CS/NEUR125 Brains, Minds, and Machines

Lab 2: Human Face Recognition and Holistic Processing

Due: Wednesday, February 8

This lab explores our ability to recognize familiar and unfamiliar faces, and the potential use of "holistic" processing to perform this task. We will first discuss possible hypotheses for the outcome of the Cambridge Face Memory Test (CFMT) and Famous Faces Memory Test (FFMT) for the class, and how we might visualize and analyze the data to explore these hypotheses. You will then use MATLAB to analyze the class data (combined with some additional data from a published study). Next, you will consider the possible implications of the so-called *Thatcher Illusion* for face processing, and create a fun demonstration of this illusion in MATLAB. Finally, you will run a simple perceptual experiment that combines the *Face Inversion Effect* and *Face Composite Effect* to explore the use of holistic information when processing faces.

These activities introduce some basic aspects of the design of perceptual experiments. You will also learn how to create a MATLAB *script* to store a sequence of code statements, and new commands to plot and analyze data, and to load, display, and manipulate images. Code and image files for some of the activities will be downloaded from the CS file server. At the end of lab, you will upload your script file to your individual account on the CS server.

To begin, partners should create a copy of this Google document, modify the title of the copy to include both names, and share the copy between the two partners, as we did in Lab 1. (For a review of the process, see the Working with Google Docs for Lab Handouts webpage.) Questions here that you should try to answer during lab are shown in blue, and those that you could answer later are shown in purple.

Start MATLAB on the lab Mac that you are sharing. We will demonstrate how to **use the Fetch program to download a folder of code and image files, named Lab2_files,** from the CS file server to the Desktop on your Mac. You will be able to access this folder through your individual account on the CS server. For later reference, details of this process for both Macs and PCs can be found on this webpage.

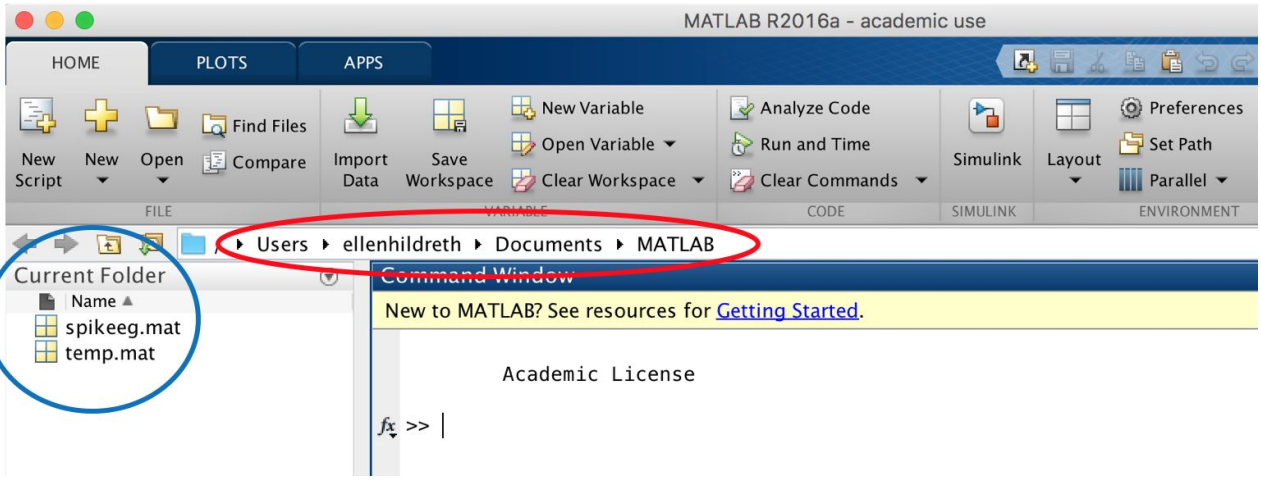
I. Recognizing Familiar Faces and Learning New Faces

We can visualize data from the CFMT and FFMT experiments with a scatter plot in MATLAB, and also determine the average performance on each task and spread in performance over a group of participants. Your code for this analysis will all be placed in a single MATLAB **script file**.

