RSVPKeyboard Display Documentation

Aziz Koçanaoğulları, Tab Memmott

October 14, 2017

1 Introduction

RSVP Display is used to present stimulus on the screen for different tasks. Python implementation aims to make it as modular as possible which includes being easy to populate and easy to dispose if not required. Code documentation (variable names and function descriptions) is already included in the code scripts. For further reference refer to the comments in the code.

2 Design Choices

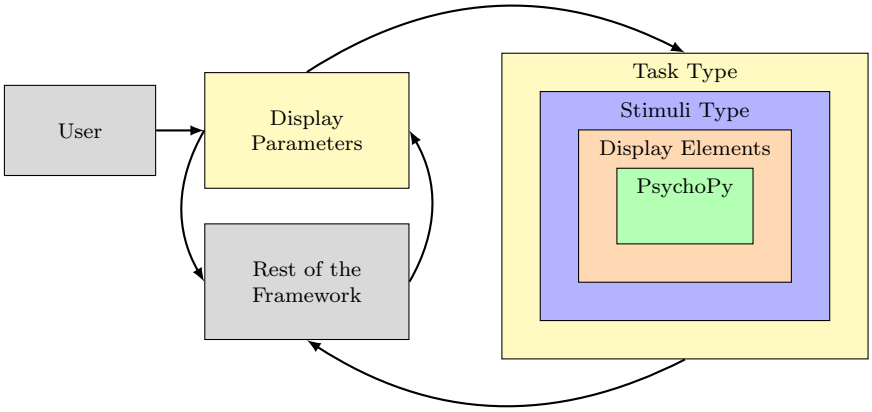

Figure 1: We denote the relationships with arrows. All boxes includes elements using the boxes inside them. For example display elements are either cherry picked from the PsychoPy or designed using the elements within the Psychopy.

The display is built on top of PsychoPy, a display module written in python specifically designed for brain computer interface (BCI) stimulus presentations.

While defining different elements we use the design procedure shown in Fig.1. Elements in the figure are,

- **Display Elements:** Basically text boxes, rectangles etc. There are concatenation of elements declared for special use.(Bar Graph: Designed for feedback mechanism, Multicolor text: Psychopy does not support text with different colored letters.)

- **Stimuli Type:** Represents a sequence. Controls where,when and what to type in a sequence. For example in RSVP, the stimuli type flashes letters with specified color on a specified location for a defined interval.
- **Task Type:** Covers the task of different stimuli types. This module is just an extension of the stimuli type module. For example copy phrase. This module is aware of what is going to be presented, therefore specifically inserts display elements on the screen. However this module specifically defines the epoch.

We design the elements within the display being aware of the designed modules they include. As can be observed from the figure, **stimuli type** are designed using the **display elements** and aware of their properties. However display elements are not designed specifically covering the properties of stimuli types. This allows us to populate the module folder without constraining the developer. The developer should be only aware of the sub-modules.

3 How to Design

3.1 Display Elements

Display elements are just an extension to the conventional items provided by the PsychoPy. For example PsychoPy does not support multiple colored text. In order to have such application we require to define multiple text stimuli and concatenate it into a single string. Based on the design procedure if we have further ideas of making each of these elements more modular, it will be easy to do as long as we do not interfere with the I/O relations. Elements require different types of inputs where the data structure of the inputs are specified in the script. But basically, the display element should be aware of,

- window: screen window to be displayed on
- position: position on the window. Where the window size is described using $[x, y] \in [-1, 1] \times [-1, 1]$. Where $\{-1, -1\}$ denotes bottom left corner.
- size: usually controlled via height of the bounding box.
- context: characters if string, color etc.

based on these constraints, display elements can be defined and called independently. As the next step we want to incorporate the error handling messages. As the most basic element in the design road map we can also state that display elements are only aware of the PsychoPy, therefore can be created freely. The display elements have to incorporate these main functions:

- draw(): draws the item on the window.
- update(): updates the information of the element.

