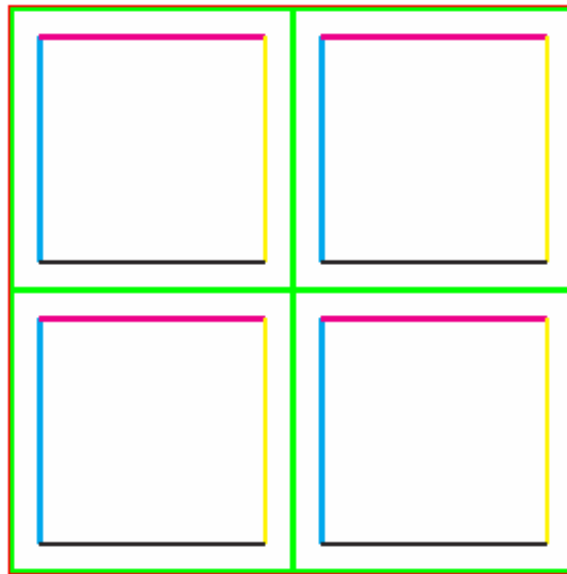
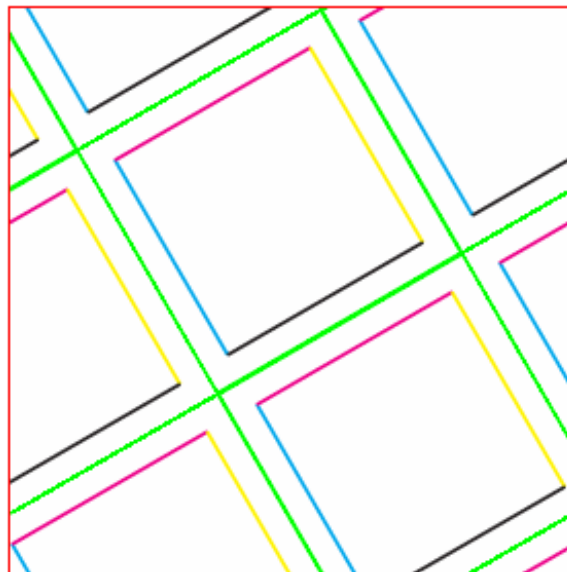


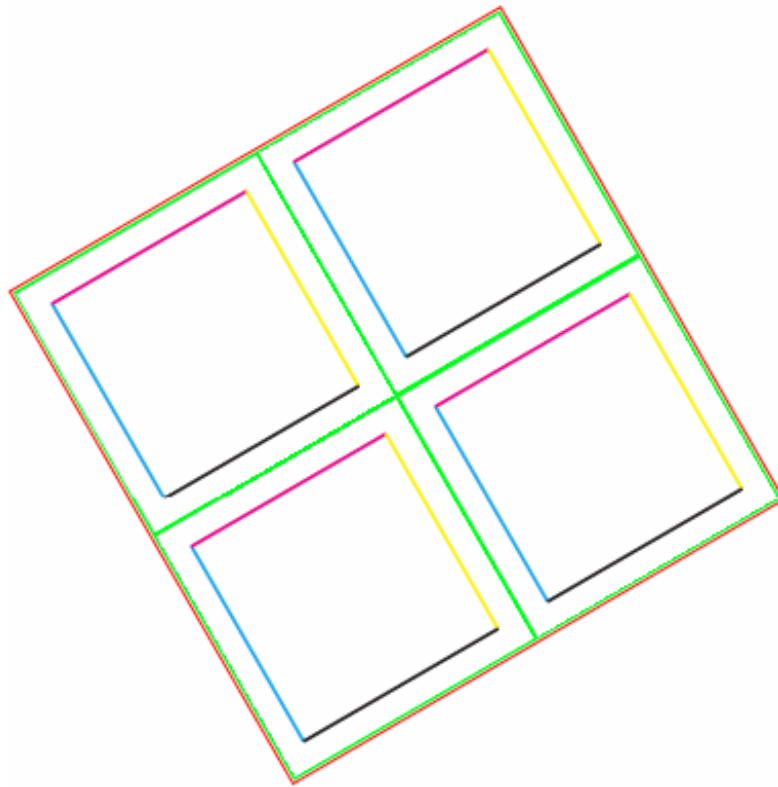
Consider some example patterns, the first is a straight forward rectangle filled with a rectangular pattern. The red strokes indicate the boundary of the filled rectangle, the green indicate the boundary of the halftone cell. The CMYK lines demonstrate how the cell is filled.



