tACS Experiment Instructions

Step 1: Software Preparation

Installing Python

Download and install Python. The link provided is to the Python installer for Windows.

https://www.python.org/ftp/python/3.2.5/python-3.2.5.msi

If you do not have Windows, you will need to visit the Python webpage and find version Python 3.2.5 for your system

Installing Pygame

Download and install Pygame. The link provided is to the Pygame installer for Windows.

http://pygame.org/ftp/pygame-1.9.2a0.win32-py3.2.msi

If you do not have Windows, you will need to visit the Pygame webpage and find version 1.9.2a0 for your system

Extract test software

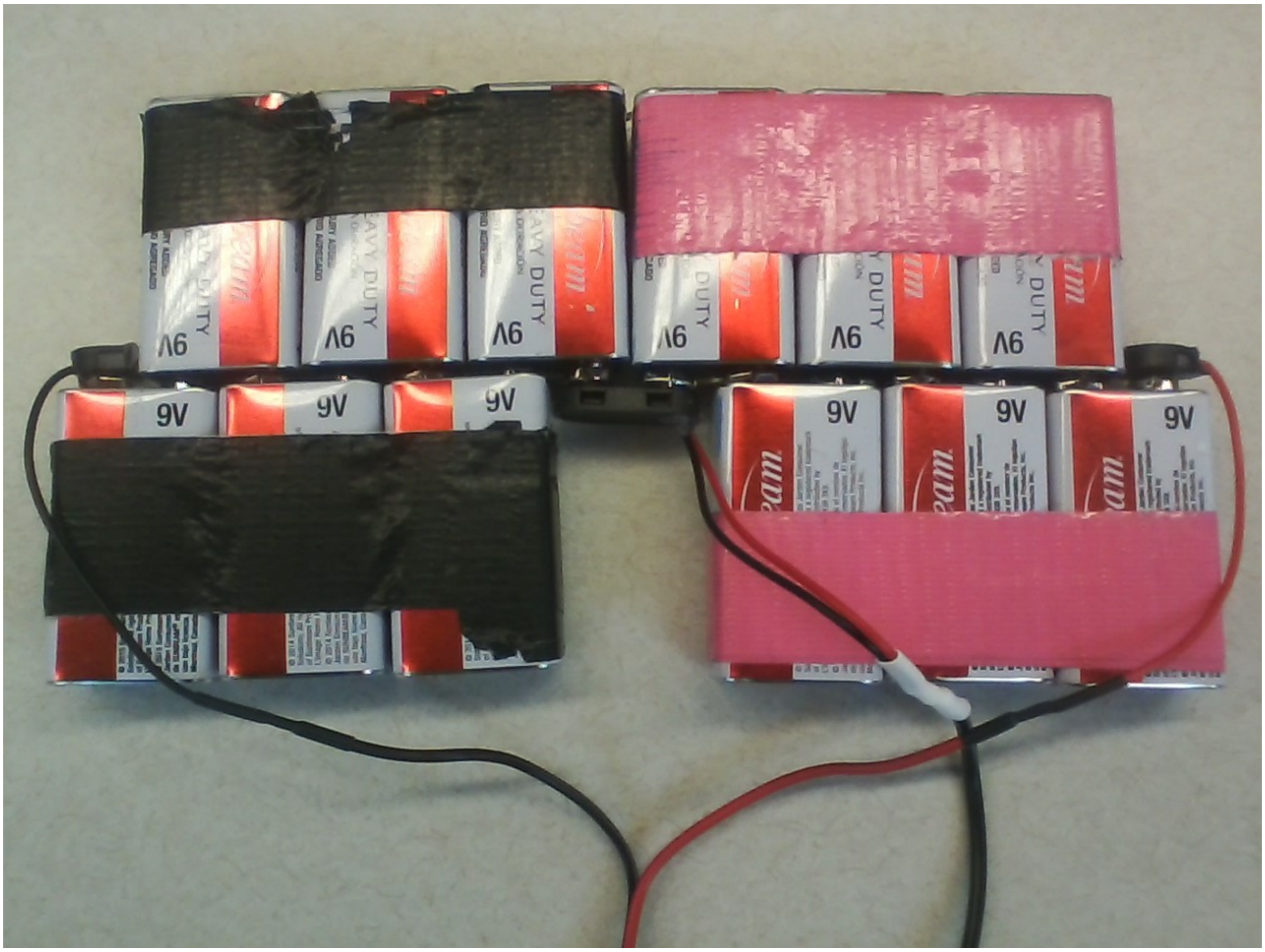
If you have not already done so, download and extract the test software from tACS_experiment.zip (right click > extract all). Please don't open the folder “donotopenbeforetest” before taking the test, at least not in any way that would make the images in it visible to you. You can also find the tACS experiment software and files in the GitHub repository.

