**Coding Challenge**

**Task 1**

You are provided with a real, though quite simple, experiment code ( so called TRST for Triangle Rotated Stimulus Test ).

1. Open main.m, run it and go through all the trials. How much points you were able to earn? How easy/difficult it was earning points?
2. What are your observations about the program? Was it easier to interact with? What can you improve with the code?
3. Introduce your improvements. For example, if you run the program on two screens you might find the central triangle difficult to rotate. Why is it so? What can you do to improve it?
4. The user results file and log file overwrites each time you run the main.m (for given date). Please modify the code to prevent overwriting of data files.
5. Each time the code generates a new taskmap for any subject except for subject # 0. However, if the taskmap for given subject already exists it does not allow to proceed with this subject within the same session throwing the following error:

Error using initSubjectInfo (line 57)

ERROR! Result data file already exists! Choose a different subject/session number.

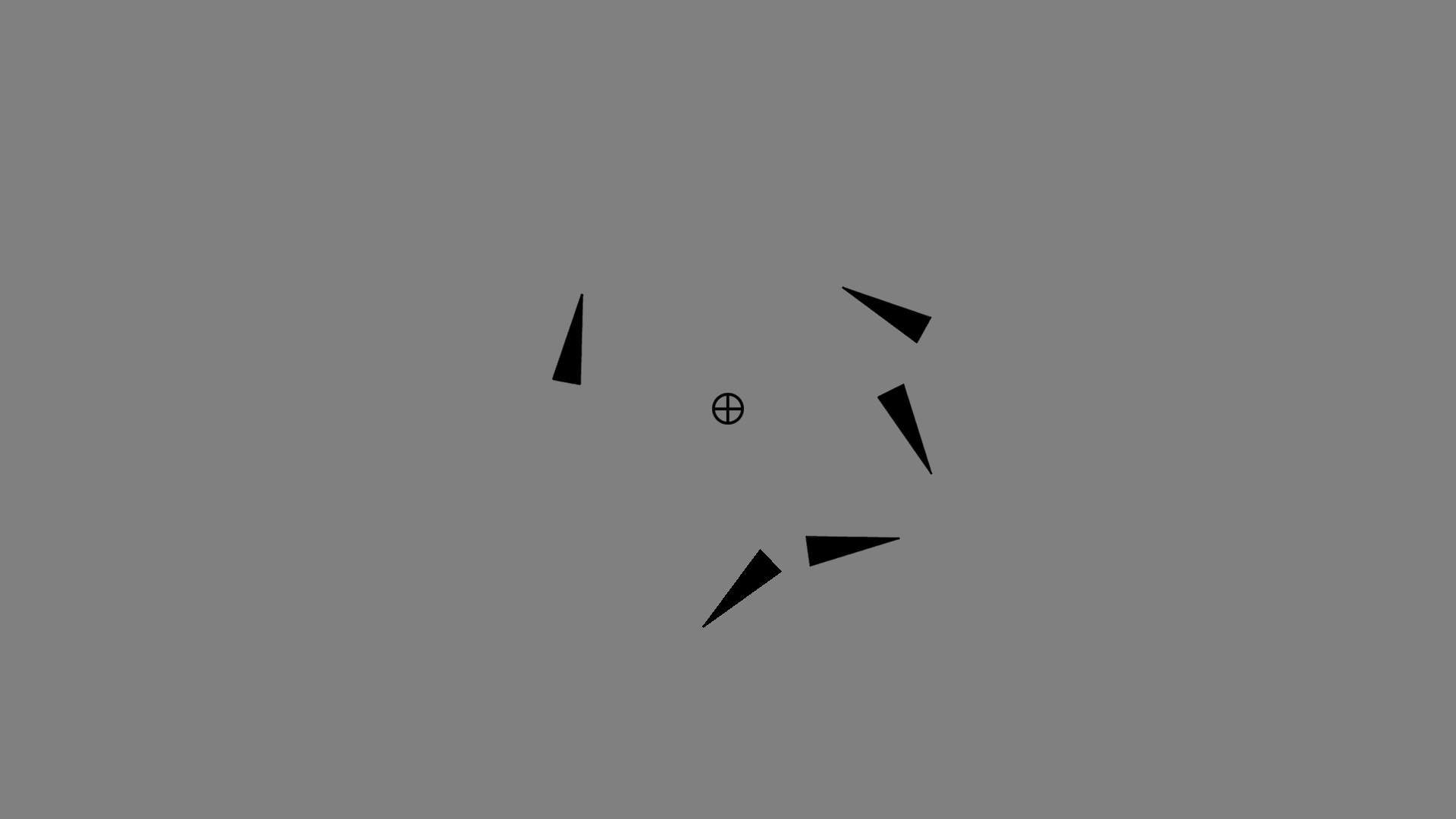
What can you do to allow to run the subject multiple times for one session? The experimenter must receive a warning that subject.