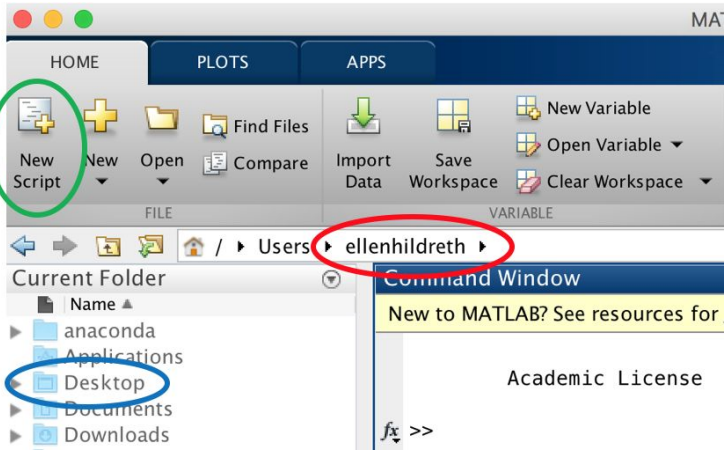
Change the Current Folder and create a MATLAB script

In Lab 1, you entered MATLAB commands one by one in the Command Window. Using a text editor that is built into MATLAB, you can create a *script file* that contains multiple commands that

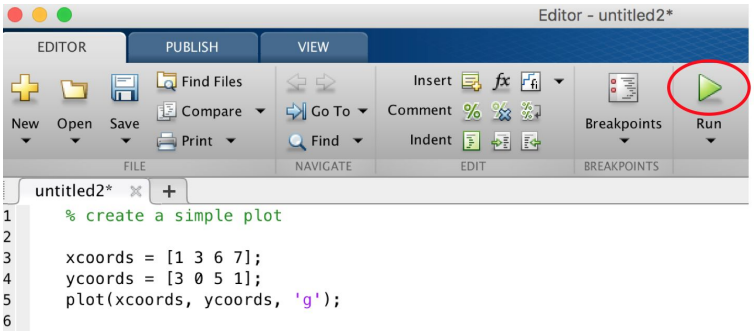
can be executed all at once. Before creating a script, we will set the **Current Folder** in MATLAB to the Desktop of your Mac. The Current Folder is the default folder where MATLAB stores new code files that you create, and finds existing code files that you want to execute. MATLAB initially sets the Current Folder to a folder named MATLAB inside the Documents folder on your Mac. The path to this folder is displayed above the Command Window (circled in red below), and its contents are displayed on the left (circled in blue) - the contents of this folder may initially be empty:

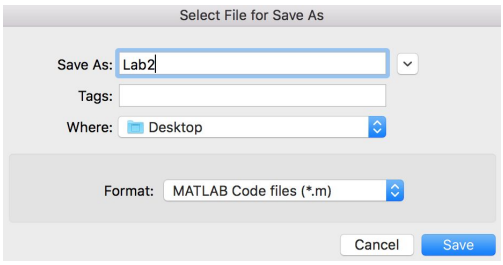


For this lab, you will set the Current Folder to the Desktop on your Mac. Click on your username in the path (circled in red below), and then double-click the Desktop folder on the left (circled in blue):



To create a new script, click on the **New Script** icon in the upper left corner of the MATLAB window (circled in green in the above picture), which will open an **Editor** window. Enter a *comment* and code statements to create a simple plot (you can modify the code shown in the picture below). In the example below, the first line (shown in green) begins with %, which indicates to MATLAB that the rest of the line is just a comment that should not be executed. To save and run your script file, click on the green triangle in the menu bar at the top of the window. In the dialog box that appears, enter **Lab2** as the name of the file in the box labeled **Save As:**





Your plot will appear in a figure window, and a file named **Lab2.m** will appear in your Current Folder (the Desktop). MATLAB code files end with **.m** and are referred to as *M-Files*. You can also execute a script file by entering its name in the Command Window:

>> Lab2

Analyze the CFMT and FFMT data

To display a scatter plot of points with the same coordinates that you used in your initial script, replace **plot** with **scatter**, and again click on the green triangle to save and run your modified code. By default, the points appear as small open circles. You can change the size of the points and display filled circles with additional inputs to scatter:

scatter(xcoords, ycoords, 200, ‘g’, ‘filled’)

You can use help or doc in the Command Window to learn more about scatter:

>> help scatter
>> doc scatter

When first displaying a plot of some sort, MATLAB automatically sets the range of values on the x and y axes to encompass the actual range of values in the input vectors of coordinates. You can adjust the range displayed with the **axis** command. The general format for this command is:

axis([*xmin xmax ymin ymax*])

Where *xmin* and *xmax* specify the range of x coordinates you want to display and *ymin* and *ymax* specify the range of y coordinates. For the sample code shown in the above figure, the actual x values range from 1 to 7 and y values range from 0 to 5. The following statement will change the range of values displayed on each axis:

axis([0 8 -1 6])

Add a statement to your script after the scatter command, to expand (slightly) the range of values shown on the x and y axes.

Data from 15 subjects (including students in the class) for the CFMT and FFMT experiments will be written on the board. Modify the code in your script to create two vectors to store the two sets of

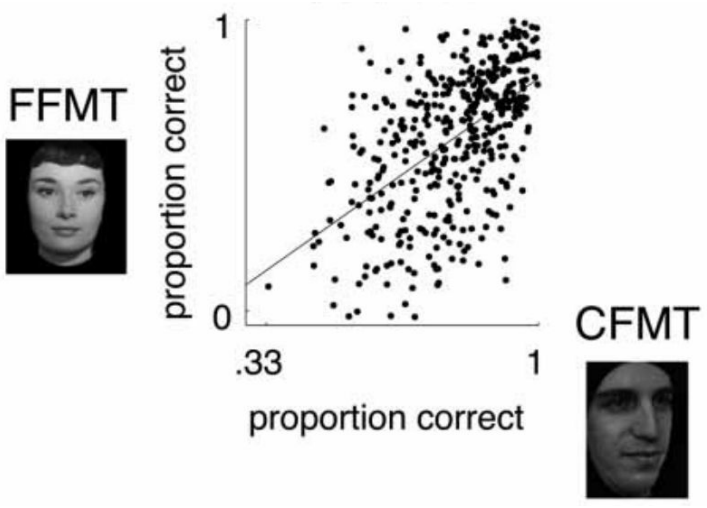
data, using informative variable names for the vectors. Create a scatter plot with the CFMT data shown on the x axis and FFMT data shown on the y axis. Add the axis command to adjust the range of axis values so that there is some white space all around the data points in the plot. Add code in your script to label the two axes with the xlabel and ylabel commands, and add a title. The average performance level and spread in performance (e.g. standard deviation) for each dataset can be computed with the **mean** and **std** commands, for example:

```
avg = mean(data)
stdev = std(data)
```

Add statements to your script to print the average and standard deviation for the CFMT and FFMT data (you can just omit the semi-colon at the end of the statements that compute these quantities, so that their values will be printed in the Command Window when you run your script).

- **Q1. Paste a copy of your figure** into this Google document (as you did in the first lab) and add answers to the following questions:
    - For the CFMT data, what is the **range** in performance observed (i.e. the minimum and maximum performance), the **average** performance level, and **standard deviation** in performance level?
    - For the FFMT data, what is the **range** in performance observed, the **average** performance level, and **standard deviation** in performance level?
    - Viewing the data as a whole, does performance on the two tasks appear to be correlated, i.e., do people who perform at a higher level on one task also appear to perform at a higher level on the other?

- **Q2.** The figure below shows a scatter plot of data from the CFMT and FFMT tasks for over 400 subjects, published by Wilmer et al. (2012). Although there is a large spread in performance on both tasks, statistical analysis of the data reveals that there is a correlation in performance on the two tasks. The thin diagonal line in the figure below roughly captures the relationship between the two. Reflecting on our discussion at the beginning of lab, why might you expect there to be a correlation between these two abilities?



