My little Tornado

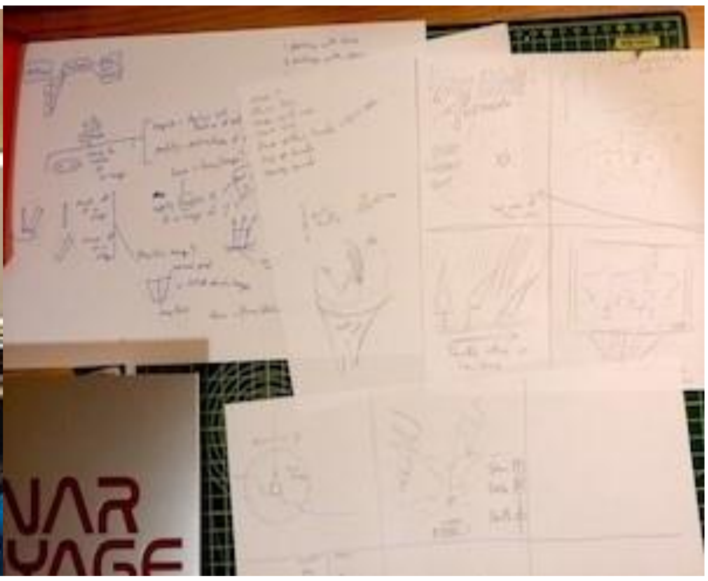
This is a follow up design document containing the research, tech description and post-mortem for the My little Tornado game I did In class. It was a week-long project and mostly was just to get familiar with Unity again as I had not worked with it in about 3 years.

A few key technologies I wanted to use: DOTS(ECS) System, Terrain design and LWRP with custom shaders.

Tornado Development:

My original idea was to create some sort of static tornado that you could throw objects into. That static tornado would be views probably from a single angle or point. However I made a paper model tornado and started playing with it in my apartment.

When you start playing with a paper tornado you notice a few things. Firstly it's not very fun if it's not moving. A moving tornado goes and sucks things up and that's way cooler than one just sitting there. Secondly, you start to want more tornados. One tornado on its own is fun, but get multiple tornados together and they can be like rivals.

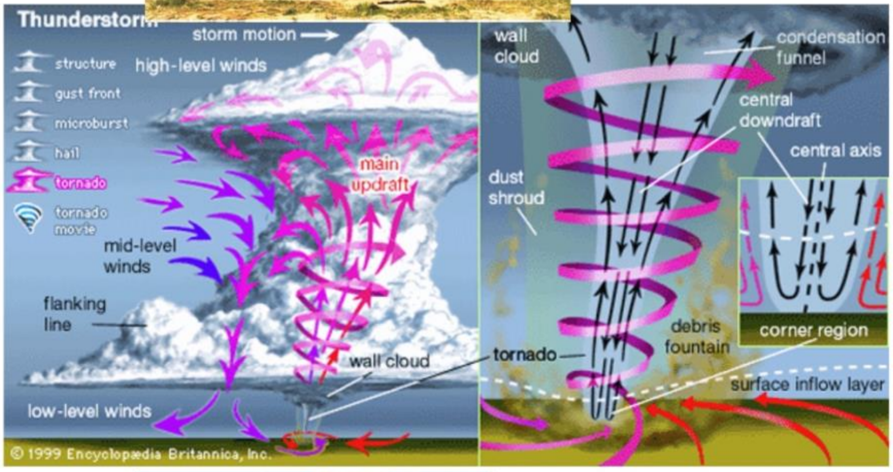

Above is the paper model and some early sketches.

This changed the idea quite a bit, and introduced some technical questions. If simulating tornado physics from one place was not enough. Now that simulation had to be a moving point. On top of that there could be multiple moving points.

It's also worth noting that in my test area (aka my apartment). I was not testing on flat surfaces. The multiple tornados should be on some sort of rough terrain.

So once you combine those needs. Moving tornados, multiple tornados, rough terrain, objects for tornados to suck up. You pretty much have everything that defined the final game sorted. Although at this point it was totally unclear if this was all possible in the timeframe.

Take a look at the next page to see a reference and design images that went into the tornados and tornado related things.



