1. Introduction

This document describes coding schemes and system configurations for the 'Real-time Experimental Control with Graphical User Interface' (REC-GUI) Framework.

The graphical user interface (GUI) is coded in Python using the Tkinter package. The source code for the GUI is provided for customization.

Python code and MATLAB scripts for implementing the REC-GUI Framework are provided. Depending on experimental needs, additional resources may be required.

The REC-GUI Framework is designed to be an open-source project, so we encourage others to contribute to its continued development.

2. Contact

You can visit us at: https://rosenberg.neuro.wisc.edu/, https://recgui2018.wixsite.com/rec-gui, or https://github.com/rec-gui.

REC-GUI support email: recgui2018@gmail.com.

You can directly reach Byounghoon Kim at bkim10@wisc.edu or
Ari Rosenberg at ari.rosenberg@wisc.edu.

3. Citation

If you use the REC-GUI Framework in your research, please include a citation to:

Byounghoon Kim, Shobha Channabasappa Kenchappa, Adhira Sunkara, Ting-Yu Chang, Lowell Thompson, Raymond Doudlah, and Ari Rosenberg (2018) Real-Time Experimental Control using Network-Based Parallel Processing. bioRxiv 392654; doi: https://doi.org/10.1101/392654.

4. Installation and Setup

4.1. Required Packages

The REC-GUI Framework is coded in Python and MATLAB, and requires several other packages. Since these packages have different developers, we recommend checking with them regarding updates and usage. Skipping required packages can cause serious system errors.

Main code for the REC-GUI Framework.

- Download/clone code from GitHub (https://github.com/rec-gui).

Basic packages required for the REC-GUI Framework:

- Python2.7 - base python interpreter.
- Numpy - for calculations using arrays or matrices (http://www.numpy.org).
- Tkinter - for building the GUI (https://tkdocs.com/tutorial/install.html).
- MATLAB (2017a or greater is recommended).
  - MATLAB's Computer Vision System Toolbox.
  - MATLAB's Data Acquisition Toolbox (if using USB-1680G for IO).

- Psychtoolbox 3 for visual stimulus generation (http://psychtoolbox.org/).

Selected packages for specific system configurations:

1. To use EyeLink (SR Research, Inc.).
    - Go to: https://www.sr-support.com/
    - Download 'EyeLink display software' for your operating system:
        - Linux - https://www.sr-support.com/forum/downloads/eyelink-display-software/46-eyelink-developers-kit-for-linux-linux-display-software
        - Windows - https://www.sr-support.com/forum/downloads/eyelink-display-software/39-eyelink-developers-kit-for-windows-windows-display-software
        - Mac OS X - https://www.sr-support.com/forum/downloads/eyelink-display-software/45-eyelink-developers-kit-for-mac-os-x-mac-os-x-display-software

2. To use USB-1680G (Measurement Computing, Inc.).
    - USB-1680G is a multifunctional DAQ used to handle analog IO or digital IO.
    - Linux
        - Ready for use with REC-GUI: https://pypi.org/project/pydaqflex/
        - https://github.com/mccdaq/uldaq
    - Windows
        - https://github.com/mccdaq/mcculw

3. To use a PC Parallel Port.
    - This is a simple solution for digital IO.
    - Linux
        - Ready for use with REC-GUI - https://github.com/pyserial/pyparallel
    - Windows
        - https://pypi.org/project/pyparallel/

4.2. Downloading and Installing the REC-GUI Framework

- **THE REC-GUI WEBSITE (https://recgui2018.wixsite.com/rec-gui) INCLUDES A VARIETY OF RESOURCES INCLUDING A FORUM TO SHARE IDEAS AND SOLUTIONS. WE WELCOME DEVELOPMENT CONTRIBUTIONS.**

- A history of coding developments is maintained on GitHub: https://github.com/rec-gui.

4.2.1. Configuration Files

default.conf – required default system configuration file (**SEE SECTION: 5**).

<subject name>.conf – required subject configuration file (SEE SECTION: 5).

fixation.conf – an example system configuration file for implementing a gaze fixation task. To be used in conjunction with 'Fixation.m' (SEE SECTION: 7.1.1).

calibration.conf – an example system configuration file for implementing an eye calibration task. To be used in conjunction with 'Calibration.m' (SEE SECTION: 6.2.3).

receptive_field_mapping.conf – an example system configuration file for implementing a receptive field mapping task. To be used in conjunction with 'Receptive_Field_Mapping.m' (SEE SECTION: 6.2.4).

stim_default.mat – a MATLAB configuration file that contains default stimulus parameters. It is tailored for particular stimuli rendered in MATLAB.

4.2.2. Python Files

The following is a list of files required for the GUI. Essential classes that need to be modified in order to adapt the REC-GUI Framework to new experimental paradigms are marked in bold.