1. The probe triangle orientation is selected randomly. However, in several cases it overlays the stimulus triangle and the subject has nothing to do but just to submit the response without rotating the triangle. Modify the code to ensure the difference btw stimulus and presented probe triangle orientations no less then 30 degrees.
2. Currently the task consists of 10 trials. However, if the number of trials increases we need to let the subject to rest after some number of trials completed. Please introduce this functionality into program. You can split program by block, say 2 blocks with 5 trials each and ask the experimenter to type the number of block he wants to run (1 or 2). If the block has been already run, the warning pops-up asking the experimenter to confirm his/her intention to re-run the block again. Save results of consecutive runs of the same block into separate files to avoid overwriting. Make the number of blocks modifiable, but each time ensure there is equal number of trials in each block.
3. In terms of interactivity, is it correct to ask the subject to press a space bar to start or to finish the experiment while still using mouse for responding? Write two versions of code – for responding with mouse only and with space-bar only.
4. Modify the code to allow the experimenter to exit program any time when ‘Q’ key is pressed. In this case data for previous trials must be saved.
5. For some MEG and fMRI experiments we use a buttonbox to submit the responses (buttons 1-4) and a dial for rotations (assume ‘t’ key – rotates clockwise, ‘b’ –key counterclockwise). Modify the code to allow subject to select and submit the responses using buttonbox and dial. Assume we are performing fMRI task. Allow the experimenter to initialize the scanner by pressing ‘5’ key (in some modifications it is ‘~’ key, please also take it into account). Please make a separate screen informing the subject that scanner initialization is in process.

