September 2018

Developer's Notes on
Cenodrome: The Lost Sub-Basement of TWHL Tower

Note: minor map spoilers are contained below, be sure you've played the map first if you care.

Background

TWHL Tower was a collaborative Half-Life mapping project born on the TWHL forums back in 2015, and officially released in early 2016. The concept was straightforward: each mapper would create a "floor" of the tower, and the maps would be connected via stairways.

The mod chose to use Spirit of Half-Life as its base (version 1.8a1). Spirit is a "mapper's mod" in that it doesn't add new weapons or enemies or maps, but it provides a ton of tools for mappers to use that enhance Half-Life. Much of Spirit's magic is probably transparent to typical Half-Life players. I mean, I don't expect players to notice that a particular sequence only triggers when they are looking directly at it – but that's something that's only possible in Spirit compared to standard Half-Life.

The allowed size of each floor was relatively small – only 1024 x 1024 x 160 units high. If you're not familiar with units in Half-Life and the Hammer editor, you can think of each unit as **roughly** one inch. This isn't a perfect conversion, since Half-Life was not designed with real-life scale in mind. It's a useful general comparison of sizes in-game, but keep in mind Gordon is more or less the size of a refrigerator in the game, so you can understand that this is just an approximation.

So the usable volume (not including the connecting stairways or the hallway leading to the main square-shaped "floor") was on the order of 85 ft x 85 ft x maybe 12 ft high, or 26 m square and 4 meters high. In any case, not a lot of room to build a map. But that's what was great about this project – limitations breed creativity.

But this map wasn't made for TWHL Tower.

As far as I can recall, I simply didn't consider participating in the TWHL Tower project. At the time I was working on my own mod, still yet to be released. And I don't think I was mapping very much in 2015 in general.

TWHL Tower was damn fun. It was pretty well-received in the community too. In fact, I even streamed it on the RunThinkShootLive Twitch channel – but the stream's VOD seems to be lost to the ages unfortunately.

I had a blast playing it. That said, I was a bit disappointed that more mappers didn't take full advantage of Spirit of Half-Life. I noticed a handful of things in the tower that used Spirit, but I really wanted to see someone harness Spirit and run with it.

So when the follow-up to TWHL Tower was born on the forums, I jumped at the chance to participate.

This project was called TWHL World, and to cut a long story short – it never was completed or released. But the map I made for it was Cenodrome. Even then I was thinking of it as a lost floor of TWHL Tower, since I really liked the severe space restriction of the map. And that was what probably doomed TWHL World – there were no restrictions whatsoever on the maps for it.

September 2018

Developer’s Notes on
Cenodrome: The Lost Sub-Basement of TWHL Tower

The Map Source

The original rmf and map files are provided in the mapsrc folder. You are welcome to examine and use the map as you wish. It is released under the following Creative Commons License:

Attribution-NonCommercial-ShareAlike 4.0 International (CC BY-NC-SA 4.0)
https://creativecommons.org/licenses/by-nc-sa/4.0/

This means you’re allowed to use it for any non-commercial purposes, as long as credit is given to me and you apply the same CC license to your released work.

As a separate pdf file, you can find 3 pages of flowcharts that outline the logic used in Cenodrome. The first page is for the relatively simple Round 1, where you don’t get any choice in the combatants. The second page is the typical logic for Rounds 2 through 6. Yes, it’s pretty complex but there is a reason for everything and I’m pretty sure simplifying it isn’t feasible without changing how the map plays. For example, I really wanted the player to be able to change their mind about which combatants were chosen, right up until the Start button is pressed. That in itself demanded a set of entities that would have been much simpler had the player not been able to change their selections.

The final chart is an example set of entities for Round 8, where you’re the one fighting in the arena against the waves of enemies. These charts probably don’t quite reflect the final entities exactly but everything should be pretty accurate.

Cenodrome is complicated, there’s no clearer way to put it. And it pushes the entity limit to the max in Half-Life, so much so that I just had to delete a few env_glows so that saving the map didn’t crash the game. So I hope it’s stable for you. I do get the occasional crash (and if you pay attention sometimes entities temporarily become invisible when there is a lot going on simultaneously) but none of my testers reported any repeatable crashes.