In display elements bar graph elements has a different location. We decided that, the bar graph response to the user should be dynamic. Changes position over time, instead of directly flashing. Therefore having a direct draw function does not cover entire properties of the bar graph. The dynamic objects should have,

- schedule_to(): Schedules the dynamic object to the final destination.
- animate(k): Animates the object using the step k. Definition of the step, allows the developer animate forward and backward using the same function.

For bar graph draw() function is incorporated into the animate function.

3.2 Stimuli Type

The stimuli type is the main design choice taken in the display so far. Stimuli type is different for different BCI displays (e.g. RSVP, Shuffle, Matrix etc.). In different stimuli types we know number of static information on the screen and number of dynamic elements in the screen differs. Stimuli type is the cornerstone of defining the display. Stimuli type is aware of how to display a sequence on the screen since it differs from stimuli to stimuli. For example RSVP keyboard flashes symbols on a fixed location in the screen for a specified inter stimulus interval and for a fixed duration. Once finished the sequence gives feedback based on the result. This information is embedded in the stimuli type.

3.3 Display Mode

On top of stimuli type there are different types of tasks such as free spelling, calibration, copy phrase etc. We can find similarities of these mods among different stimuli types, however we keep the bottom up design here. Therefore all the top modules should be aware of the inner modules. However they should be unaware of the top modules, which is essential for modularity. This allows us to define different modes easier. For example in copy phrase besides the stimuli, everything works similar in matrix speller and RSVP. However this information is neglected. In future releases this information can be incorporated in order to get rid of redundancy.

The design choices sets display modes to communicate with the rest of the framework and the user.

4 Objects in Display

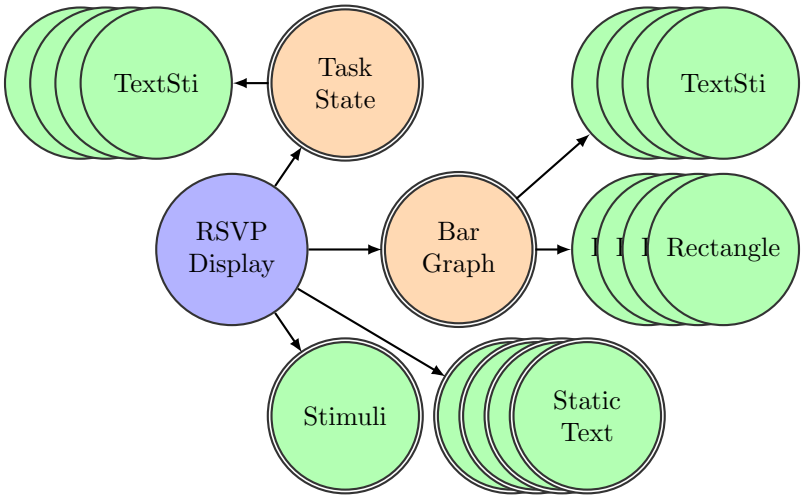

Figure 2: Design graph and object relations in RSVPDisplay object. Double circled elements are in direct interaction with the RSVPKeyboard display. Single circled elements are used by other objects. Orange denotes the objects designed as explained in Fig.1, green denotes the elements from PsychoPy.

The design choice of the RSVPDisplay is explained in Fig.2. Colors are consistent with Fig.1 to help developers to add/discard items from the design. In this section each element is explained

to be a reference for further development and integration. Similar to the previous sections the green elements are well documented in PsychoPy's website, therefore the information about the green elements are skipped. **Display elements** are explained in order to justify design choices and serve as a handbook to the code. For further information refer to the code comment lines.

4.1 Display Elements

4.1.1 Bar Graph

Feedback display element to the user after a sequence. Given a score function of letter. Normalizes and animates to the designated location. Currently bar graph animates horizontal and does not support vertical animation. in future releases vertical bar graph can be supported.

