Survival of the

CARROT PEOPLE

Interim report

Introduction

As presented on the 1st of April, the functional minimum has already been implemented. After this, we worked on implementing the low target and the desired target.

Implemented Features

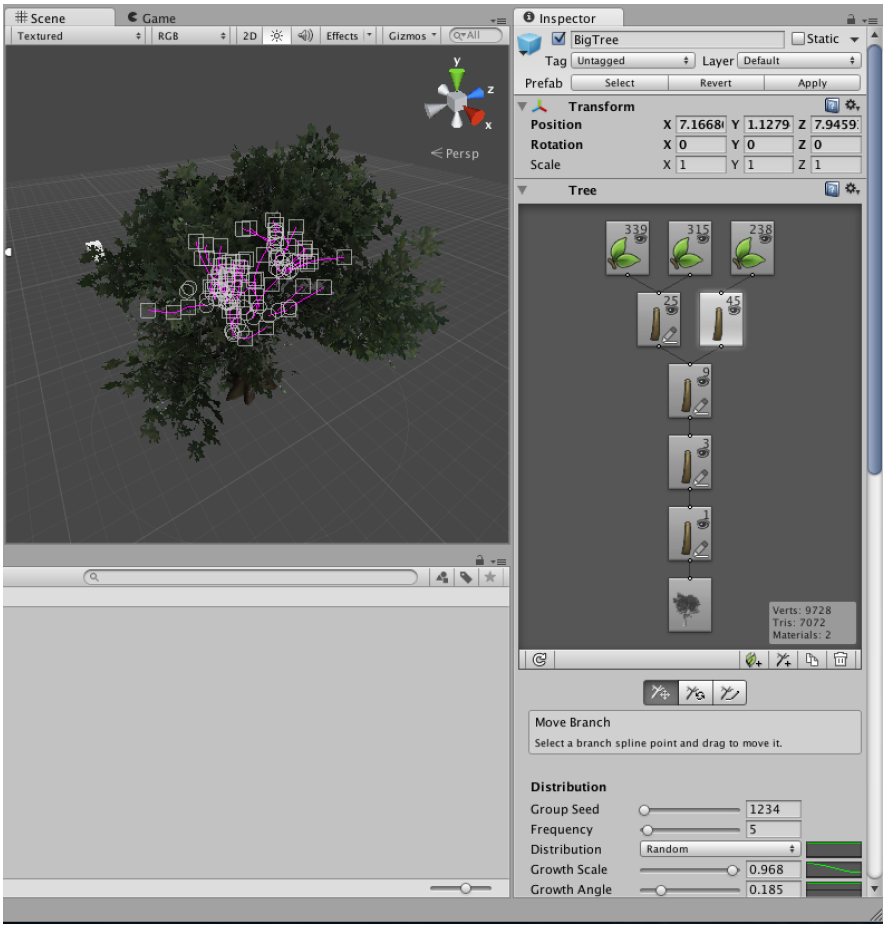
First, the functional minimum was implemented. This was a good way to start with Unity. After implementing the functional minimum, instead of continuing with the low target, it was decided that the core game idea should be tackled next: growing trees using an L-System.
Here is an overview of the progress:

Fully implemented features	**Functional minimum:** Basic tower defense functionality: ● Tower placement ● Tower attacks ● Monster spawning ● HUD ● One playable level ● One enemy monster
In progress	● Growing tower (graphical, L-System) ● Smooth path
Planned	New: ● Software engineering, refactoring Low target: ● Growing tower (technical, no L-System yet) ● A tower can split a branch to grow a new tower (technical, no L-System yet, every split tower looks the same) ● Winning condition: Enemy's Castle is overgrown by the towers / plants of the player (technical, no L-System yet) Desired target: ● Different towers/branches ● Different monsters ● Attack mode for towers ● Winning condition: Enemy's Castle is overgrown by the towers / plants of the player (graphical, L-System

	• Animated monsters and towers • Special tower skills (poison, slow-down, etc.)

Growing Trees / L-System

Unfortunately, this task was harder than anticipated, and therefore it is still work in progress. Currently, the idea is that the unity tree framework can be used. However, it is optimized to be used from the GUI, and therefore is difficult to use from the code (it has to be dynamic, that is the major problem).



Tree editing in the Unity GUI

New Requirements

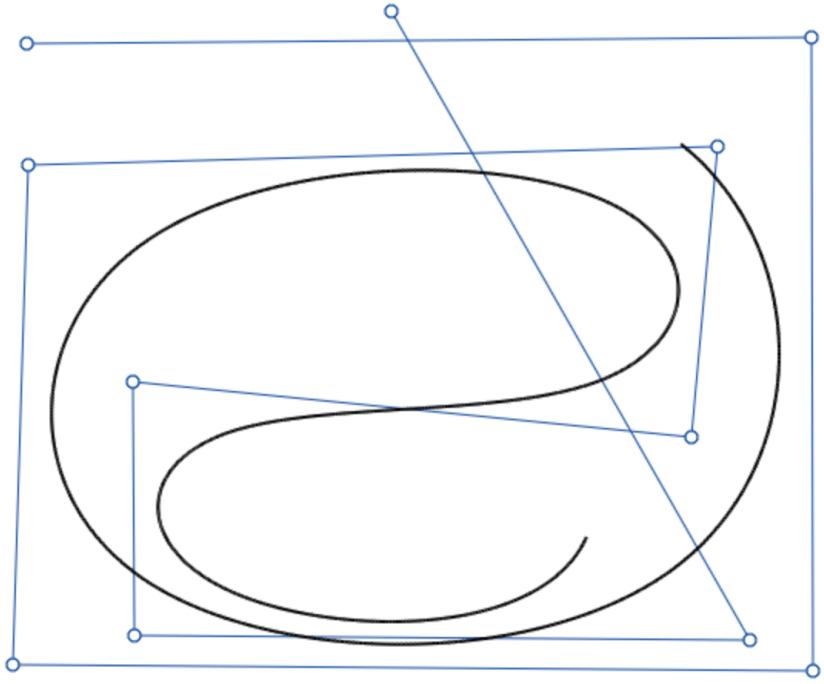
During the last weeks, the following new Requirements were discovered.

Software engineering / refactoring

In the complex Unity framework nearly everything is possible. There are many online tutorials to implement certain things, but many of those are hacks which don't work in large projects. After implementing the functional minimum, the project was, from a software engineering prespective, a mess.
There is an article about best practices on http://devmag.org.za/2012/07/12/50-tips-for-working-with-unity-best-practices/. It is planned to implement some of those concepts and to rewrite some stuff, so that these software engineering issues can be resolved.

Smooth Path

In the current implementation, the path can be defined by multiple checkpoints. They then are connected automatically and the path is drawn on the map. The monsters follow the path until they reach the last checkpoint.
Currently, the checkpoints are connected by straight lines. This doesn't look nice and smooth, and also the enemy units only walk straight.
The new functionality is that that the line is interpolated between the chechpoints in a way that the path looks smooth (e.g. using B-Splines / Bézier curves).



Example of B-Splines