Capítulo 1

Resultados

Este trabajo de título consistió principalmente en implementar una herramienta de software para configurar y ejecutar experimentos, por este motivo se opta por presentar los resultados a modo de un manual de usuario. De esta forma, a continuación se mostrará, paso a paso, el proceso de configuración y ejecución de un experimento.

1.1. Configurar un experimento.

Para configurar y almacenar un experimento se requieren las siguientes dependencias:

```
1 import os
2 from saccadeApp import SaccadeDB
3 from saccadeApp import Experiment, Test, Frame, Component
```

Donde `os` se utiliza para la manipulación de directorios y archivos, `SaccadeDB` para el manejo de la base de datos y `Experiment, Test, Frame, Component` para configurar todos los elementos asociados a un experimento.

En primera instancia, es necesario crear el directorio en donde se almacenarán las configuraciones:

```
1 base_dir = u'C:\\Experiments'
2
3 if not os.path.isdir(base_dir):
4     os.mkdir(base_dir)
```

Luego, se crea la base de datos en este directorio, para lo cual se utiliza el módulo `SaccadeDB` . Al inicializar un objeto de este tipo debe indicarse la ubicación y nombre del

archivo de base de datos a utilizar, si este no existe se creará uno. En caso de no especificar el archivo de base de datos, esta función crea uno en la carpeta donde se encuentra el ejecutable del programa con el nombre por defecto ”saccadedb.sqlite3”.

```
data = SaccadeDB(base_dir + u'\\test_database.sqlite3')
```

Una vez que la base de datos fue creada y es posible almacenar el experimento se comienza su configuración. El experimento a construir se compondrá de dos tareas que se ejecutarán una sola vez de forma secuencial y luego de que el usuario presione la tecla ”espacio”.

La primera corresponderá a una tarea asociada a movimiento prosacádico con efecto de *overlap*, lo que significa que el estímulo central se encontrará presente en todos los cuadros. Los componentes a configurar son dos: El estímulo central (cruz de color blanco) y el objetivo (cuadrado de color rojo), que serán ubicados en el centro de la pantalla y $16°$ a la derecha, respectivamente.

```
cross = Component()
cross.set_name(u'CP')
cross.set_size(0.5)
cross.set_shape(u'cross')
cross.set_color(u'white')
cross.set_position(0.0, 0.0)

target = Component()
target.set_name(u'TP')
target.set_size(0.5)
target.set_shape(u'square')
target.set_color(u'red')
target.set_position(16.0, 0)
```

Con los componentes creados, es posible configurar los cuadros de la tarea. El primer y tercer cuadro son iguales y solo presentan el estímulo central, mientras que el segundo presenta además el objetivo. Los tiempos de presentación de cada cuadro son $1{,}8[s]$ para el primer cuadro, $1{,}5[s]$ para el segundo y $1{,}0[s]$ para el tercero.

```
frame1 = Frame()
frame1.set_name(u'Central Stimulus')
frame1.set_time(1.8)
frame1.set_as_task(False)
frame1.set_color(u'black')
frame1.component_add(cross)

frame2 = frame1.copy()
frame2.set_name(u'Target')
frame2.set_time(1.5)
frame2.component_add(target)

frame3 = frame1.copy()
frame3.set_time(1.0)
```

Por último, se crea la tarea y se le asigna un nombre (además de una descripción, opcionalmente) y se agregan los cuadros configurados.

```
test1 = Test()
test1.set_name(u'Overlap task')
test1.set_quantity(1)
test1.frame_add(frame1)
test1.frame_add(frame2)
test1.frame_add(frame3)
```

La segunda tarea a configurar corresponde a una única imagen centrada. En este caso, el cuadro fue configurado para no depender del tiempo pero si de la entrada de teclado realizada por el usuario.

```
image = Component()
image.set_image(u'couple.png')

frame4 = Frame()
frame4.set_name(u'Image Presentation')
frame4.set_as_task(True)
frame4.set_keys_allowed(u'space')
frame4.set_keys_selected(u'space')
frame4.component_add(image)

test2 = Test()
test2.set_name(u'Search task')
test2.set_quantity(1)
test2.frame_add(frame4)
```

Para crear un objeto de experimento es necesario, en primera instancia, asignarle una base de datos en donde se almacenarán sus configuraciones. Luego, se indica su código único de experimento y se le asigna una dupla única de nombre (o tipo) de experimento y la versión a la cual corresponde. En este caso, se selecciona que cada tarea requiera que el usuario presione la tecla espacio para iniciar y que no se muestre un diálogo para ingreso de características del usuario al inicio de cada experimento. Finalmente, se agregan las tareas previamente configuradas y se almacena en la base de datos.

```
experiment = Experiment()
experiment.set_database(data)

experiment.set_code(u'expcode_01')
experiment.set_info(u'Experiment test', u'v1.0')
experiment.set_space_start(True)
experiment.set_dialog(status=False)
experiment.test_add(test1)
experiment.test_add(test2)
experiment.save()
```

1.2. Ejecutar un experimento.

Para ejecutar un experimento que se encuentre almacenado en la base de datos es necesario crear un perfil de configuración que indique tanto el monitor como el *eye tracker* a utilizar, además de la carpeta donde serán almacenados los eventos resultantes. Para ello son necesarias las siguientes dependencias:

```
import os
from saccadeApp import SaccadeDB
from saccadeApp import Master
from saccadeApp import ExperimentHandler
```

Donde `os` se utiliza para la manipulación de directorios y archivos, `SaccadeDB` para el manejo de la base de datos, `Master` para la configuración del hardware a utilizar y setear la carpeta donde se almacenarán los resultados y `ExperimentHandler` para ejecutar el experimento seleccionado.