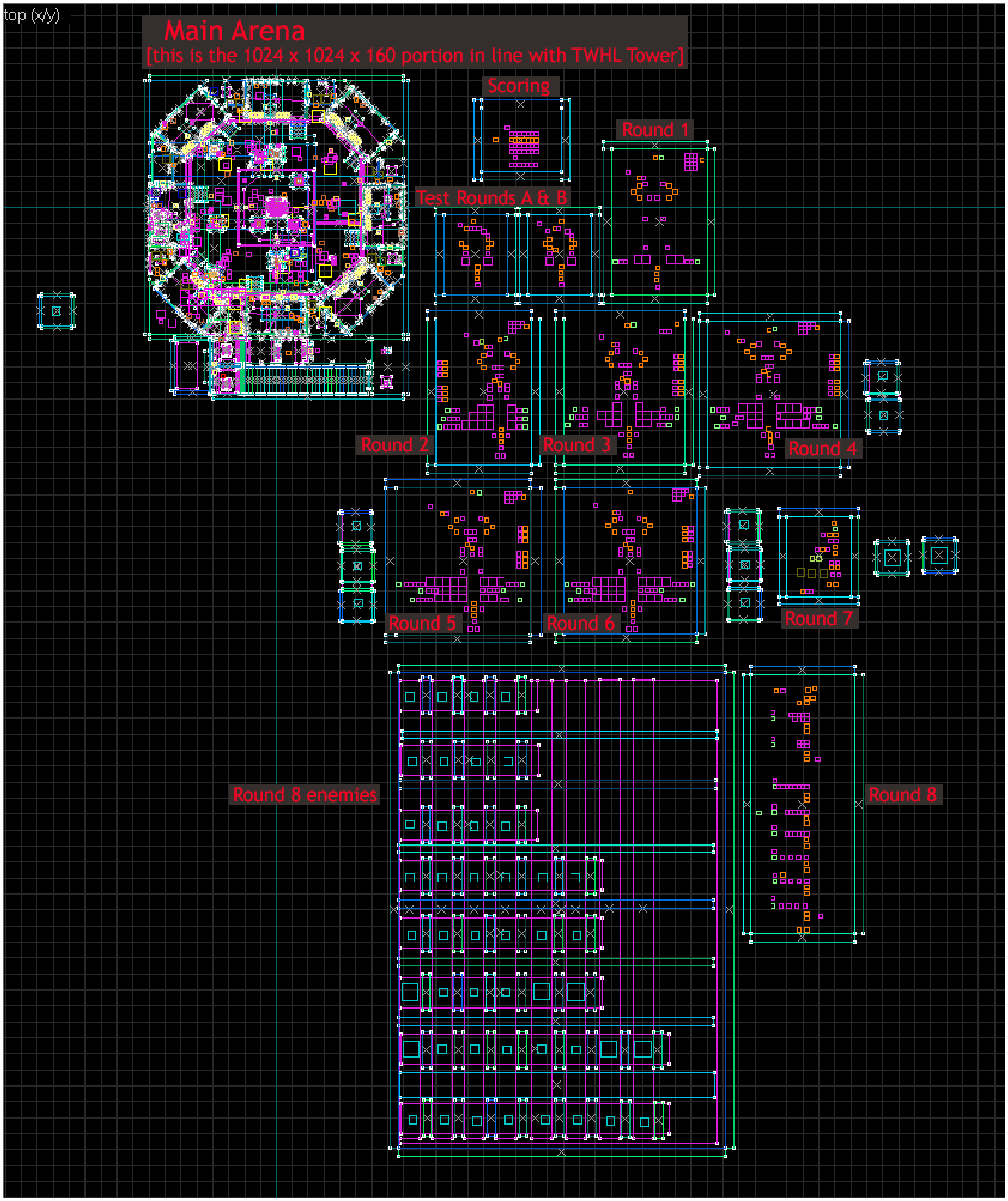
For the record, Cenodrome uses 1262 non-light point entities and 317 brush entities. The brushes aren’t a problem, as 512 is the limit. But the number of point entities means the map lives dangerously on the edge of crashing the game.

On the next page is a labeled overview of the map. The playable area is confined to the arena mentioned above along with the stairway and adjacent hallway. But I took the liberty to organize the entities by round into separated sections so I could easily find what I was looking for from the top view. This overview should help orient you if you dig into the entities and logic.