Attributes	texts(list[TextSti]): Uses PsychoPy defined text stimuli to insert the domain names of the bar graph. These texts has their own parameters controlled using Bar Graph. bars(list[RectSti]): Uses PsychoPy defined rectangle stimuli to denote the weights of the domain elements.
Arguments:	domain related: Accepts an integer as [number of elements] to be displayed. Number of texts and bars are equal to the number of the elements in the domain. position related: Accepts two tuples to represent the position of the bar graph. One tuple for [top right] corner, the other for the [bottom left] corner of the rectangular region of the bar graph. visual: Accepts [color] information for both the bars and the texts. Accepts [text font] information. animation: Accepts an integer to define an interval for the bar graph [maximum number steps] in animation.
Functions:	update(letters,weight): Changes domain variables and the weights. draw(): Draws the bar graph. schedule_to(letters,weight): Schedule the weights of the endpoint for animation. Observe that with update bar graph can nor animate. animate(step): Animates the bar graph using the step parameter. The boundary is maximum number of steps. reset_weights(): Sets the weights of the bar graph to zero.

4.1.2 Multicolored Text

Psychopy does not support a multicolored text. The implementation consists multiple text stimuli with different color information. The font however is kept consistent. The multicolored text accepts all fonts, measures the width of the letters of the specified fonts in order to write them in a visually fashionable way. Currently updating the multicolored text is achieved by recreating it, as modifying element location is non trivial. In further releases this problem can be tackled in a fancier way.

Attributes	texts(list[TextSti]): Uses PsychoPy defined text stimuli which later concatenated into one string with different colors.
Arguments:	domain related: Accepts a string [text] to make it multicolored. Color information is passed as a list, if the list has just one element, the multicolored text is basically a TextSti. position related: Accepts a tuple to determine the center of the rectangular region the text covers. Center location is chosen in order to preserve the consistency with the Psychopy objects. misc: Accepts all other variables Psychopy objects accept.
Functions:	update(text,color_list,pos): Changes elements, color information and the position of the multicolored text using the arguments passed. draw(): Draws the multicolored text.

4.2 RSVP Display

In this subsection the architecture of the RSVP Display is explained which is visualized in Fig.2. RSVP Display is specified to stimuli a single flashing symbol on a fixed location.

Attributes	Task(TextSti): Task state of the framework. Texts(TextSti): Static textures during the stimuli presentation sequence. Stimuli(TextSti): Dynamic element that flashes the symbols on the screen. Bar(BarGraph): Bar graph of the framework for giving a feedback to the user. clock(core_clock): System clock for synchronization.
Arguments:	Task Text(MulticoloredText): RSVPKeyboard accepts arguments the multicolored text object accepts in order to put it on the screen. Static Text(list[TextSti]: Accepts all arguments that is required for text stimuli object from PsychoPy accepts in a list form. The number of static elements are determined by the length of the list. The lists covering the texts should be of same length. Stimuli(TextSti/FigStu): Accepts all arguments that is required for text stimuli object from PsychoPy accepts. The difference between text and figure is the definition of the element. If given location, the stimuli becomes a figure otherwise a string. Bar (BarGraph): Accepts all arguments that is required for BarGraph object.
Functions:	draw_static(): Draws the static elements during a sequence (task, static texts) on the screen. schedule_to(ele_list, color_list, time_list): Schedules the flashing information of the next sequence using the arguments of the function. Sequence length is defined by the length of these functions. update_task(text,color_list,pos): Updates the task state of the framework. do_sequence(): Animates a sequence if scheduled. show_bar_graph(): Animates the bar graph if scheduled.

RSVP Display modes are constructed on top of this module. Therefore the modes are basically specifications of the general case of the RSVP Display. They share the functionality, however number of static texts, color of the task state may differ. In order to preserve simplicity, information is stored in objects.

Should we cover the how the modes differ and what they are used for? Or is it someone else's bussiness.