II. The Thatcher Illusion and Holistic Processing of Faces

In the video that you viewed earlier, What you can learn from studying behavior, Nancy Kanwisher presented the *Thatcher Illusion* as evidence in support of holistic processing of faces. This illusion was first described by psychologist Peter Thompson in 1980. In the original demonstration, a photograph of Margaret Thatcher was altered by flipping her eyes and mouth upside-down. When the altered face is viewed upside down, one hardly notices anything unusual, but when viewing it upright, the face looks like something from a horror movie!

In this problem, you will create your own demonstration of the Thatcher Illusion, with Ellen as the "victim." Along the way, you'll learn how to load, display, and alter a two-dimensional image in MATLAB.

- **Q3.** How does the Thatcher Illusion support the idea that faces are processed holistically? **Hints:** again view the one-minute segment on this illusion in Nancy Kanwisher's video (in the time frame from 4:15 to 5:15) and see the article, The Thatcher illusion: Are faces special? published a few months ago in *The Guardian*.

Add the Lab2_files folder to the MATLAB search path

The Current Folder is the first place MATLAB looks for a code file that you want to run or other files that you want to access. Your Current Folder is now set to be the Desktop of your Mac. For this exercise, we want to access an image file containing a photograph of Ellen that is stored inside the **Lab2_Files** folder on the Desktop. First try entering the following command in the Command Window, to read the image of Ellen into the MATLAB Workspace:

```
>> img = imread('ellen.png');
```

You'll see that this generates an error, because this image is not stored directly in the Current Folder, it is instead inside a subfolder on the Desktop. We can, however, instruct MATLAB to also check the Lab2_files folder for files that we trying to access, by adding this folder to the MATLAB **search path**. To do this, execute the following command in the Command Window:

```
>> addpath(genpath('Lab2_files'))
```

Again execute the above imread statement to read Ellen's image into your Workspace.

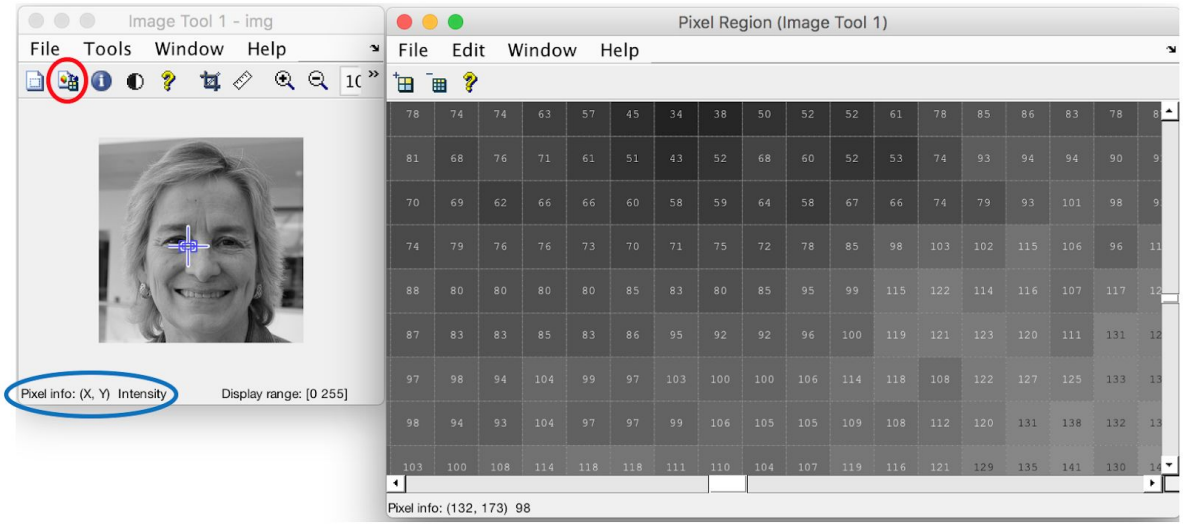
Display an image with imshow or imtool

An image can be displayed with the imshow and imtool commands:

```
>> imshow(img)
>> imtool(img)
```

imshow displays the image in a figure window. imtool allows you to see the location and brightness of individual pixels in the image. As you move the cursor around the image, the X and Y coordinates and image intensity at each image location are shown at the bottom of the window