MY LITTLE TORNADO

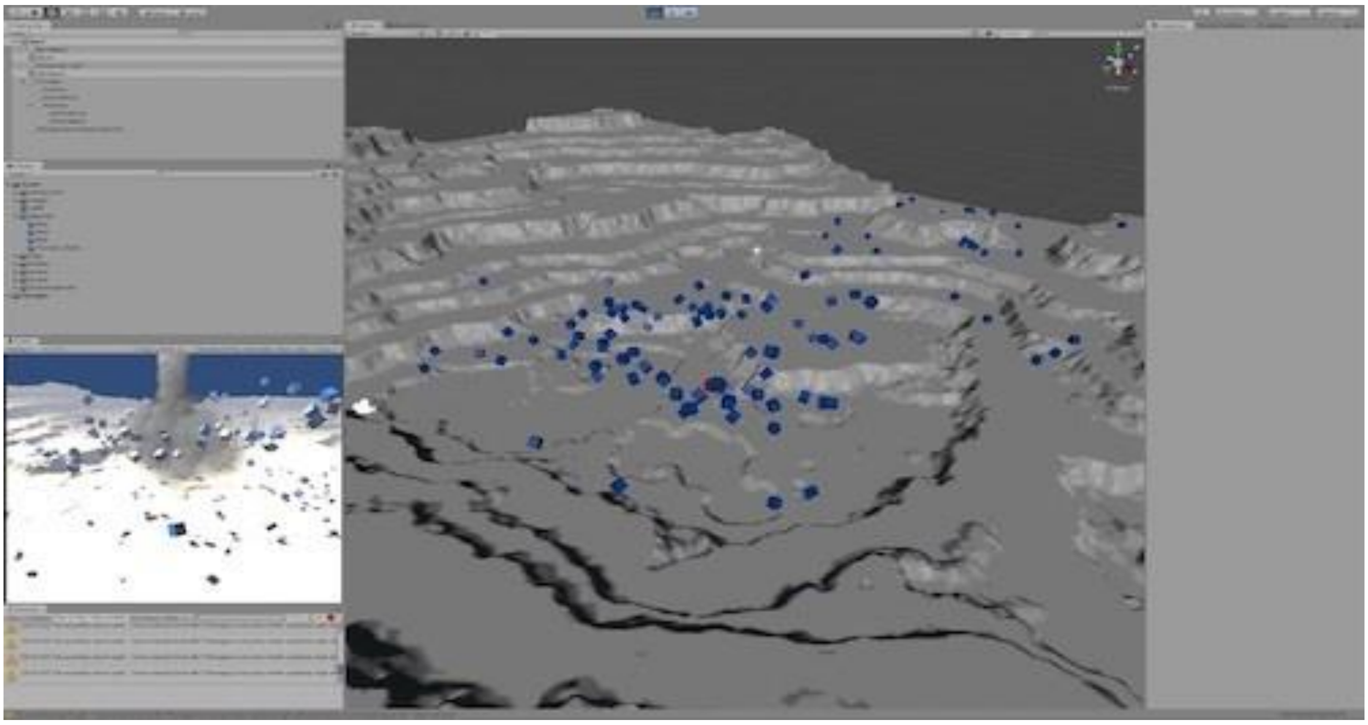
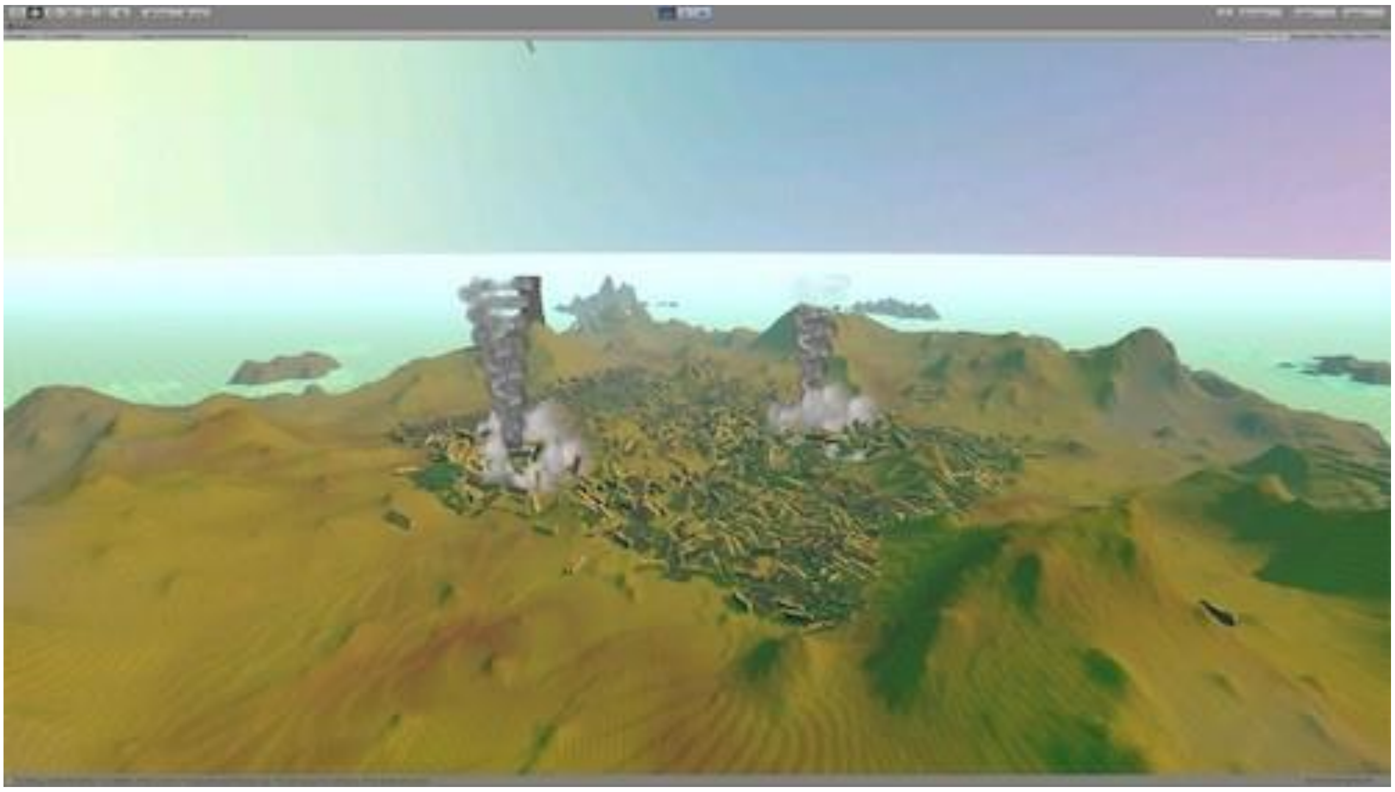