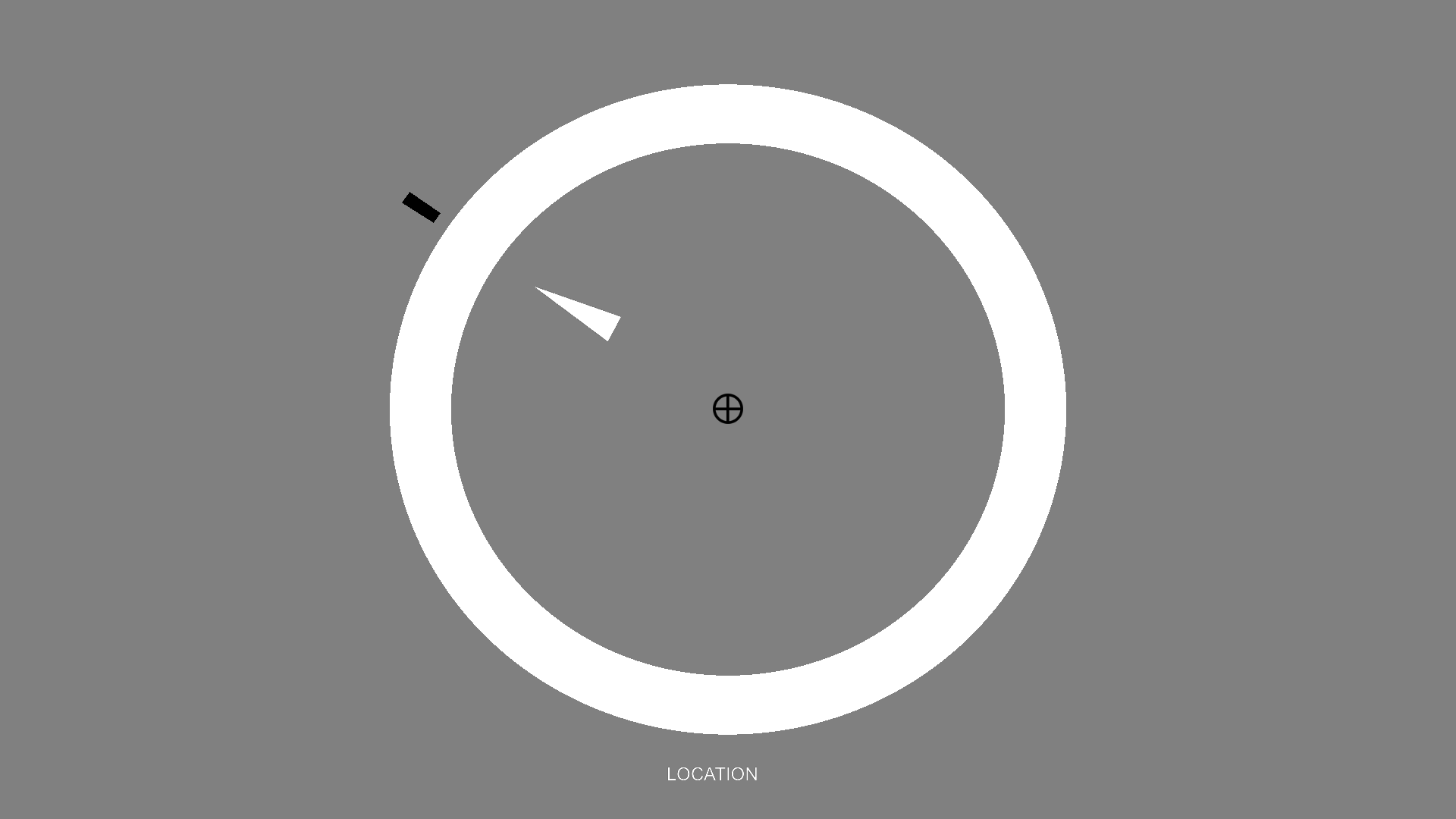
**Task 2**

There are different ways to measure the working memory capacity. The widely accepted one is closely related to the notion of “set size” – the number of items stored in the working memory. Currently our code asks to remember only one stimulus.

