Human Connectome Project - Lifespan Task fMRI Battery Install and Run Instructions

This archive contains Psychopy scripts for the Practice and Scanner fMRI Tasks used in the Human Connectome - Lifespan Project. There are two primary task scripts to run: the compiled `_Practice.py` introduces subjects to the task they will perform, and `_Scan.py` presents the complete task during fMRI data acquisition.

This README contains information on initial setup, how to run each script, output files, and how to get help. The tasks were built using the Psychopy "Builder" view; the Psyexp and patch files are available upon request to the HCP Mailing List but not distributed by default to avoid accidental changes during task administration.

Table of Contents

IMPORTANT: All scripts in the HCP-L Battery are configured to run on a 2-monitor setup. Instructions for RAs are shown on the primary screen, and the actual task that the participant will see is shown on a second, larger monitor. *Running on a single monitor is not supported.*

Example videos of both the `_Practice.py` and `_Scan.py` tasks are available to preview the task; the participant screen is shown on the left and the experimenter's screen is shown on the right.

Task Introduction - Resting State / mbPCASL EyeCam Scripts

This script displays a fixation cross to the participant while recording eye video via frame grabber or webcam (tested with Epiphan DVI2USB 3.0), and displays real-time eye video for RA on the main monitor. Additionally, the REST and mbPCASL scan times for different age groups are included, so that the length and number of scans match the HCP-Lifespan protocol.

Initial Setup

The EyeCam script reads frames from the connected video camera or EyeLink display using a frame grabber (we tested with the Epiphan DVI2USB 3.0). The frame grabber converts the analog video frames to a USB input, which is read in this script. Before use, the script must be calibrated with

the correct aperture (with the script `calibrate_eyecam.py`) to set the region of the video input to display and save into a video for further processing or analysis.

Unique EyeCam `siteConfig.yaml` Options:

- **dualCam**: This is an indirect way of choosing the camera to grab frames from. Set this to 1 for if runnign the eyecam script on a computer with a built-in camera (e.g., on a Macbook), and set to 0 if the computer does not have a built-in camera or if you are testing script w/ built-in camera.
- **record**: 'yes' or 'no' to record video from the camera (vs. simply presenting the fixation cross for the correct length and number of scans)
- **useAperture**: 'yes' or 'no'; set to no to record the entire window (not recommended)
- **aperture**: Region of the screen to grab. This is set by the `calibrate_eyecam.py` script and should probably not be set by manually editing the `siteConfig.yaml` directly.

Output Files

Simple long-form style csvs are written indicating ScanStart, Countdown (for rest BOLD-weighted scans, during which the scanner is recording but not at steady-state equilibrium), fixation start and run end.

EyeCam videos are saved with H.264 codec and an mp4 container extension with the same prefix as the text file. Video recording time coincides with scan start and duration.

Additionally, a `_ts.csv` file is saved with the timestamps of individual frames (registered to the start of the experiment) to be used for differencing frame times.

Quick Start: Running the Task

Practice

The practice script is not provided in this package. Participants were given a description of the fixation cross prior to practicing other tasks.

Scan

To run the *scan* task, open `REST_Scan.py` or `mbPCASL_Scan.py` in Psychopy Coder View and click the green "Running Man" icon.

Hit `OK` to use the following default options:

- age = `<Subject Age>` (for selecting correct scan duration)
- sessionID = `<SubjectID>`
- runNumber = `1` (unless earlier scans were aborted)
- scanType = `REST` or `mbPCASL` (dropdown, default should be correct)
- testMode = `<unchecked>` (force usage of the built-in camera if available)

Logfiles and videos are saved in the `data` directory with the task name, subject id, run number, and datetime. See below for more information on the information stored in each output files (.log, .csv, and .psydat). For more information on the dialog box options, see the `Input Option Glossary` below.

Initial Setup

Psychopy: One-time per computer installation

Download Psychopy version **1.83.04** from the appropriate link:

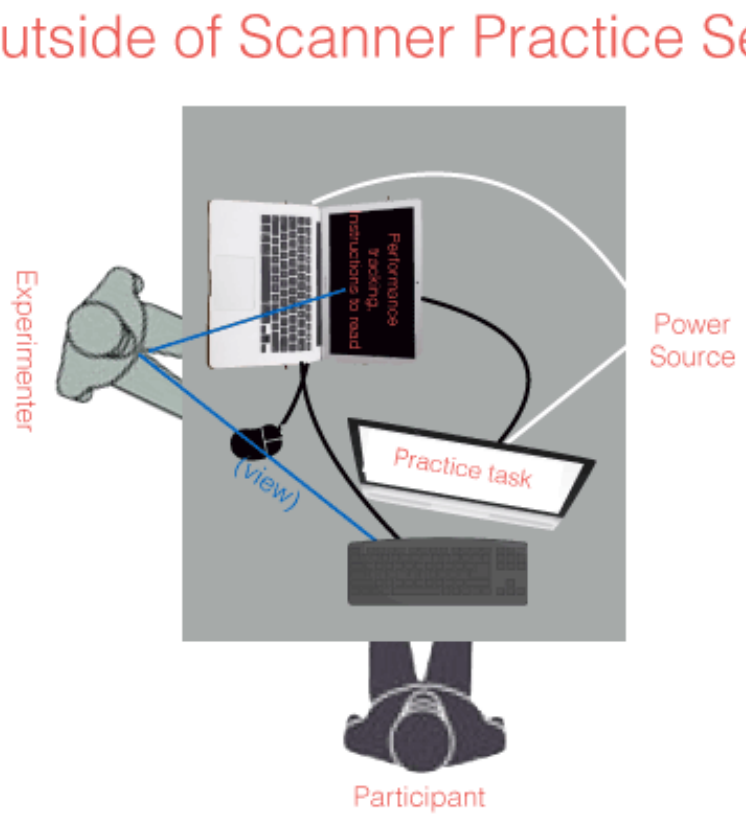
- **Mac**: https://github.com/psychopy/psychopy/releases/download/1.83.04/StandalonePsychoPy-1.83.04-OSX_64bit.dmg
- **Windows**: https://github.com/psychopy/psychopy/releases/download/1.83.04/StandalonePsychoPy-1.83.04-win32.exe
- **Linux**: `apt-get install psychopy=1.83.04` on neurodebian, or ask for instructions

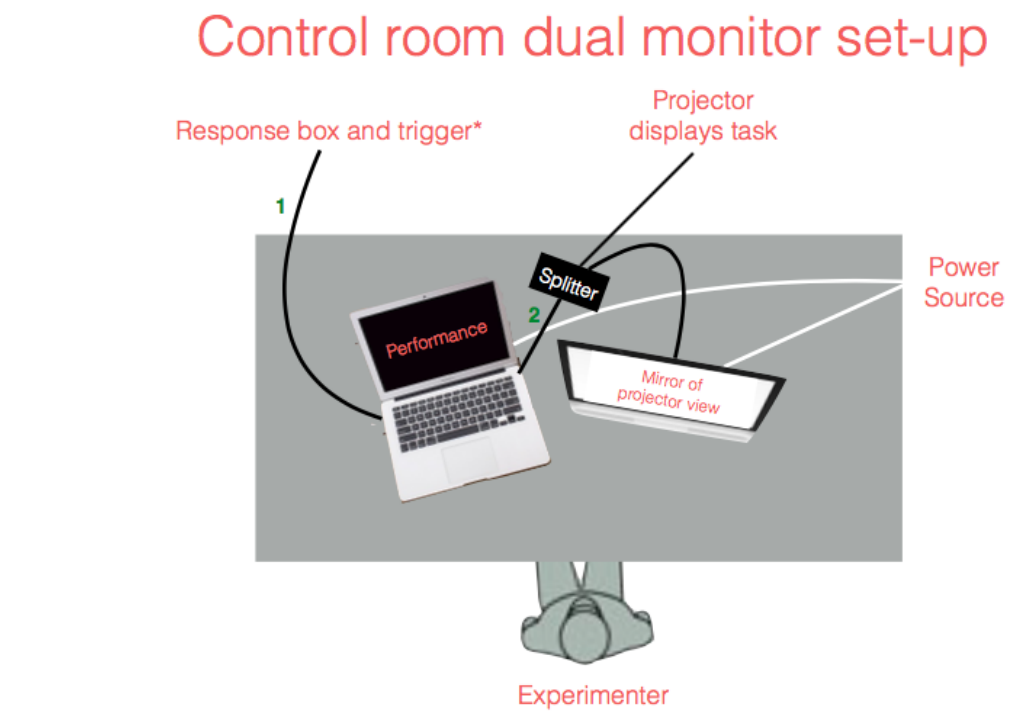
The tasks have been extensively tested on version **1.83.04** and are not guaranteed to work correctly with later versions. Newer versions are likely to work, but **have not been tested**.