Image above is an early stage tornado with entity physics in place.

There was an issue with entities and terrain, basically entities (the blue boxes above) don't work with Unitys internal terrain system. In this case it was solved by stacking a terrain object under the Unity terrain.

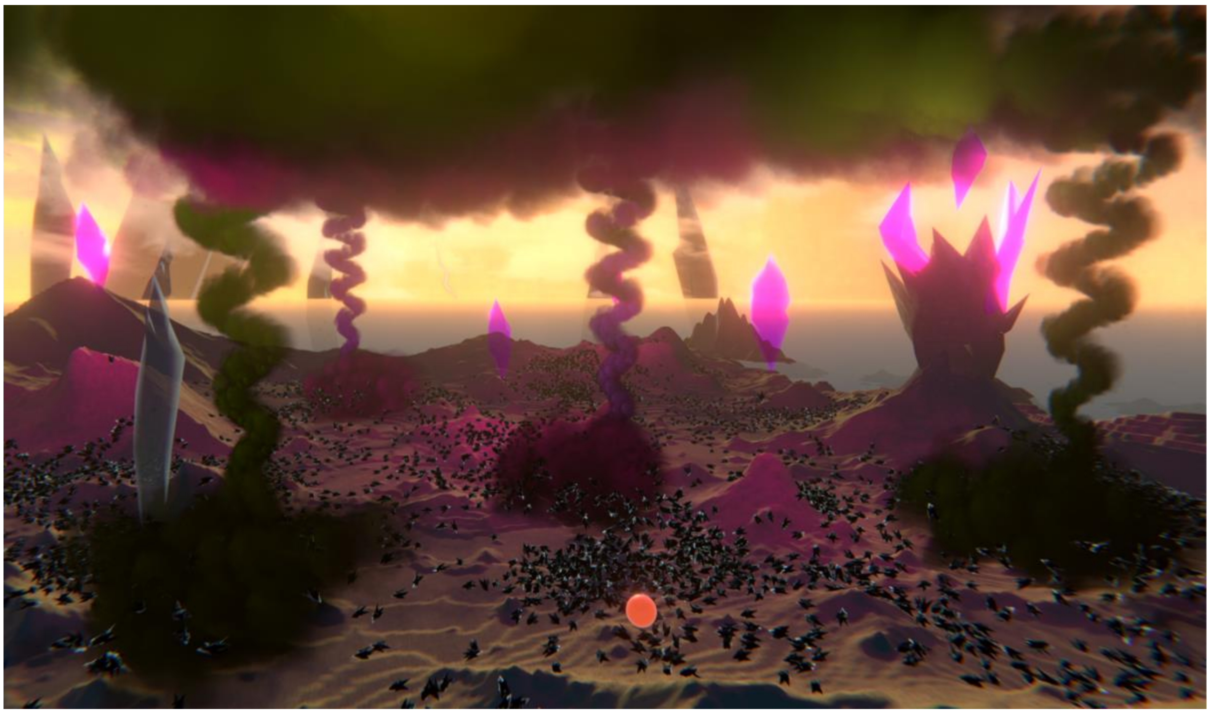


In the image above you see entities interacting with a terrain object but vanishing as they fall through the surrounding landscape.