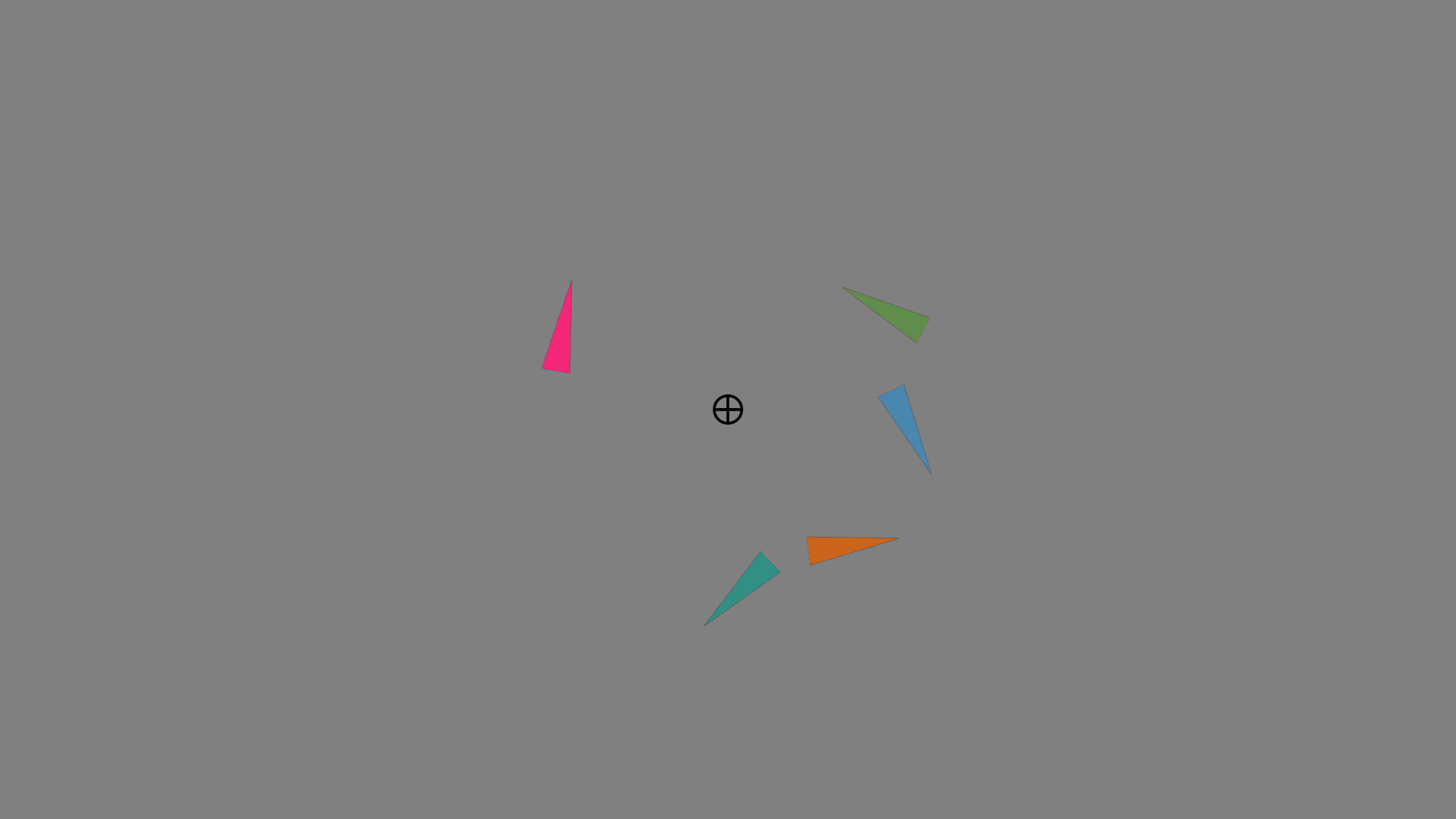
1. Modify the code to remember 5 (n, in general) differently-oriented stimuli, placed in randomly selected 5 main orientations (say, we have selected following 5 out of 12 main orientations [30, 90, 150, 270, 340]. The triangles must be placed in these orientations plus some jitter between [-10,10] degrees. Please ensure the triangles are not overlapping)