Physical Display Configuration

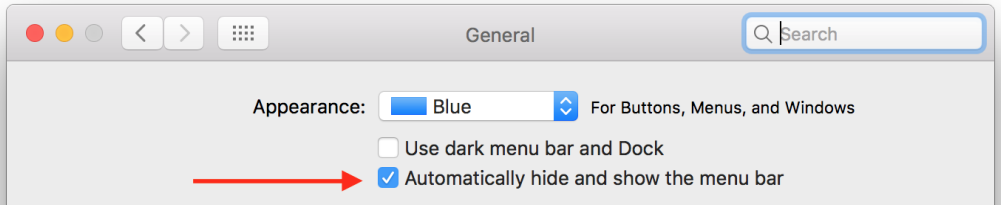
All scripts are configured to run on a *2-monitor setup*: Instructions for RAs are shown on the primary screen, and the actual task that the participant will see is shown on a second, larger monitor.

Please note that for the task practices that happen outside the scanner, it is ideal for participants to respond using a practice 2-button button box identical to the one they will use inside the scanner. However, any keyboard keys can be configured; see the Button-box mapping section below for instructions.





NOTE: On some versions of Mac OS, the menu bar is drawn on top of the second screen. To avoid showing it, you can set the menu bar to "hide" and not be shown. To do this, go to `System Preferences -> General` and make sure that the option "`Automatically hide and show the menu bar`" is checked.



NOTE: Windows machines at some sites have reported that the screens have been flipped. The Psychopy scripts seem to pay attention to the display number (when you press "Identify"), but we haven't found a good way to change the identity of the monitors aside from changing the video output port to which they are connected.

Computer hardware

Most modern desktop and laptop computers are suitable for running the tasks, but before running participants please be sure to check your setup. Psychopy runs a Benchmark Wizard providing a report on your computer's graphics card when it is first run, which can also be accessed from the `Tools --> Benchmark Wizard` menu. Ideally you should see very few or no dropped frames.

As recommended, be sure to **turn off the internet and close background programs such as Dropbox or web browsers when collecting data**. For more information on recommended graphics cards, see the Psychopy Install page.

Button Box Mapping and Scan Trigger Setup - siteConfig.yaml

The button mappings and scanner trigger key used to sync the onset of scan acquisition with the start of the task are configurable, and are set in a plain-text file called `siteConfig.yaml`.

To open the config files, right-click and select "Open With...", then choose a plain-text editor (e.g. `Notepad`, `Atom`, `Sublime`, `Text Wrangler` or `Psychopy Coder View`). If those suggestions aren't available from the right-click drop-down menu, open your editor first and then use `File -> Open` (or drag-and-drop directly into Coder window).

> Warning! Be careful **avoid** a full word processor like `MS Word` (or even `TextEdit` on a Mac), because in some cases it can corrupt the file by adding "pretty" formatting. We recommend the text editor built into Psychopy ("Coder" view) if you don't already have a preferred editor.

An example looks like this:

```
keys:
  pointerFinger: ['1']
  middleFinger: ['2']
trigger: '7'
```

For example, if your site uses a button box with red and blue buttons that are received as 'r' and 'b' keypresses, and the scanner sends an equal sign for a trigger instead of a 7, the siteConfig.yaml would look like this:

```
keys:
  pointerFinger: ['1', 'num_1', 'r']
  middleFinger: ['2', 'num_2', 'b']
trigger: 'equal'
```

Any key in the list will be accepted as a matching response, so you could press any of '1', 'num_1', or 'r' to indicate the pointer finger was pressed.

The default trigger key is '7' and comes in as a USB keypress; you can advance the experiment when it's waiting on the "Get Ready!" screen either by pressing the trigger key (default: '7') on the keyboard, or by plugging in a USB trigger from the scanner. It is possible to adapt the tasks for a serial TTL trigger, but this is not built in and you'll have to change the tasks yourself.

Setting Coder to Read-Only

Once configured to your liking, the `Coder` window should be set to Read-Only mode in order to avoid accidentally changing the tasks. To do this, go to `Psychopy Menu -> Preferences -> Coder Tab` and select the Read-Only box.

Psychopy must be Quit and Restarted after updating before the change will take effect.

Running the tasks at each visit

Practice Scripts

We have provided a **pre-instructions** script to introduce participants to the testing environment. Immediately before the first session (before going into the scanner and as close to scanning time as possible), participants should be shown the `Pre-instructions` script which includes general information about how to perform the tasks: pictures of the button box they will use and which fingers they need to use to press the buttons, the meaning of the fixation cross that they will see throughout the tasks, etc.