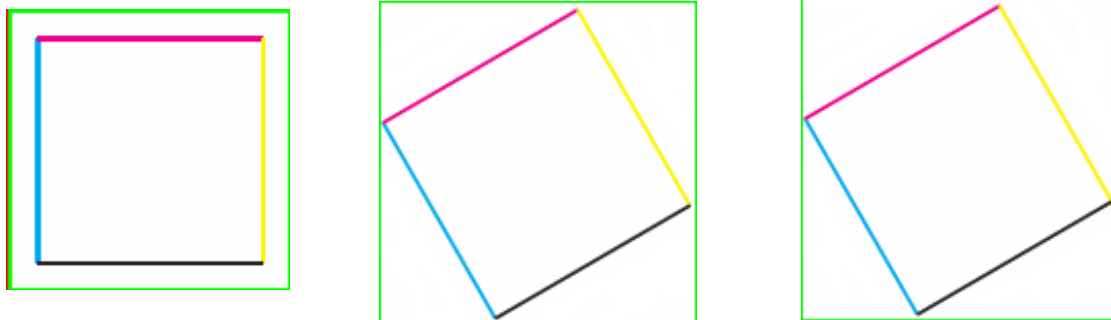
Next we rotate the pattern cell, but continue to tile orthogonally.



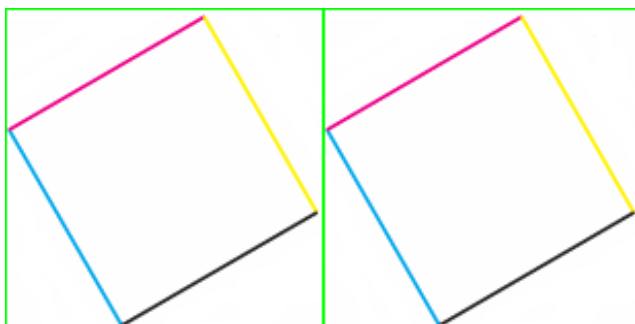
Finally we rotate the CTM before creating the pattern or the rectangle to fill and we rotate the pattern cell.



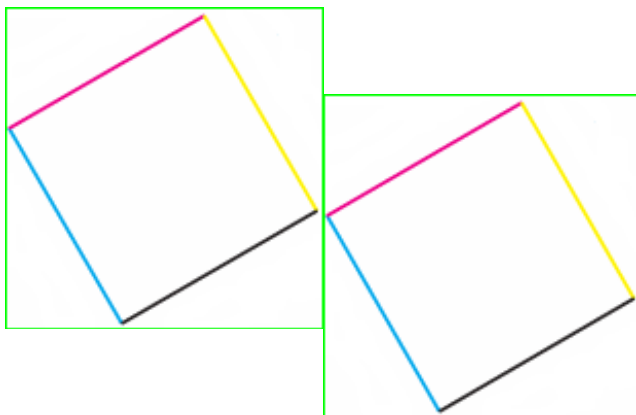
Now, Ghostscript captures the pattern cell in device space, lets see what actually gets captured in each case.



Notice that the captured pattern cell is the same for the two rotated cases. Now, when emitting the captured pattern, we can only tile orthogonally, which causes us a serious problem, if we tile these cells orthogonally we will get the incorrect answer for both the rotated cases:

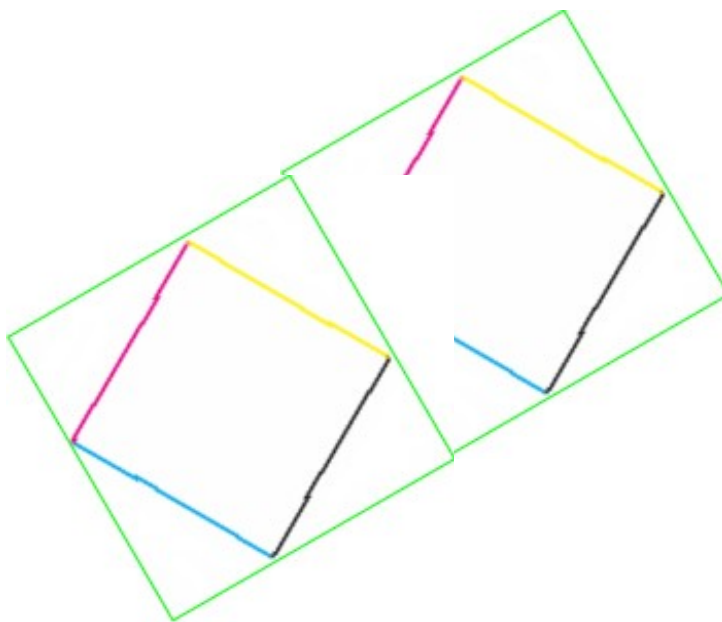


Ghostscript solves this by having a step matrix, which means it need not tile orthogonally:



Unfortunately this is not possible with PDF files. The only way to have the pattern tiling non-orthogonal is to alter the CTM **before** the pattern is drawn.

Now because the pattern content has been captured with the CTM applied this would give rise to the following result:



To solve this we must first 'unrotate' any rotation in the cell.

If rotation is required (**either** because the CTM has been rotated or the pattern matrix causes rotation) then we need to apply the rotation as part of the emitted pattern matrix.