- **ANALOG.PY – A CLASS FOR HANDLING ANALOG TO DIGITAL CHANNELS WHEN USING AN EXTERNAL ANALOG-TO-DIGITAL CONVERTER (**e.g., USB-1680G**).**
- **ARBITRATOR_SERVER.PY – A CLASS FOR COMMUNICATION BETWEEN THE GUI AND ITS COUNTERPARTS.**
- **BUTTON_FUNS.PY – A CLASS THAT DEFINES CALL-BACK FUNCTIONS FOR THE USER PROGRAMMABLE BUTTONS PRESENTED WHEN A USER SELECTS 'NONE' FOR THE EYE TRACKING SYSTEM (SEE SECTION: 6.2.4).**
- **CALIBRATION.PY – A CLASS FOR HANDLING EYE CALIBRATION PROCEDURES.**
- **CONSTANTS.PY – A CLASS THAT DEFINES ALL SYSTEM CONSTANTS.**
- **DATA_COLLECTION.PY – A CLASS FOR SAVING THE DATA FILE.**
- **DIGITALIO.PY – A CLASS FOR INPUT/OUTPUT DIGITAL BITS TO CONTROL TTL PULSES USING USB-1680G (MEASUREMENT COMPUTING, INC.)**
- **EYE_INTERPRET.PY – A CLASS FOR COMMUNICATION BETWEEN THE GUI AND AN EYE TRACKING SYSTEM (E.G., EYELINK, ANALOG SEARCH COIL, ETC.).**
- **FILE.PY – A CLASS FOR WRITING THE DATA FILE.**
- **GLOBAL_PARAMETERS.PY – A CLASS THAT DEFINES ALL SYSTEM AND CONFIGURATION VARIABLES.**
- **GUI.PY – A CLASS FOR BUILDING THE GUI (IN TKINTER). START HERE TO CHANGE ANY GUI COMPONENT. A PROGRAMMING REFERENCE IS FOUND ON THE TKINTER WEBSITE: (**https://tkdocs.com/tutorial/firstexample.html**).**
- **LOGGER.PY – A CLASS FOR UPDATING THE DATA LOG WINDOW.**

- MAIN.PY – MAIN FUNCTION TO SPAWN THREADS FOR THE GUI, EYE TRACKING, AND SYSTEM COUNTERPARTS (E.G., A STIMULUS RENDERING/PRESENTATION CPU RUNNING MATLAB).
- READ_TASK_PARMETERS.PY – A CLASS FOR IMPORTING PARAMETERS FROM A SAVED TASK CONFIGURATION FILE.
- SELECT_EYE_RECORDING.PY – A CLASS THAT PROVIDES A GUI POP-UP WINDOW FOR SELECTING THE EYE TRACKING METHOD.
- STAIRCASE.PY – A CLASS FOR IMPLEMENTING A STAIRCASE PROCEDURE IN A PSYCHOPHYSICAL TASK.
- UTILITY.PY – A CLASS CONTAINING CONVERSION ROUTINES FOR CALCULATIONS.
- VERGENCE_VERSION.PY – A CLASS FOR CALCULATING VERGENCE AND VERSION EYE POSITION ERRORS.
- WIDGET_API.PY – A CLASS FOR PRESENTING SPECIALIZED TOOL PANELS SUCH AS EYE CALIBRATION, RECEPTIVE MAPPING, ETC.

4.2.3. Example MATLAB Scripts (Useful Learning Tools)

Three MATLAB scripts that implement specific tasks are provided as examples for learning the overall coding scheme of the REC-GUI Framework. Two additional skeleton scripts are provided as templates for preparing new tasks.

- Fixation.m – an example script that implements gaze fixation training using standard operant conditioning procedures. This code presents visual targets at user-specified locations. If a subject fixates their gaze on the target for a specified duration of time, positive reinforcement is provided. This script also provides a useful starting point for learning how to use Psychtoolbox with the REC-GUI Framework. Use ‘fixation.conf’ to configure the GUI to run this script.

- Calibration.m – an example script for performing eye calibration. This code implements a manual eye movement calibration procedure using the GUI. The user manually selects the location of a fixation target, and indicates when the subject’s gaze is fixed on the target. After the subject has fixated multiple target locations, the GUI calculates offsets and gains to apply to the raw eye movement signals in order to map those signals to the true fixation locations. Use ‘calibration.conf’ to configure the GUI to run this script.

- Receptive_Field_Mapping.m – an example script for mapping the visual field location of a neuron’s receptive field during neurophysiological recordings. In this script, real-time interactive control of a visual stimulus is coordinated between the experimental control and stimulus CPUs. This script also shows how to implement the SDK toolbox (Ripple, Inc.) as an example of how to update event codes in a data acquisition system (e.g., used for electrophysiological recordings). Use ‘Receptive_Field_Mapping.conf’ to configure the GUI to run this script.

- start_coding.m – a script that provides a template for preparing a new task. It contains code for initializing a UDP connection and a while-loop in which experiment specific code can be added. Use ‘start_coding.conf’ to configure the GUI to run this script.
- start_coding_closedloop.m – a script that shows how to establish interactive control between the GUI and MATLAB. It contains simple functions with a basic coding structure to receive computer mouse information from the GUI and send it back to the GUI. This input-output cycle can provide the basis for closed-loop control. Similar functions are used in ‘Receptive_Field_Mapping.m’. Use ‘start_coding_closedloop.conf’ to configure the GUI to run this script.