En primera instancia, es necesario crear el directorio en donde se almacenanarán los resultados. En este caso, corresponderá en una subcarpeta del directorio creado en el proceso de configuración del experimento.

```
base_dir = u'C:\\Experiments'
data_dir = base_dir + u'\\Events'

if not os.path.isdir(data_dir):
    os.mkdir(data_dir)
```

De forma posterior, debe abrirse la base de datos para poder almacenar el perfil de configuración que será creado.

```
data = SaccadeDB(base_dir + u'\\test_database.sqlite3')
```

Para asegurar que los estímulos presentados por pantalla tengan las dimensiones definidas en la configuración del experimento, es necesario verificar que el monitor a utilizar tenga un perfil de configuración. Dichos perfiles deben ser definidos por medio de la herramienta dispuesta por PsychoPy llamada *"Monitor Center"*[1] que puede ser ejecutada directamente desde el script mediante el comando `Master.open_psychopy_monitor_center()` . En ella (ver figura 1.1) es posible definir las proporciones físicas del monitor, sus características lumínicas y distancia al usuario. Para efecto de este ejemplo, se creará un perfil llamado

[1] Centro de monitores.

”my$_monitor''yseingresarnunicamentelosatributosasociadosaladistanciaalusuario$(53[cm]), $tamaoer$
y ancho efectivo de la pantalla $(47{,}5[cm])$.

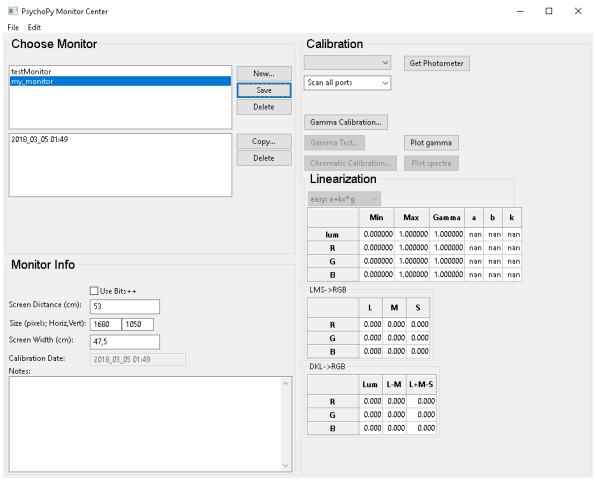

Figura 1.1: Centro de monitores de PsychoPy.

```
1
2 data = SaccadeDB(u'C:\\Experiments\\test_database.sqlite3')
3 execution = ExperimentHandler()
4
5 if execution.load_experiment(db=data, mas=u'test_profile', exp=u'expcode_01'):
6     execution.save_parameters()
7     execution.execute_experiment()
```

Algoritmo 1.1: Ejecución de un experimento.

```
    Master.
1 conf = Master()
2 conf.set_database(data)
3
4 conf.set_name(u'test_profile')
5 conf.set_screen(1)
6 conf.set_monitor(u'test_monitor')
7 conf.set_tracker(u'eyetribe')
8 conf.set_experiment_path(data_dir)
9 conf.save()
```

respecto del objetivo: 1) Es posible configurar y almacenar [referencia a código de ejemplo de configuración y screenshot de los datos almacenados en la tabla]: - Configuración del sistema en donde se ejecutará el experimento : * Eye tracker a ser utilizado. * Qué monitor será utilizado y cuál es su perfil de configuración. * Directorio donde se almacenarán los

resultados del experimento. - Configuración de los experimentos: * Configurar sus tareas, cuadros y componentes. * Opciones de ejecución: + Tiempos de descanso. + Tipo de ejecución de las tareas: aleatoria o secuencial. + Si se requiere alguna acción del usuario para comenzar cada tarea o la ejecución será ininterrumpida. * Opciones asociadas al diálogo previo a cada ejecución (sesión). Esto dice relación con la información que interesa rescatar del paciente que realizará el experimento.

2) Es posible ejeutar el experimento de forma tal que se almacene tanto la información del estímulo presentado como de las respuestas producidas por el usuario [screenshot del árbol de capetas producido dado la configuración anterior junto a imágenes del archivo de resultados y el contenido de los diccionarios generados]. * Se almacena un respaldo de la configuración utilizada en cada sesión en un archivo de diccionario de formato yaml, lo que facilita su lectura por parte de los investigadores. * Se almacena un registro de los eventos ocurridos duraante el laboratorio en un archivo de formato estándar (hdf5) para: + Eye tracker. + Acciones de teclado. + Eventos en el monitor dependientes de los estímulos presentados.