Running the tutorial

Double-click on the file “tutorial.py”. If the installation of Python and Pygame has gone properly, a window will appear with a grey box on the left and cheerful blue text on the right. If tutorial.py fails to open properly, email me, and I will help troubleshoot the problem. Troubleshooting fills my soul with unbounded joy to an extent never experienced by mere mortals ever before.

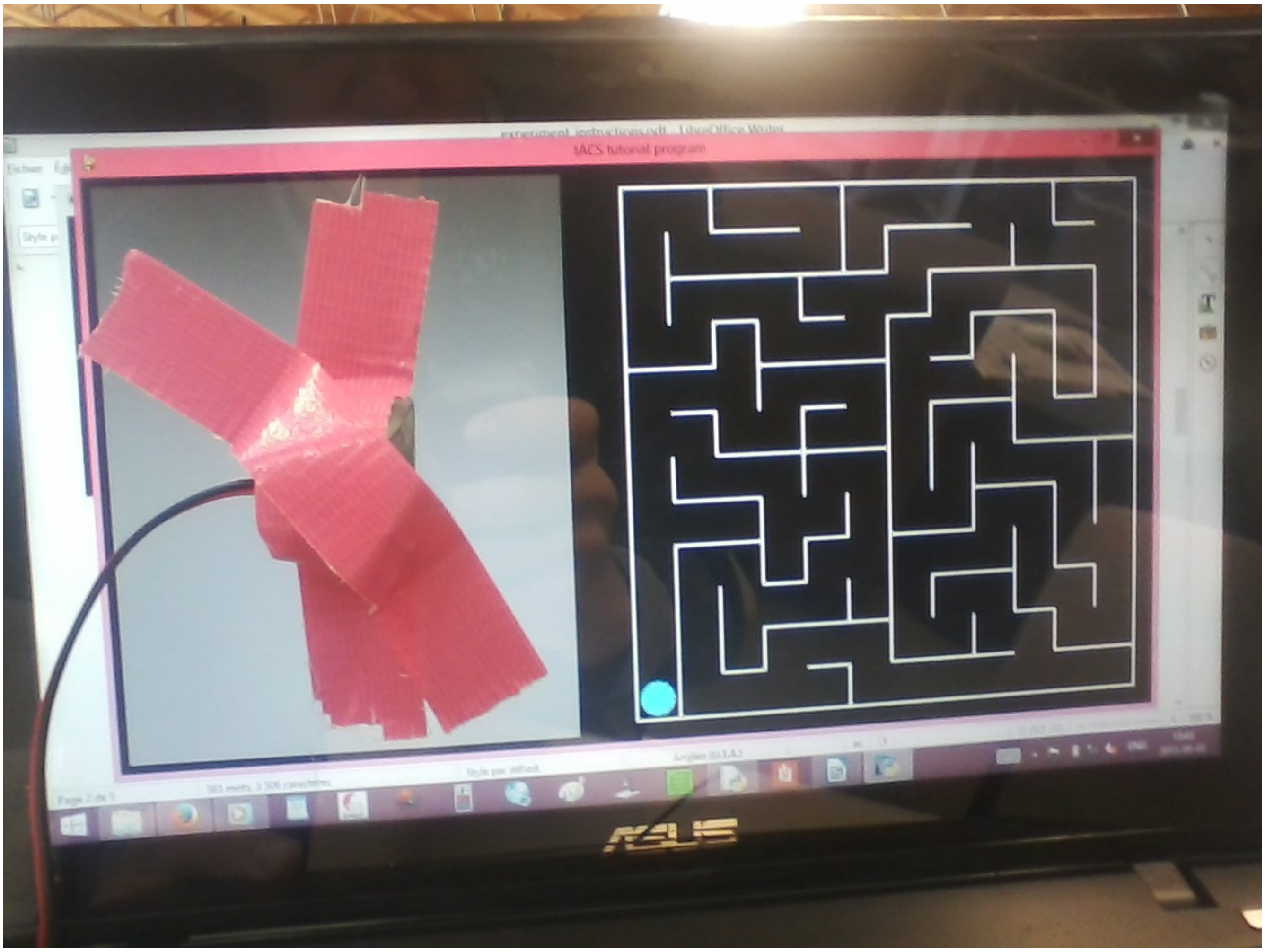
Step 2: Hardware Preparation

Batteries