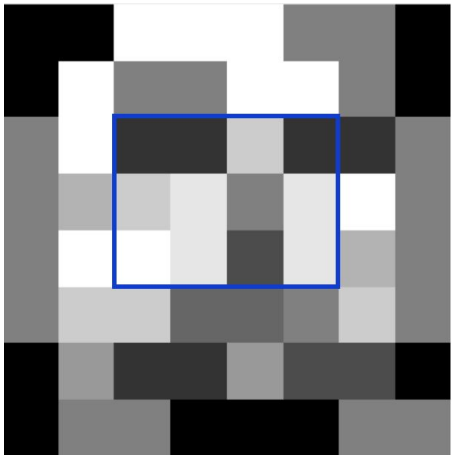
(circled in blue below). Click on the Pixel Region icon (circled in red along the menu bar at the top of the window) to view more pixels, as shown below on the right.



To create a demonstration of the Thatcher Illusion, we need to alter Ellen's photo and display the result upside-down. To see how this can be done, we will first do some exercises on the board to illustrate how the content of one or more locations of a vector or matrix can be changed with an assignment statement.

Building on these ideas, consider the following tiny image below, stored in an 8 x 8 matrix named img. The red numbers indicate the indices of the rows and columns of the matrix, and the picture on the right displays the value 0 as black and 10 as white, with shades of gray in between.

	1	2	3	4	5	6	7	8
1	0	0	10	10	10	5	5	0
2	0	10	5	5	10	10	5	0
3	5	10	2	2	8	2	2	5
4	5	7	8	9	5	9	10	5
5	5	10	10	9	3	9	7	5
6	5	8	8	4	4	5	8	5
7	0	6	2	2	6	3	3	0
8	0	5	5	0	0	0	5	5

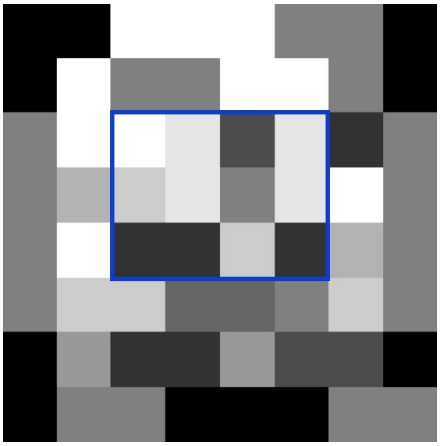


Suppose we want to flip upside-down, the rectangular region surrounded with the thin blue line. We can refer to this region of pixels as img(3:5, 3:6), i.e. rows 3 to 5 and columns 3 to 6 of the img matrix. An upside-down version of this region can be created by referring to the rows in reverse order, i.e. img(5:-1:3, 3:6). In a single assignment statement, we can flip the contents of this region upside-down:

img(3:5, 3:6) = img(5:-1:3, 3:6);

This yields the following matrix and image display:

	1	2	3	4	5	6	7	8
1	0	0	10	10	10	5	5	0
2	0	10	5	5	10	10	5	0
3	5	10	2	2	8	2	2	5
4	5	7	8	9	5	9	10	5
5	5	10	10	9	3	9	7	5
6	5	8	8	4	4	5	8	5
7	0	6	2	2	6	3	3	0
8	0	5	5	0	0	0	5	5



Finally, an image can be flipped upside-down when displaying it with imshow or imtool, by reversing the order of all of the rows, e.g.

imshow(img(end:-1:1, :))

end refers to the last row of the matrix in this case, and the single colon for the columns specifies all of the columns.

