# Practical 3.2 - Visual Scan Tool Part 2

Dr. Butler's graduate student has requested additional functionality for the Visual Scan tool. She wants a second presentation mode to be added:

1. The stimulus is a small coloured circle, on a black background.
2. The circle wanders randomly around on the screen.
3. At random intervals, the circle changes colour (a two-colour toggle is sufficient).
4. The subject's task is to respond when the colour changes.

Add this functionality to your Visual Scan tool.

1. Provide controls to allow the user to start either of the two presentation modes.
2. Provide feedback so the user knows that his or her reponse has been registered by the system.
3. In "wander" mode, your output file data should indicate the location of the stimulus each time it changed colour, and whether the user responded to that change.

Watch carefully for moments where your original architecture is making it difficult to add the new functionality. What is causing the problem? What would have worked better?

===============================================================================

**NB**: Consider carefully how to implement this behaviour without making the screen flash unpleasantly. I suggest you use the double-buffering technique you learned in Programming 4. We can discuss this technique if you wish, but you will recall the basic approach:

1. Create a Bitmap the same size as your main canvas (Graphic instance)
2. Create a Graphic (canvas) from the Bitmap.
3. Do all your erasing and drawing for each screenful to this in-memory canvas
4. When it's all set up, draw the Bitmap to the main canvas on the Form.

The following statements are part of my implementation of the Visual Scan tool (I use small images to make the circles on the screen).

Graphics mainCanvas;

Bitmap offScreenBitmap;

Graphics offScreenCanvas;

....

offScreenBitmap = new Bitmap(displayWidth, displayHeight);

offScreenCanvas = Graphics.FromImage(offScreenBitmap);

mainCanvas = pnlDisplay.CreateGraphics();

....

offScreenCanvas.DrawImage(stimulusImage,stimulusXLoc,stimulusYLoc,

stimulusDiameter,stimulusDiameter);

mainCanvas.DrawImage(offScreenBitmap, 0, 0);

===============================================================================

**Optional Extra Functionality:** The purpose of this tool, with its two stimulus modes, is to locate blind spots in the visual field. At the moment, your output isn't very helpful for this. The clinician will have to import your list of trials into Excel and sort and rearrange them to see where on the screen stimuli were missed. Consider providing a facility which reads in the csv file and translates it into a 2-dimensional output report showing the exact location of any missed stimuli.