The tACS device requires twelve 9V batteries. Connect them to the device as shown. Touching the ends of the batteries on opposite sides of the bank with the center connector attached may result in electric shock.

Sensor Attachment



Attach the photosensor to the screen as shown in the picture.

Calibration

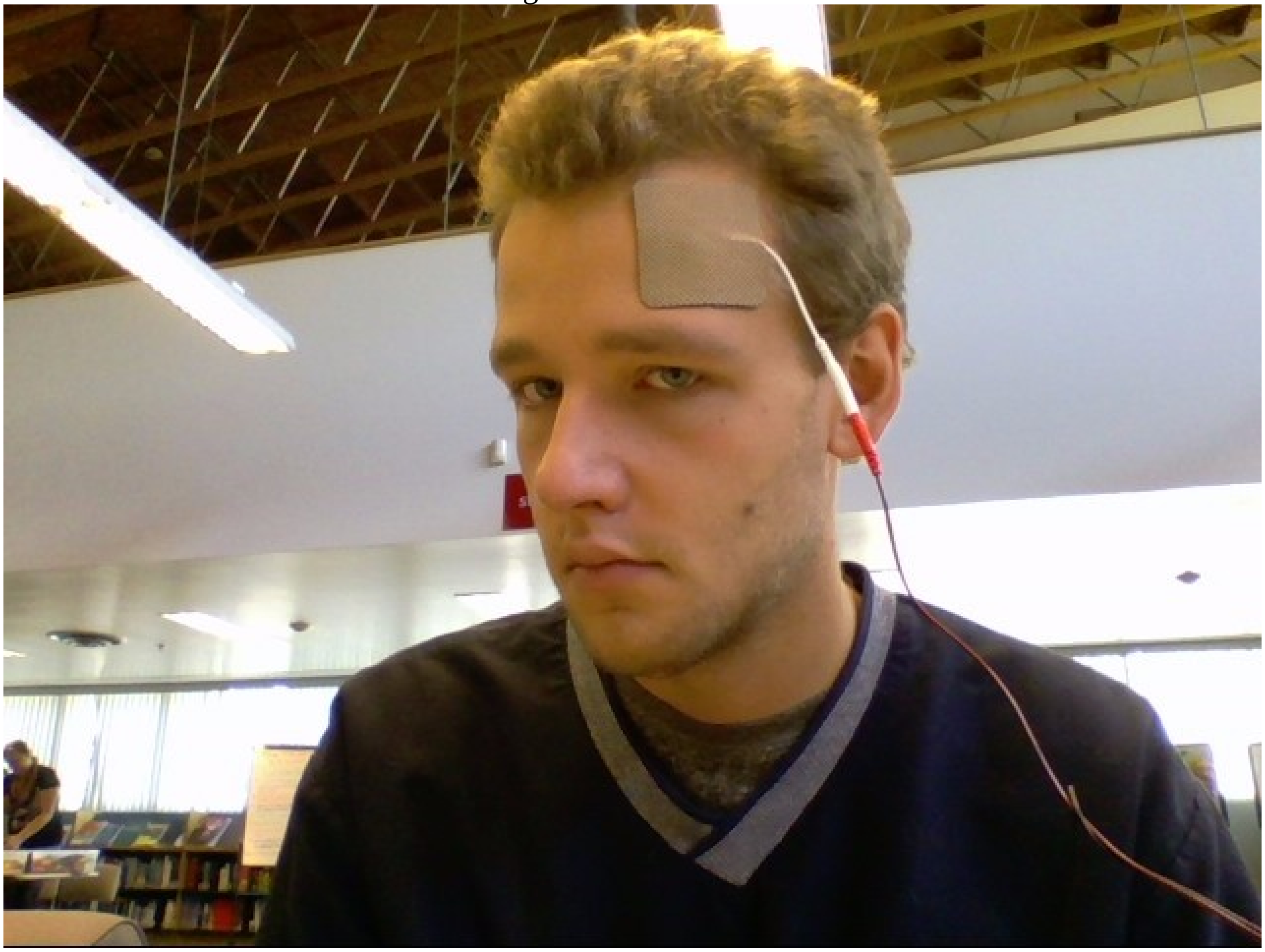
After you've launched takemazetest.py (a few steps down), you'll need to calibrate the device. The tACS device is controlled by screen brightness, and every screen has a slightly different brightness. To calibrate, with the device set to CAL

