Interim Report – Giguesaur Project

Ashley Manson

Co-Workers: Joshua La Pine, Shahne Rodgers

Supervisors: Geoff Wyvill, David Eyers

The Project

We are creating an augmented reality jigsaw puzzle game. Players will try to solve a virtual jigsaw puzzle using iPads connected by a central server. There are three parts to the project, the server side which is being developed by Shahne, the localization and vision part that is being developed by Joshua, and my part which is the game logic and rendering of the game to the iPad's screen. The game logic determines how the puzzle pieces interact with each other and how the jigsaw puzzle is solved. The rendering of the game is having the image of the board displayed on the iPad's screen with the puzzle pieces on it, making it look like the pieces are laying on the physical board in the real world, simulating the augmented reality nature of the overall project. I am also developing the user interface for the game which is picking up and placing of the puzzle pieces.

Background

There are many jigsaw puzzle games that are available for iOS and other hand-held devices, such as Magic Jigsaw Puzzles [1] and Jigsaw Puzzle [2], but the majority of them are limited in the way they look due to them using an orthographic projection to render the jigsaw puzzles. What this projection means is that the puzzle pieces of the jigsaw puzzle are flat on the screen, the player can only look at the jigsaw puzzle from top down, there is no depth and all the puzzle pieces are displayed as the same size. This is something that won't work for the project as the puzzle pieces will have to be rendered in a perspective projection, so that when I start rendering the jigsaw puzzle onto a board it will look more realistic, as puzzle pieces that are further away from the camera will be shown to be smaller than pieces that are closer to the camera. Another limitation of jigsaw puzzle games is the way they can be interacted with, by

which I mean the way they can be solved. The puzzle pieces have to be placed into a predefined grid. Farms And Animals Puzzles [3] is an example of a game that has this grid layout for the puzzle pieces to be placed in. This means that all the puzzle pieces will be placed in the centre of the screen. This is something I don't want to do as I want to have the jigsaw puzzle able to be solved anywhere on the board, be it in the centre or off in a corner of the board. I also believe that it will make for a more interesting game.

What I have Achieved to Date

I started to develop the logic for the jigsaw puzzle in the MacOS environment, with the plan to port the code over to iOS in future once I had a grasp on OpenGL (which I am using to render the game) and an idea of what the logic of the jigsaw puzzle would look like. I have finished the majority of the game logic. I have puzzle pieces, which are simple square polygons, being displayed on the screen with textures. Pieces are able to be picked up and placed back on the board, pieces can have a rotation, and pieces can be joined together for the puzzle to be solved. Currently I have ported the majority of the code over to iOS and have simple prototype working for the iPad. Puzzle pieces can be picked up and placed back down, and pieces can be joined together for the puzzle to be solved. Rotation of the pieces has not been implemented on the iPad version of the game.

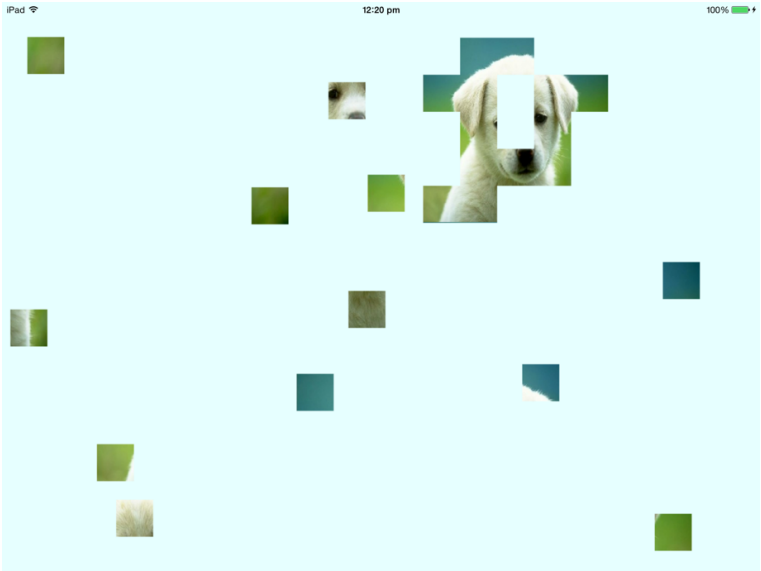

Figure 1: A Giguesaur puzzle being solved on the iPad

So what is a Jigsaw Puzzle?