Immediately after these general instructions, participants should be introduced to the specifics of the task by running the `[TaskName]_Practice.py` script, which follows the general pattern:

i. *Intro instructions* (task to perform, how to respond (presses / button mapping, different conditions they will see).

ii. 2-5 *Example trials* / condition with feedback (e.g. to ensure that they press at the correct time with the correct fingers, and generally understand the requirements of the task).

iii. *Dress rehearsal* - Short 1-2min practice without feedback (identical structure and timing to the in-scanner task)

Scan Scripts

The scan scripts are called `[TaskName]_Scan.py`. Whenever you run the script for one of the tasks, they will first show one or two slides summarizing that task's instructions as a reminder to the participant, and then they will show a 'Get ready!' message that will wait for the scanner trigger to start the task.

To run an Experiment Script (each time)

Open Psychopy (from Applications or the Start Menu).

Choose `File -> Open`, navigate to the task directory, and open the `[TaskName]_Scan.py` experiment file.

Click the large Green "Running Man" icon to run the experiment. A small window will pop up to input participant ID and other values - see the Input Option Glossary below for more detail.

Scripts that have multiple runs on the scanner are programmed to run all runs consecutively, so they only need to be started once. The number of runs that will occur is controlled by the `nRuns` value in the dialog box and can be adjusted (e.g. if you had to restart a run; see below for more detailed information).

Once you open a given script, the primary screen will show instructions that the experimenter will read to the participant as he/she moves through the task instructions/practice, while the second monitor will show the actual task that the participant will see.

Input Option Glossary

Whenever possible, input fields for the dialog boxes entered at the start of each task include are labeled consistently between tasks:

age Participant age in years

mriMode Either `scan` or `off`; determines whether the script waits for a trigger at "Get ready!" or whether it continues without waiting. Be careful because this is case-sensitive; entering "Off" or "OFF" will cause the script to incorrectly pause.

nRuns Number of runs to loop through. May be left default unless something broke on the scanner (see Q3 below in the FAQ)

counterbalance Controls which version of stimuli are presented. Available options should be shown automatically.

sessionID Subject / Participant ID - Any string of letters and numbers is acceptable

runNumber The starting scanner run; usually leave this at 1 and change only if you had to quit a run on the scanner and need to start later in the scan series (e.g. re-run Run 2).

Output files

- **wide.csv** : Wide format where each row is a trial, listing onset times, responses, rt's, etc. The most common log file for analysis.
- **design.csv** : Long-format where each row is an event, and the columns are "condition", "duration", "onset", and "run". This is useful for basic analyses (checking collinearity), and gets fed directly into analysis pipelines (based on lyman design file, also similar to FSL's 3-column format (but without a height of 1's column)
- **.log** : A timestamped verbose log of all actions that occurred (e.g. "started showing fixation text", "set shape to 'circle.png' "... This is most useful for debugging and being confident in timings, but less so for analysis.
- **.psydat** : A Psychopy file saved from memory. An original source that is useful for processing in python or recreating the csv.

Advanced Configuration

Visual Angle and Field of View (FOV) Calibration

Stimuli sizes and task layout in the HCP Battery are coded in degrees of visual angle to ensure a uniform experience across all HCP sites. When running at your own site, you may choose to either:

1) match the visual degrees to exactly what was shown in the HCP study by accurately entering the screen width and distance, or
2) use the distance value to arbitrarily zoom the screen in order to maximize visibility.

For either choice, screen configuration is done by editing the `width` and `distance` values in the `monitor` section of the `siteConfig.yaml` file with a text editor (again, we recommend the Psychopy "coder" view). `Width` is the width in centimeters from left to right that is visible from inside the scanner, and `distance` is the distance in centimeters from the participant's eye to the screen when inside the bore.

For example at our site, the viewable width is 41.9cm and the distance is 104cm, so we would replace the initial values with updated ones:

```
monitor:
  width: 41.9
  distance: 104
  screen: 1
```

The task is programmed to fit within a frame that is explicitly drawn in the `ParticipantSetupScreen` task (available as a separate download from HCP). Using either the exact measurements or arbitrary zoom values, you should run this script during setup to make sure that the task will not be cut off.

If using an arbitrary zoom (choice 2) and the frame is too small, increase the distance (for example, from 104 to 110cm), and if it is too large decrease the distance (e.g. from 104 to 94cm). You will likely have to test the distance several times to get the appropriate fit for your screen.