From an architecture point of view you’d think the map is simple. But given how thin the buttons and glass of all the windows are, it was looking ugly snapping everything to grid, so early on I decided to live dangerously. I started using the modified Hammer that supports floating point coordinates (HLFix was causing problems), and I went off-grid. Like, really off-grid. I still recommend mappers stick to on-grid mapping as much as possible. But at least for me, I’m certainly not reluctant about going off-grid now. Never look back.

September 2018

Developer’s Notes on Cenodrome: The Lost Sub-Basement of TWHL Tower



Labeled Overview of Cenodrome

September 2018

Developer’s Notes on Cenodrome: The Lost Sub-Basement of TWHL Tower

Design Notes

So, other than taking the map size restriction from TWHL Tower, my main goal with Cenodrome was to really take advantage of things that Spirit had to offer. After some churning on ideas, I decided on the arena where you selected the combatants. One Spirit example: other than the NPCs who generally hate each other in standard Half-Life (e.g. Barney will shoot at aliens), Spirit was required to make the NPCs enemies and fight each other.

This isn’t a comprehensive list, but here are other places I used Spirit in Cenodrome:

- The very premise of this map is not possible without Spirit. All allies are set to Faction A and all enemies are set to Faction B in the **Monsters behave as** property of monstermakers. Thus they fight each other, even if in standard Half-Life they would be allies (e.g. soldiers with MP5s vs. soldiers with shotguns).

- The NPC spawning effects mostly use **env_warpball**, which was borrowed from Blue Shift. It’s a coordinated alien spawning effect that in the original Half-Life was a handful of entities triggered at specific times. The warpball is a single entity that takes care of this automatically. Note that the Round 8 enemies are teleported in, instead of spawned with monstermakers. This is because I wanted them to be alert and hostile immediately – you’ll notice that the previous rounds always allow a few seconds before the round starts because spawned NPCs are pretty dumb and just sit there for a short time before doing anything.

- Many things (the button highlights and some doors) are distinctly triggered on or off using Spirit’s ability to **designate on and off using + and -** in front of the target names. For example, when you select a specific ally, the other 2 button highlights are told to turn off. Normal behavior in standard Half-Life would toggle the button states.

- To make the two ceiling turrets go active and inactive (and not wake up when new NPCs spawned) I make them into prisoners and non-prisoners using **env_customize**, which allows you to change properties of an entity during the game.

- Each control booth has simulated smartglass that goes opaque when it’s not in use. This is actually a gray func_illusionary behind the glass, and I use Spirit’s new **Fade Time** property on env_renders to fade the gray illusionary between opaque and transparent over a short time.

- The “Boss” is just a monster_bullchicken with its new **Scale** property set to 1.5. I actually wanted it bigger but the hitbox doesn’t scale with the visible model, and 1.5 seemed like a good balance between an obviously larger bullsquid and one that actually took damage when it looked like you hit it with the crowbar.

- The health bar for the boss is a combination of entities:
  - A red **momentary_door** for the actual health bar

  - A **calc_numfroment** [number from entity] set to get the ratio of health/maxhealth from the boss
  - A **trigger_relay** set to target the health bar momentary_door, and set its number (position) to the result of the calc_numfroment entity
  - A **multi_manager** that targets the trigger_relay at a delay of 1 second, and set to loop so that the health bar position is updated each second

- At the end of Test Round B, the surviving NPC (Barney or zombie) is killed by lasers. These target the NPC's head, using a **trigger_motion** that moves the info_target (which is the laser endpoint) to the NPC's eyes using a **calc_posfroment** [position from entity].

- The fire effects are done partially using Spirit's Aurora particle system through **env_particle**.

- I use a lot of **multi_watcher** entities in Cenodrome. For each round, I have multi_watchers to detect when both an ally and an enemy has been selected using the buttons. And then there are multi_watchers to monitor when the entire ally or enemy team has been killed to determine whether the round is won or lost. They are pretty powerful entities.

Final Notes

Thanks for your interest if you read this far. Spirit is quite powerful for Half-Life mapping and notoriously poorly documented, and often unstable. That said I had a blast experimenting with what it offers and pushing an otherwise tiny Half-Life maps to the limits.

If you have any questions related to this map or how I built it, do let me know.

Thanks,

Don

AKA Unq
AKA Unquenque