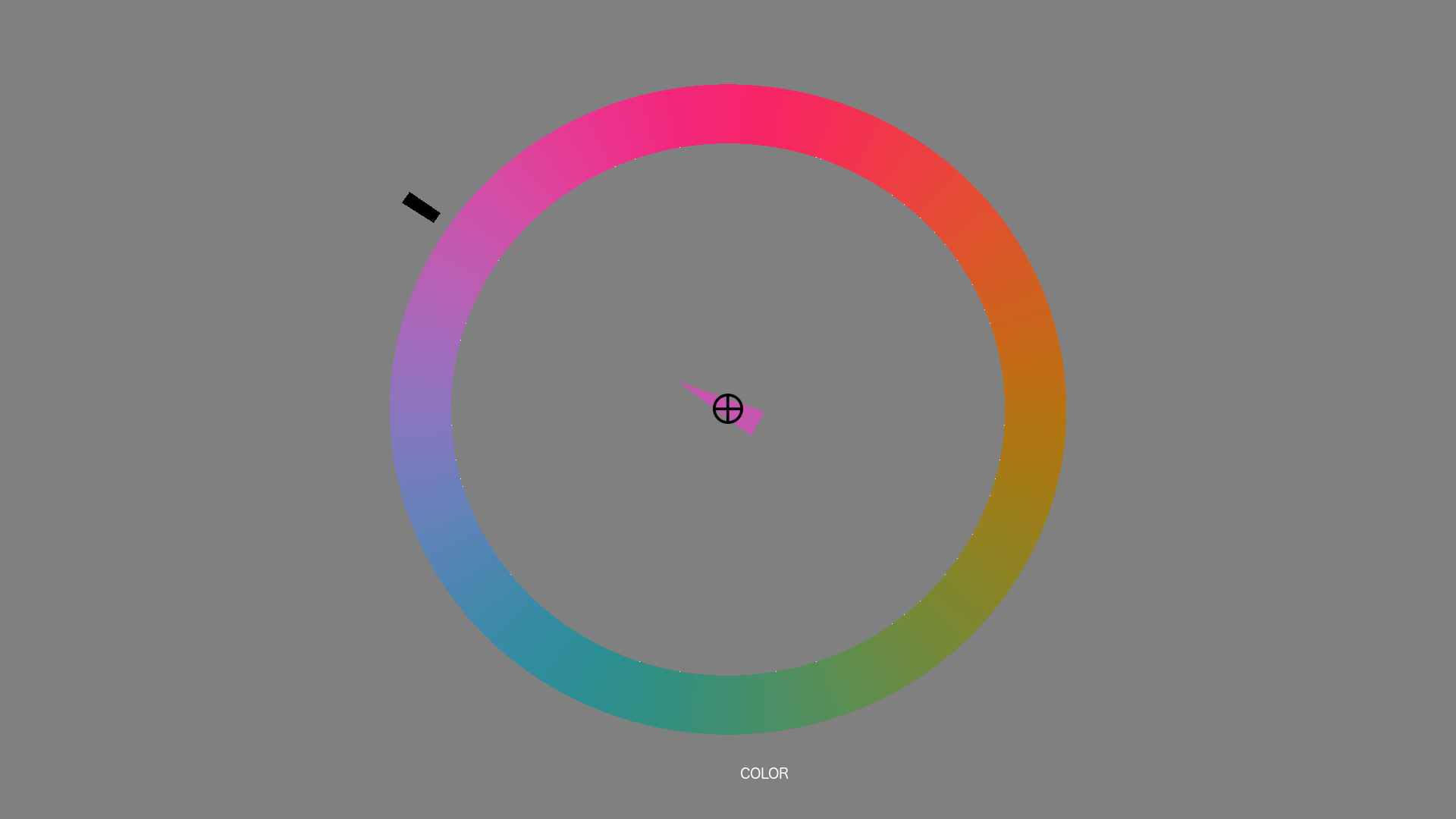
1. After some delay one of the triangles with fixed orientation appears in some random location. Modify the code to allow the subject to match the triangle location. We expect something like this:



1. The subject must receive a feedback and a score earned, like in the basic task.
2. Make another version of task that cues the color. In supportFiles folder load the colorwheel360.mat and pick up 5 main colors for the triangles (must be picked up from 12 equidistant colors in the matrix plus the jitter – within 5 rows up or down). You must get the following stimulus screen:



1. after some delay one of the triangles appears at the center. Modify the code to allow the subject to match the triangle color:



1. Make sure the probe screen duration is limited, usually, the subject has to respond within 20 seconds. If no response has been received – the program marks the corresponding values in datafile as “NaN” and moves to another trial.
2. Allow the subjects to submit their response a) by clicking the mouse and rotating with mouse. b) with keys 1,2,3 and 4 and rotate with ‘t’ and ‘b’ (mimics buttonbox and dial).
3. It is important to ensure the stimulus duration as exact as possible. For this reason it is advisable to count the exact number of frames from the stimulus presentation interval and update the screen exactly after the last frame appears. For more info please refer to Peter Scarfe’s PTB “Accurate Timing Demo” and “Wait Frames Demo” tutorials: <http://peterscarfe.com/ptbtutorials.html>

Software Engineering Questions

1. The original code is designed quite poorly. It relies heavily on global variables, that are getting overwritten during the runtime. Please modify the code following the principle of information hiding whenever it is possible.
2. Modify the modules/functions (you are free to add new modules/classes/functions with reduced component size) whenever it would be possible.
3. The original code is quite complex and hard to read. What you can do to reduce the complexity and make the code readable even for novices.
4. Alter your code to ensure the weak coupling and strong cohesion.
5. Add necessary comments to make it understandable for non-programmers.