experiments fast and simple. In the following we will briefly introduce the most important of these functions, for full reference and documentation please refer to the homepage of the project. Dependencies and logical hierarchy of the most important functions of the toolbox are depicted in Fig. 1.

2.4.1. Visual stimulation

For visual stimulation the toolbox provides three functions used to initialize a stimulus-window (*initWindow*), to load images (*loadImage*) and to show the images on screen (*showStimuli*). These functions can be used to display still and animated images.

2.4.2. Auditory stimulation

For auditory stimulation the toolbox provides functions for loading sounds as audio players (*loadSound*) which allows easily playing and pausing the playback of sounds without locking other Matlab-functions (*controlSound*).

2.4.3. Recording of behavioral events

Responses of the subjects can be traced over time and returned as a structured output (*keyBuffer*). The structure of the output allows to directly assess if a behavioral response was correct or incorrect and in turn to quickly give feedback.

2.4.4. Behavioral feedback

Top-level functions for the intuitive control of behavioral events (*feeding*, *punishment*) are available.

2.4.5. Control and analysis

Information of the state of an experiment can be displayed at any given time-point in the experiment (*userInfo*). Remote computers can be notified of the end and outcome of an experiment (*sendMessage*). Basic functions for displaying experimental results (*dispResults*), basic analysis and plotting of results (*plotLearning*) are available. Advanced options for the saving of experimental results are implemented (*save2File*).

2.4.6. Randomization

Most experiments require a randomized but highly controlled order of events. For this we incorporated a function that is capable of randomization with clearly defined parameters (*randomOrder*).

2.4.7. Event-codes

The functions for initialization (*start*, *mySetup*) hardware-interaction (*bIO*) and visual stimulus-presentation (*showStimuli*) provide options for triggering external hardware and software-events, for example in electrophysiological recordings.

2.5. Biopsychology design-rules

In order to guarantee full compatibility and portability of all functions, we recommend sticking to a small set of simple design-rules for programming. These rules are intended as a general guide, allowing full compatibility of extensions and added functions. Whenever any of these guides needs to be violated, we ask authors to explicitly state this in the documentation. As a first rule, all functions should be downwards-compatible; functionality of functions should be extended only by introducing additional parameter-value arguments or by inheritance. Second, all new functions should rely on the hardware definitions in *mySetup*, no hardware-specific code should be introduced in high-level functions. Whenever more definitions are required, *mySetup* should be extended. Third, to assure compatibility of analysis-functions, the output of all functions should comply with the documentation.

2.6. Implemented hardware-components

The Biopsychology-Toolbox at its current state supports four different types of interface, based on parallel-port, PCI-port, USB and touchscreen.

2.6.1. IO-Warrior 40 (Code Mercenaries, Hard- und Software GmbH)

IO-Warrior 40 is a USB-interface, supporting 32 general purpose IO-pins organized in four ports. Code Mercenaries offers a cheap starter kit that allows for easy testing and implementing of the IO-Warrior. For a full description of the interface, please refer to the vendor (http://www.codemercs.com/E_index.html). The Biopsychology-Toolbox provides a function for the direct access to eight inputs (port 0) and eight outputs (port 3). The outputs are capable of driving LEDs directly without any additional components. In addition PCB-schematics can be downloaded for the use of IO-Warrior in behavioral experimentation. This PCB allows driving of more power consuming hardware (e.g. feeders). The IO-Warrior allows a very easy approach to setting up experiments, only little prior experience and technical know-how is required. We have been using the IO-Warrior for more than two years successfully for the control of experiments.

2.6.2. Custom parallel-port interface

While the IO-Warrior offers a quick and convenient approach to experimental control, it is limited in terms of processing speed and flexibility. With a focus on electrophysiological experiments, we designed a parallel-port based interface. This IO-interface offers eight input and eight output pins to control behavioral experiments. Additionally, 16 behavioral event-codes can be sent to the recording hardware, for example to indicate onset and offset of stimulation or behavioral responses of the subject. Event-codes can be triggered either by Matlab or for more accurate timing by a hardware-trigger (e.g. a photo-sensor). For a full description and schematics please contact the authors. The parallel-port interface has been successfully used in experiments for over a year.