If a stimulus computer or MRI screen changes, don't forget to re-measure and update the monitor settings in the `siteConfig.yaml`.

> Note: You will see the message `Monitor specification not found. Creating a temporary one...` because the task is using its own monitor and not one pre-configured by Psychopy. Please ignore this warning.

Quirks and Gotchas

Here are a few quirks that may help:

1. On a Mac, if you close all the Psychopy windows without quitting the application it may look like nothing pops up when you try to open the app, even though it's already open. To fix this, go to the top bar, click on `View -> Open Coder View` to open a window you can navigate from.
2. Make sure the monitor is connected before running tasks, or both the participant window and RA window will show up on the same screen.
3. You can't open the tasks just by double-clicking on them; you must use `File -> Open` or `Cmd-O / Ctrl-O`

FAQ

Q1) Do I *really* need the specific Psychopy version above?

A1) Sorry, but yes please. We can't guarantee that everything will work if you're not using the right version of Psychopy, so downloading the right one will prevent headaches for everyone. Psychopy installation allows for different runtime versions to be installed side-by-side. Administrator privilege is required to install into /Applications, but the application can be run directly from the desktop on mac if required.

Q2) Is it possible to exit the task halfway through or do I have to wait until it is over?

A2) You can exit a practice or scan script by pressing `<Escape>`.

Q3) What do I do if the scanner had to stop while one of the tasks was running?

A3) If the script needs to be cancelled (e.g. the scanner broke) you should re-run the task script, but you will need to adjust some of the input values for the pop-up. Set the `runNumber` to the current run (e.g. if the scanner broke on the first run, set `runNumber` to 2) and set `nRuns` to the number of remaining runs. You will have to go through the brief reminder instruction screen again. You should enter the subject id exactly the same; output logfiles always include a timestamp and won't be overwritten if you use the same subject id.

Q4) Where are the data output files stored?

A4) When you run any of the scripts for the first time, Psychopy will automatically generate a 'data' folder inside the task's, next to the `.py`, where it will save the data files generated from your session. Data files will be stored in `data/<task>_<subid>_run?_YYYY-MM-DD_HHMMSS_*`, and will be timestamped and labeled with the name of the task, participant ID and run date and time as part of the file name. In this way, logfiles should never accidentally be overwritten because an existing file already exists.

Q5) How is timing recorded? When did the first scan trigger occur?

A5) All `StartTime` and `EndTime` columns in the .csv logfiles are stored in seconds relative to the time that the first TTL pulse (trigger) was received (the `fmriClock` is zeroed when the trigger is received and used to count time for onset timestamps), and the time of the wall clock is recorded in the timestamp of each run's output logfile. The full timing of the script (including the time the trigger was received after the script was started and preliminary instruction screens were shown, as well as drawing of stimuli to the screen and presses received) can be reconstructed in the detailed .log files if required. All `.rt` columns are relative to the start of the stimuli they are paired with (e.g. the go/nogo rt is sec from the start of the shape).

Q6) Where can I find detailed information about the wide logfiles for each task?

A6) Detailed information are provided in data dictionaries within each task folder as `columns-dictionary-<TASK>.{csv|json}`. These dictionaries contain Brain Imaging Data Structure (BIDS)-compatible meta-information ("sidecar json") about each column in the `_wide.csv` logfiles, including units, relative timings, and the interpretation of valid values.

Getting Help

The Psychopy versions of these tasks were created by Erik K Kastman, Cony M Vidal Bustamante, and Leah H Somerville. Questions about the tasks should be directed to the HCP Mailing List: *hcp-users@humanconnectome.org*. We will make a best-effort to help, but the tasks are provided as-is with no guarantee of support!

Full Installation Instructions

For Mac:

Open the disk image and drag "Psychopy" to the "Applications" folder.

Rename "Psychopy" to "Psychopy-1.83.04". This will ensure that you're using the same version even if a newer one gets installed.

When launching Psychopy for the first time, your Mac might give you a pop-up window warning you that this is an application downloaded from the internet and that it is not safe for your computer,

so it won't let you open it. To avoid this issue, instead of just double-clicking on the Psychopy application icon to open it, *right-click* on it, and then select Open. The same pop-up window warning will show up, but this time it will give you the option to open the application.

For Windows:

Double-click the installer and follow the commands.

For even more information and recommended hardware, see the Psychopy Installation documentation at: http://Psychopy.org/installation.html.