If you have an open figure window when you execute the imshow command, MATLAB will just replace the existing content of the figure window with the new image. If you instead want to open a **new figure window**, you can just add the figure command to your code, before displaying:

figure

Note that you can place two MATLAB statements on one line in your script, with a comma separating the two statements, for example:

figure, imshow(img)

You now have the tools to create a demonstration of the Thatcher Illusion:

- Use imtool to identify the upper left and lower right coordinates of two rectangular regions around Ellen's eyes and mouth. **Important note:** at the bottom of the imtool window, the X coordinates refer to the columns of the matrix that stores the image (i.e. the second matrix index), and the Y coordinates refer to the rows of this matrix (first index to the matrix)!
- Add code to your Lab2.m script file to implement the following steps:
    - load the ellen.png image using imread and store it in a variable named im1

- create a copy of im1 named im2 (this can be done with a simple assignment statement, im2 = im1)
- add two assignment statements to alter im2 so that the two rectangular regions around the eyes and mouth are flipped upside-down
- add the figure command to open a new figure window and display im1 upside-down
- add the figure command to open a new figure window and display im2 upside-down
- add the figure command to open a new figure window and display im2 upright

- **Q4.** When you have completed this exercise, add some comments to your script file and copy-paste the contents (code and comments) into this Google doc. Please also **paste the upside-down version of Ellen's altered face** into this document.

Upload the Lab2.m script to the CS file server, to the individual accounts for both partners

Use Fetch to connect to the individual account on the CS file server for each partner, and drag the **Lab2.m** code file from the Desktop of your Mac to the **cs125** subfolder in each account.

- **Q5.** When you are ready to submit your final code file, **drag the Lab2.m code file into the drop subfolder** in your individual account on the CS file server. Mike and Ellen will then be able to access your code file directly.

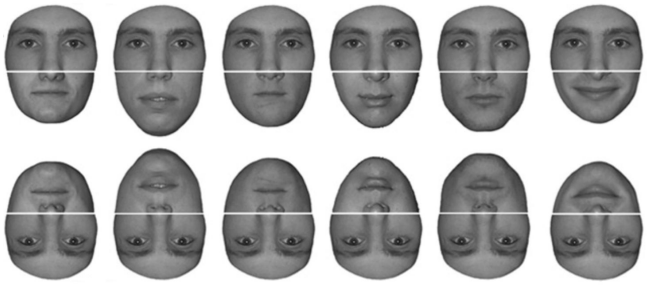
III. Combining the Face Inversion Effect and Face Composite Effect

In Nancy Kanwisher's video and in class, two perceptual effects were used to support the idea that we use holistic information when processing an image of a face, the **Face Inversion Effect** and the **Face Composite Effect**. The figure below illustrates the Face Composite Effect using unfamiliar faces (from Rossion (2008)):

In all of these face images, the top portion of the face is the same. Within each row, the bottom halves of the faces are different. When the same top half is *aligned* with different bottom halves (top row), the top halves sometimes appear different, but when *misaligned* (bottom row), it becomes more apparent that all the top halves are the same.

The basic Face Inversion Effect refers to the fact that recognition is disrupted when a face is inverted, more than when other types of objects are inverted. Perceptual experiments by Rossion (2008) suggest that the Face Composite Effect disappears when faces are inverted, as in the figure

below, where it is more apparent in the inverted (and aligned) faces (bottom row) that the half of the face containing the eyes (now at the bottom) is the same in each image.

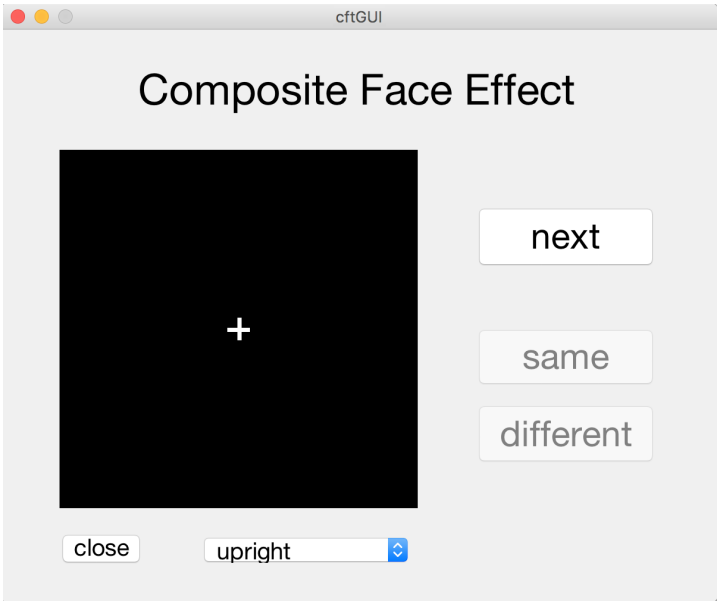


In this activity, you'll run two blocks of a perceptual experiment to experience (hopefully) this phenomenon yourself. Each block will take about 5 minutes to run, and each partner will complete the experiment on a different Mac in the classroom.