A jigsaw puzzle in its simplest form is a grid of rectangular polygons with specific number of rows and columns, and each piece has the same side length. Each piece has four edges, and an edge either has a neighbouring piece or not. An edge of a piece can be open, meaning it has not joined to its neighbour currently or closed meaning it has joined to its neighbour currently, or if the edge has no neighbour it is invalid, meaning it can't join or be joined to another piece for that edge. An invalid edge would be the outside edge of the puzzle. Each piece of the puzzle has a unique ID, a position in space, or an x, y coordinate on the board, and a rotation with a value between 0 and 360 degrees. This is how I started to develop the game logic. I made up my own data structure to store these details, the ID determines the index of the array of pieces where the piece is held, the x, y coordinate determines where on the board the piece is displayed and the rotation affects the orientation of the piece. The board that the pieces are placed on has a width and length, which are along the x, y axis, which confines where the puzzle pieces can be placed, as all the pieces have an x, y coordinate.

Rendering Pieces and Moving them Around

The first major goal was to have a jigsaw puzzle rendered to the screen and the ability to move the jigsaw puzzle pieces around. I used OpenGL to do all the rendering of the jigsaw puzzle, using the coordinates of the jigsaw puzzle pieces, their current rotation and side length of the puzzle pieces as parameters to get them displayed correctly. With the parameters of each of the puzzle pieces, I get the four coordinates of their corners, with these I pass them to OpenGL and the puzzle pieces are rendered to the screen.

To pick up a piece the user taps the image of the piece on the screen. That piece that the user has tapped is then stored in the users “inventory” and they are no longer able to pick up another piece, until they have placed the piece that they are holding back onto the board. The inventory is just a term I am using to state if the user is holding or not holding a piece, meaning it can be easily checked to see if it is empty or not. This is also allows me to increase the storage space of the inventory later on in the project if I wish to allow the user to hold more than one piece. Currently the inventory is displayed in the lower left corner of the screen, meaning the

user can see if they are holding a piece or not. The screen x, y coordinate from a finger tap are converted to a board x, y coordinate. If this is close enough to a piece then that piece is added to the user's inventory. This allows for the board to be rendered with a perspective projection, making the scene more realistic and achieving the augmented reality feel.

Pieces Snapping Together

Now that I have pieces moving around the board, I wanted to enable a feature that allowed pieces to snap together, meaning if a piece was in a specified range to its neighbour, the piece being placed back on the board would move so that the two piece were right next to each other, or they snapped together, which is shown in the Java Jigsaw Puzzle game made by centurio23. [4] This also has the added bonus of making it easier to check if pieces are right next to each other, rather than relying on the user to carefully place pieces together. How the check is made to see if two pieces should snap together is by comparing the distances between the corresponding corner points, and if the distances are less than the snap variable, the pieces snap together. So if the piece being put back on the board is being snapped to its left neighbour piece, the original piece top left point and the neighbours top right point would be checked, as well as the original piece bottom left point and the neighbour's bottom right point would be checked. The reason two distances are checked is to make sure that piece rotations are taken into consideration when snapping pieces together, so as avoid an unusual snap where a piece rotated 90 degrees snaps to its neighbour not rotated. When a piece is snapped to another piece, the piece saves the rotation of its neighbour as its own, so that when the change is rendered, they both have the same rotation when beside each other. For a puzzle to be considered solved, all the pieces should be snapped to their corresponding neighbours. Once a piece has snapped to its neighbour, a variable for that edge is set as closed, as well as the neighbour's edge. So if a piece has snapped to its right neighbour, its right edge is set as closed and the neighbour's left edge is set as closed. The check to see if the puzzle has been solved goes through all the pieces' edges to see if they are all closed.

