Surrey Tetrominos Report

Motivation: Our group chose the recreation of the classic game of Tetris in matlab as our project. But why? Well, Tetris in many ways is perfectly suited to matrices. The game is played on a board, 20 squares high, 10 squares high. It consists of tetrominos, shapes that fill 4 of these squares at once, falling from top to bottom, with movement left and right and rotation being available. At any time, you could take a snapshot of a game of Tetris and describe it using a 20x10 matrix. By extension, a game of Tetris is merely the player changing a 20x10 matrix which represents the board with respect to certain rules.

Methodology.

First, in creating a game of Tetris a graphical user interface had to be created. While every stage in Tetris can be described by a 20x10 matric, playing Tetris on the command window of matlab is understandably, not a very friendly experience. While we tried looking at <https://uk.mathworks.com/matlabcentral/fileexchange/34513-matlabtetris>

for inspiration, eventually what was settled on by was instead using guide, creating a 20x10 array of panels, and using them as pixels, their colours changing as blocks fell. By doing this, no objects needed to move in the ui at any time, only change colour, and the number of objects on screen at once remains constant, leading to constant performance regardless of the shape of the game. After that, an array containing the handles of all the panels representing the game board was created, a 20x10, which for ease of use corresponded to their positions on the game board. From that, the matrix on which the game was played could be transformed into a simple graphical display similar to retro versions of Tetris. Having created this, three main matrices were created. The GPM (global positioning matrix, a 26 by 16 matrix with the outer most 3 layers representing impassable invisible edges to the board) on which the game is played and its composite parts, the GPMstatic, the matrix without the currently active block on it, and the ABP (active block position) a 2x4 matrix of the co-ordinates of the currently active block. Programing the rules from there wasn’t too hard, but was very time consuming. The other major hurdle was controls, being able to use the keypad to issue commands to the game. Qian Sean and Michael eventually settled on what you see in the code, a function that activates upon a button press and changes depending on which case (which button) is pressed. After working out the kinks such as being able to update the blocks positions outside of the main game function, and ensuring key controls aren’t being processed at the same time as a block is falling down a level we completed the game. Michael then added the high score feature we had as an expansion wish.

Results obtained:

The net result of our hard work is a functional retro inspired version of Tetris, programmed entirely on matlab, with a score counter, persistent high score, and increasing difficulty in decreasing the time between blocks falling down a level as score increases. This was exactly what we set out to achieve from the start, and the high score persistence was a stretch goal over and above what we had aimed for when starting this project.

Contributions Table

Peter Taylor (NotAdeer on github). 40%.

Sean Khosla (SK00709 on github) 20%.

Qian Zhou (Krystalzq) 20%

Micheal Ung (mikeung1 on github) 20%

Hadley Metcalf (mrhaders on github) 0%