There seems to be no LWRP friendly trees out there, so I was eventually forced to scrap the use of terrain trees. You'll see later on this was the reason I put crystals into the landscape instead.

The crystals also serve as continent spawning locations for the gems that the tornados blow around, so they served a handy double function. I think that makes sense to the player too as a source of these evil things.

One neat trick was to strap the sun to the camera, meaning tornado particles were always in "shadow". This give the moody dark tornados you see in the final version.


Above you see the final game.

Technical Stuff:

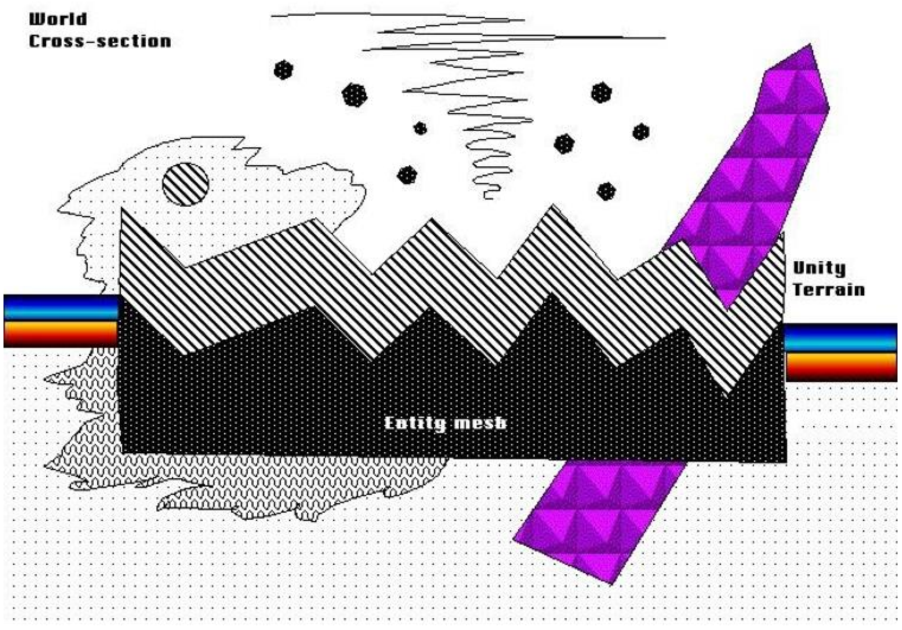
The meat of it is the DOTS system and entities. All power gems are entities, they individually poll for tornado locations and are then moved in relation to their closest tornado. Terrain physics also require an entity physics mesh of the terrain.

Tornados themselves are traditional game objects. Each tornado contains a number of particle systems, and a tornado script. This script manages all aspects of tornado movement, spawning, destruction and also informs the entity system of its location.

The players mouse position is converted to a 3D object called the manipulator via a raycast. The manipulator script handles most direct player interactions. Its location is used for spawning and moving tornados.

Script interaction between game objects and the entity layer happens via a static class called Earth. This class is also used to inform game scripts if the menu is on display, and to inform the jukebox of the tornado count.

Finally the world is made up of a unity terrain, a skybox, and some decorative game objects, some with minor scripts such as cloud rotation, or the spawning crystals which generate the power gem entities from prefabs.



What’s broken:

Particle and alpha priority and layering is a mess. It was a constant battle to get particles to show up in the correct order (At one point clouds overlayed on tornados). Tornados still have particle overlap issues.

There is lag at the start during rapid spawning. This does settle down after a while though.

Entities fall through the world outside of the terrain tile, this is an issue until unity make terrain entity friendly. Also there’s a weird lighting glitch with real-time lights on the terrain, I don't like the unity terrain system much, its clunky.

Stylistically, I don't love the world. I could use more skill at world painting and making it feel less messy. Also everyone calls the power crystals bugs or flies, so they could be presented differently.

I didn't get a whole lot of time with the shader system, I felt more like it was controlling me rather than me control it, needs more practice.

Some of the animation changeovers are time based, its not super stable and could cause issues if animations were ever changed.

Appendix:

Textures were taken from the Feudalist Texture Pack https://aetheris.wixsite.com/feudalist
Tornado sounds were provided by FreeSound.org https://freesound.org/
Game music was provided by the Mod Music Archive https://modarchive.org/
Duel Audio Script was provided by Igor Aherne www.facebook.com/igor.aherne