Step 3: Wetware Preparation

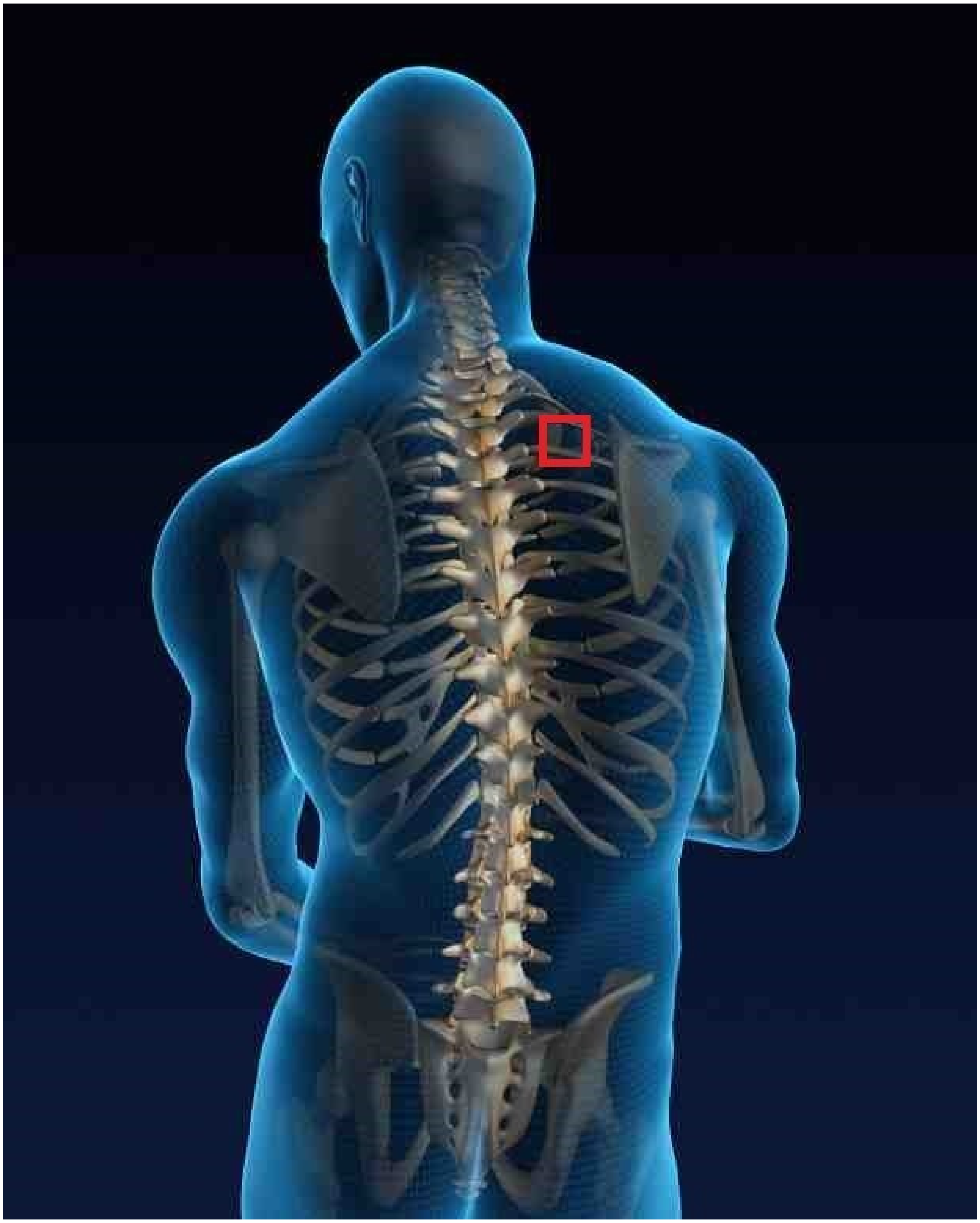
Cleaning the skin

Clean the test subject's forehead and upper back using alcohol or soap and water. This prepares the skin for adhesion of the gel electrodes.

Adhering electrodes to the human



Adhere one electrode to the left of the forehead as shown in the picture.



Adhere the other electrode to the back at the location of the red square in this image.

Note: make sure the electrode on the head is firmly attached! If you are re-using electrodes, you may want to have test subjects wear a headband to make sure the electrode contacts the skin across its entire area. Partial disconnection of this electrode will cause current to flow through the skin with a much higher current density than intended, which can be painful.

Attaching electrodes to tACS device

Connect the electrode on the forehead to the RED 2mm plug of the tACS device. Connect the electrode on the back to the BLACK 2mm plug of the tACS device.

Step 4: SCIENCE!