2.6.3. Touchscreen (Elo Touchsystems, 1528L 15 in. Medical LCD Desktop Touchmonitor)

To extend the capabilities of our Skinner boxes, we implemented a touchscreen-interface. The toolbox-functions used to access pecking coordinates and pecks on classical pecking-keys are fully compatible. This interface can be used to stimulate visually and to monitor behavioral responses. To control other aspects of the experiment (e.g. feeders) the combination with another interface is required. The touchscreen-interface has been used in experiments for over a year.

2.6.4. PCI-DIO24 (measurement computing corporation)

This is a PCI-card, one of the cheapest interfaces with direct Matlab support. This card is fast and easy to use, for a full list of specifications, refer to the manual (http://www.measurementcomputing.com/PDFmanuals/pci-dio24-lp.pdf). Note however, that this card is not supported by built in Matlab-toolboxes, it requires the Data-acquisition Toolbox. The PCI-DIO24 has been in experiments for more than three years.

3. Results

The functions that underlie the Biopsychology-Toolbox have successfully been used in our laboratory for more than three years. We used it in purely behavioral Skinner box experiments, in electrophysiology with free-moving and with head-restrained animals, in behavioral pharmacology experiments and with human subjects. All together this amounts to a total of roughly 30 distinct experiments and guarantees reliability of the toolbox. In our lab, programming novices required very little help when starting to implement behavioral paradigms using the Biopsychology-Toolbox. A brief introduction to the principles of programming in general and to the Matlab-syntax in particular suffices to understand the toolbox-documentation and examples and to self-reliantly set up basic paradigms.

While the toolbox represents an ongoing project and its current state is seen as the foundation of a lively community project, most of the aims of the project are reached. The toolbox is a reliable and easy-to-use programming environment without limiting power and flexibility of Matlab, it allows full portation of code between different types of hardware, the use of contemporary interfaces allows the implementation of novel behavioral paradigms and analysis. Using a touchscreen-interface for example allows to train pigeons on a discrimination paradigm that aims to mimic their natural feeding behavior and results in a significantly increased learning-rate, the 'multiple matching' paradigm introduced by (Huber et al., 2005).

4. Discussion

For many labs in behavioral Neuroscience, Matlab is the language of choice when it comes to data-analysis. Several toolboxes for data analysis are available and community-based projects allow a vivid exchange and free access to such tools. This is contrasted by a lack of standardization and availability of tools for the control of behavioral experiments. The Biopsychology-Toolbox project aims to fill this gap. This free and open-source project provides a forum and easy access to the basic tools needed to set up behavioral experiments in Matlab. The major aims of the project were to allow programming novices a straightforward approach to the implementation of behavioral paradigms, to maintain the full functionality and flexibility of Matlab and to allow full portability of complete experiments. In addition, the project provides all that is needed to get started without requiring much knowledge of software- or hardware-design. By providing a community-based platform, we wish to extend the scope of the project and give rise to a reliable and free set of tools that are independent of software-licenses or the doings of a specific company.

Acknowledgements

We like to thank Prof. Dr. h.c. Onur Güntürkün for supporting us and for providing the necessary means to initialize this project. Additionally, we like to thank Nadja Freund and Katja Brodman and all other colleagues at the Institute of Cognitive Neuroscience in Bochum that helped improving and debugging the toolbox by giving us valuable feedback.

References

Brainard DH. The psychophysics toolbox. Spat Vis 1997;10(4):433–6.
Dingler R, Kadden RM, Snapper AG. An introduction to state notation and SKED. Behav Res Meth Inst 1976;8(2):69–72.
Genovesio A, Mitz AR. MatOFF: a tool for analyzing behaviorally complex neurophysiological experiments. J Neurosci Methods 2007;165(1):38–48.
Huber L, Apfalter W, Steurer M, Prossinger H. A new learning paradigm elicits fast visual discrimination in pigeons. J Exp Psychol Anim Behav Process 2005;31(2):237–46.
Meyer T, Constantinidis C. A software solution for the control of visual behavioral experimentation. J Neurosci Methods 2005;142(1):27–34.
Skinner BF. A case history in scientific method. Am Psychologist 1956;11:221–33.