Instructions

The package consists of 5 main m-files. I'll describe the use of them separately. But first something about where to put your files. You can put the main folder of the gauge figure software anywhere you want. *This folders now serves as the 'root'.* You have to select this folder in Matlab to start running the programs. All stimuli and observers data is stored in the folders stimuli\ and observers\, see below on how to use them.

1) parameters.m

Here are some global variables used as parameters so you don't need to look in the code to change some small settings. It could be that I forgot a parameter you like to set every now and then, so put this parameter manually in the parameter file, like the others and replace the parameter name in the code.

2) >>contour('stimulusname')

Lets you define the contour in an image. But FIRST you need a picture, and put it in a folder as follows:

1. Get some picture and downsize it if either the width or height is larger than your screen. (i.e. if you have a picture of 3200x2000 pixels and your screen is 1600x1200 you may want to resize the picture to 1600x1000, but smaller is also fine)

2. Create a folder within the stimuli\ folder and name it *stimulusname*. Save your picture in this folder as *stimulusname.tif* (with one 'f', so no 'tiff'). For example, the file path could be: stimuli\shape\shape.tif. *stimulusname* can be any name you want.

The Psychoolbox program will let you set three kinds of contours, subsequently:

- **Outer contour creation** Now you can start the contour program by typing in the commandline >>contour('stimulusname') (use the '' to indicate it is a string, omit the tif extention). You'll see a red dot, which is the mouse position. Click anywhere on the outer contour and start clicking all subsequent points along the outer contour. If the mouse position is within some threshold range of your starting point, the starting point will turn green which means that if you click, the contour will be closed.

- **Hole contour creation**. After finishing the outer contour you will be asked whether there is a hole in you pictorial surface. If yes, press 'y' and define the hole contour just as you defined the outer contour. If no, press 'n' and continue.

- **Inner contour creation** Now you can automatically start defining inner contours, if you want. If you're ready: press 'q'. You may want to move the mouse now and then because sometimes the program is waiting for a mouse to move before it continues.

3) >>triangulation('stimulusname')

Lets you define the triangulation within the contour. With this program you'll define the sample positions where the gauge figure will appear. You can change the position of the vertices my moving the mouse (move right/down as a starter, after that it is periodic anyway).

With the arrows you can increase and decrease the triangle size. Upper arrow increases size and decreases the number of vertices. Nothing to be done about this. When you're finished press 'q'. You will be briefly shown the final result, including the barycentra, which are the actual sampling points. There will also be a picture of the triangulation in your stimulus folder: triangulationpicture.tif.

4) >> experiment('observername','stimulusname')

This is the actual experiment. Before running the experiment you need to do some file management. First, for each observer create a folder in the observer\ folder, e.g. observers\john\. Within this folder, create a folder with the exact same name as the stimulus name, e.g. observers\john\shape\. You can start the experiment by typing in the command line:
>> experiment('observername','stimulusname')
The program will start, using a randomized order of the barycentra. The probe can be controlled with te mouse, click left when satisfied and proceed to the next trial. Data will be written to a file that has has the date and time in its name, automatically. So you'll never overwrite your data if you accidentally restart the same experiment. But the data is also saved as 'data.dat', which will be overwritten. The reason is to make it easy for the last program to find the data file. If anything goes wrong, you can always duplicate the original file (with time and date) and name it 'data.dat'. Duplicate it before renaming since the 'data.dat' file is vulnerable to overwriting.

5) >> reconstruction('observername','stimulusname')

This file will do the reconstruction for you, exports 3D vertex points, and gives a 3D plot. It will look for the 'data.dat' file in the folder /observers/observername/stimulusname/. The program will carry out the reconstruction algorithm. The main output will be the 3D vertices, 'vertices3d.txt', which can be used for the analysis. It also gives a nice plot of the 3D relief superimposed on the 2D stimulus. You can rotate this plot with the mouse.

Other files

There are some extra m-files in the 'functions' directory. Feel free to modify these but since this is of not much general use, I do not explain them here.