Avoiding corruption of test data

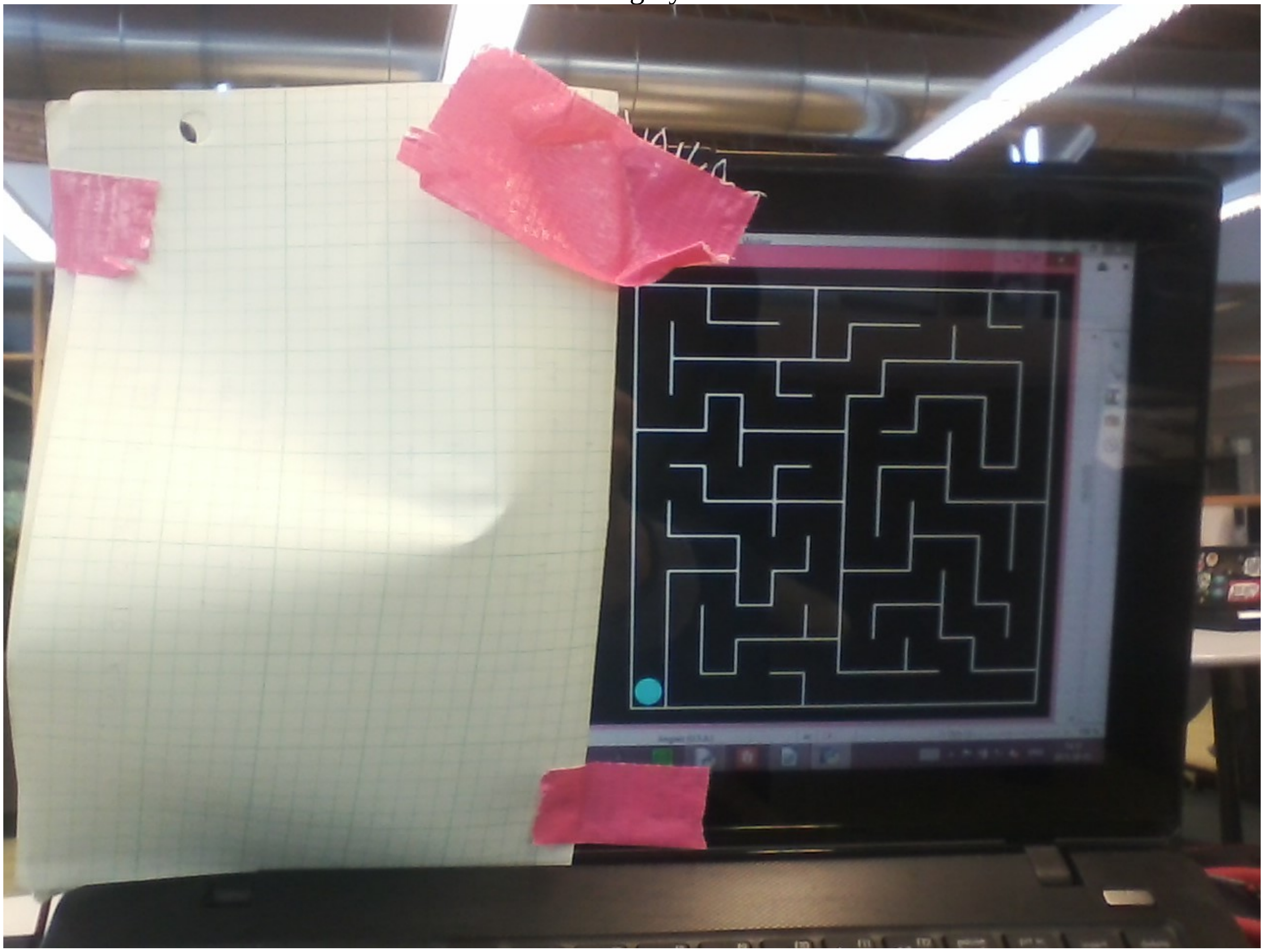
Make sure that test subjects do not see the mazes in takemazetest.py (contained in the folder donotopenbeforetest) before taking the test. Each test subject can only take the test once. Administer the test in an environment free from distractions.

Running the tutorial

Send each test subject through the tutorial (tutorial.py) before running takemazetest.py; this ensures that test subjects understand how to move the circle, and that their objective is the upper-right square of the maze.

Running takemazetest.py

Hide the grey box!



Run takemazetest.py and follow the instructions on the screen. Make sure that the photosensor is centered in the grey box and that the grey box is completely obscured from the test subject. Calibrate the tACS device by adjusting “coarse” and “fine” controls such that both the red and blue LEDs in the tACS device are as dim as possible.

Collecting the test data

After you're done running takemazetest.py, a file will be generated in the folder containing takemazetest.py. This file will have a name that looks like “TESTDATA_1441254355.801_.txt” but with different numbers. For those wondering, the number is how many seconds have passed since January 1, 1970, which produces a unique identifier for each test subject while maintaining anonymity. One file like this is generated each time a battery of 30 tests is completed. Email all the data files generated from tests to me. Send me only one test data file per subject! If you want to play with the test software more, make sure you've separated the data files from your actual tests (done on people who've never seen any of the mazes before) first, or you may get them mixed up.