To run the experiment, enter the following command in the Command Window:

>> cftGUI

(The cft part of the name stands for composite face test, and GUI is an abbreviation for graphical user interface.) A window will appear that looks like this:



Using the pull-down menu below the dark display area, select **upright** or **upside-down**. We'll divide the class into two groups - one group will run the upright condition first and then the upside-down condition, and the second group will run the upside-down condition first. The **next** button is used to initiate each trial of the experiment. Before clicking **next** for each trial, focus your eyes on the cross that appears at the center of the display area. During each trial, two face images will appear for a brief time, one after the other. The **same** and **different** buttons will then become active and your task is to indicate whether the half of the face containing the eyes was the same for

the two face images, or different. For the **upright** condition, this will always be the top half of the viewed face. For the **upside-down** condition, this will always be the bottom half of the viewed face. In half the trials, the top and bottom halves will be aligned, and in other trials, they will be misaligned. A total of 80 trials will be displayed, whose order is randomized for each viewer.

When you complete a block of the experiment, all of the buttons on the window will become inactive except for the close button and your results will be printed in the Command Window and also printed in a text file that you can view. The name of the text file will be of the form **dataUpright#.txt** or **dataUpsideDown#.txt** where the # symbol is replaced with a number. If you run the same condition more than once, a new text file will be created with this number incremented. Click the **close** button to terminate the program before running the experiment again.

The printout that you see in the Command Window and text file will have the following form:

```
>> cftGUI
Results for trials with same top part of face (eye region) in upright
condition
Experimental run number 1 for this condition
Total number of *same* trials: 40
Percent correct for aligned trials: 60.0 % correct
Percent correct for misaligned trials: 90.0 % correct
Difference in performance, misaligned - aligned: 30.0 %
```

The results reported here only include trials where the halves of the two faces that contain the eyes are **actually the same**. Half of these trials were shown with the two parts of the face aligned, and half of these trials were shown with the two parts misaligned (there were 20 trials for each of these two conditions). The printout shows what percentage of the aligned and misaligned trials were correctly identified as having the same eyes. It also shows the **difference in performance between the misaligned and aligned conditions** (i.e. percent correct for misaligned trials *minus* percent correct for aligned trials)

- **Q6.** Based on our discussion of the Face Inversion and Face Composite Effects in class, what would you predict for the results of this experiment, and why would you make this prediction?

- **Q7.** For each partner, report the ***difference*** results in the table below, for the upright and upside-down conditions. ***You do not need to report your percent correct from the separate aligned and misaligned conditions, just report the difference between the two that is printed out.***

	Partner 1	**Partner 2**
Upright (misaligned - aligned)		
Upside-Down (misaligned - aligned)		

- **Q8.** Are the results that you reported in the table consistent with the prediction you made in **Q6**? If not, can you think of reasons why your particular data (from this somewhat limited version of the experiment) might not fit the pattern you expect?

References

Rossion, B. (2008) Picture-plane inversion leads to qualitative changes of face perception, *Acta Psychologia* 128, 274-289

Wilmer, J. B., Germine, L., Chabris, C. F. Chatterjee, G., Gerbasi, M., Nakayama, K. (2012) Capturing specific abilities as a window into human individuality: The example of face recognition, *Cognitive Neuropsychology* 29 (5-6), 360-392

Cambridge Face Memory Test:

71 75 91 84 88 92 80 89 66 89 68 74 81 72 67 92

Famous Faces Memory Test:

6 7 12 10 7 10 8 13 6 12 6 8 7 10 9 14