What the Giguesaur Game Looks Like

The current build of the Giguesaur game on iOS is a simple pick up and drop jigsaw puzzle game meaning the user can tap a piece to pick it up and place the piece somewhere else on the board. The jigsaw puzzle is currently being rendered with an orthographic projection, but will be ready to be changed to a perspective projection once Joshua is ready to integrate his part of the project with mine. All the pieces are square polygons with textures on them, with no curved edges. The puzzle can be solved by placing all the pieces next to their corresponding neighbour pieces. There is very little semblance of a user interface, pieces can be tapped on, which puts the piece in the lower left corner of the screen, representing that this piece is in the user's inventory, the user can then tap the screen again to place the piece back on the board. There is currently a bug where if there are too many pieces on screen, more than twenty-five, the game will crash, I still haven't figured out what is causing the issue.

Things to do

There is still much to do and I need to add more to the project. Square pieces don't make an interesting game, so I want to implement a feature to generate curved edges of the pieces. This would mean there is an easy way to see how two pieces would slot in together, it would also make the jigsaw puzzle pieces actually look like "jigsaw" puzzle pieces.

The interface to the game needs to be changed for when pieces are being placed back onto the board. Right now, when a user wants to place a piece back onto the board they tap somewhere on the screen, and the piece appears at that location. What was originally envisioned was when the user picked up a piece, it would be placed in the centre of the screen, but partially transparent, so the user could still see the board and other pieces behind it. This piece in the centre of the screen would then cast a small shadow onto the board to show where it will land if the user drops it. For the user to change where the piece would appear they have to move and rotate the iPad around the physical game board manipulating the position of the shadow. This allows the user to align the piece they are holding to another before putting it down.

As the project is being developed in three different parts, the three parts still need to be integrated with each other. Shahne's part of the project will allow more than one user to

interact with the puzzle. A player will be able to see when another player picks up a piece or places a piece back on the board, as well as having more than one person collaboratively solving the puzzle at any point. With Joshua's part, he will be getting the position of the iPad's camera through various different matrices, he would then pass these matrices to me, so I can create the perspective projection for the game, and render the pieces correctly on the physical game board. Figure 2 shows what the jigsaw puzzle would look like when the game is rendered with a perspective projection.

Figure 2: Perspective projection from the MacOS build

Conclusion

I have achieved most of the major goals from my original aims and objectives document. I have a basic user interface, where a user can pick up and place a piece, have a way to join puzzle pieces together, and a virtual board to play the game on. I have also finished the game logic for the puzzle pieces, which was not set out in the original aims and objectives. Due to my progress from the first semester I am well on track to getting the project finished.

Update Aims and Objectives

I have now set out an updated aims and objectives for the second semester of the work I need to achieve to get the project finished. I have to figure out a way to generate curved edges for the puzzle pieces. Once my part of the project has been integrated with Joshua's part the interface will have to be updated, as well as adding a perspective projection to the game to

allow the puzzle pieces to be rendered on a physical game board. I will have to integrate my part of the project with Shahne's part, which would allow more than one user to play around with the puzzle. Once all three parts of the project have been completed, we will have to do a lot of testing and possible bug fixing of any issues that pop up.

References

[1] XIMAD, Inc. Magic Jigsaw Puzzles. https://itunes.apple.com/nz/app/magic-jigsaw-puzzles/id439873467?mt=8, 2015.

[2] Critical Hit Software, LLC. Jigsaw Puzzle. https://itunes.apple.com/nz/app/jigsaw-puzzle/id495583717?mt=8, 2015.

[3] DA Apps. Farms And Puzzles – Jigsaw puzzle for children. https://itunes.apple.com/us/app/farm-animals-puzzles-jigsaw/id543511649?mt=8, 2012.

[4] centurio23. Jigsaw Puzzle. http://sourceforge.net/projects/jigsawpuzzle/files/, 2010.