**Update:**

I could do X case studies of different FPS: CSGO, Half-Life, DOOM…

And then come up with the heuristics than define each.

Then, in the prototype, there is a pre-set for each of those games, that is to say when the app analyzes the user's level, it compares the level with the preset.

This concept helps relate theory and practice + case studies are a must

theory = 3 case studies + automation analysis in some real games + methods studies + final model ?

**How to score an FPS level so that it can be improved? 3 other questions need to be answered:**

**Which aspects to take into account?**

Aspects that can for fact define any FPS level:

* Balancing both sides of the level. Does not have to be 50/50 if the intent isn’t so, eg. a capture the flag mode where capturing is easier than defending. <http://antoniosliapis.com/papers/learning_the_patterns_of_balance_in_a_multi-player_shooter_game.pdf>

This could include any sort of objective beyond deathmatch. Capture the flag, objective domination…

Following the CSGO example, the amount of time it takes to get from one spawn to an objective is crucial and higher for the attacking team than the defending team.

* Time between combat engages/deaths/kills. Meaning, the “activity” density within the level, it can be dull → slow → average → fast → overwhelming. Could be relative to the size of the map like in:

<http://julian.togelius.com/Cardamone2011Evolving.pdf>

* Series of deathmaps/heatmaps/other maps. Where do most engages happen?
* Data for optimal weapons, used items, strongest enemies… What is the average range for kills? Which one was from furthest away? Most chosen weapon by section is useful for creating a level that takes many types into account.

A way to do this, beyond giving each team a specific weapon (balance section), would be to look at the "tile attractiveness mao", that is to say the value that a bot assigns to each tile depending on enemies, cover, weapon…

Said map depends on the weapon and thus we can compare that map between weapons and see if there are particular areas where a sniper scores much higher than a guy with a shotgun or if its more regular.

The map would be translated to a heatmap where the value is each tile attractiveness.

Even so, a tile can have different values in different frames because enemies around change. For each move, a bot processes maybe 10 tiles around him, so those tiles' scores would be stored and if the same tile is processed again and again, in the end we calculate the median between them.

* Player priorities/personas/bartle types. It can be combined with the player’s knowledge of the level (times played). Which player type accomplishes its objectives the best in the level?

<https://arxiv.org/pdf/1802.06881.pdf>

Can this be related to enjoyment? How can a bot simulate human enjoyment? Maybe every type of player has a flow graph with his skill + the enemy difficulty. Maybe it has different requirements for activity (pacing).

**How to assign a value to each variable?**

The approach would be to minimize bias when trying to come up with global heuristics that tell us how great any FPS level is and instead let the user define the level’s desired gameplay variables, which are independent between videogames.

There could be scales (sliders) that the user can set up, for each variable, normalized from 0 to 10.

At the polar opposites and throughout the slider there would be markers with other games examples.

In the case of the “activity” metric, an FPS like DOOM Eternal would be close to the maximum value and a game such as CSGO would be much lower.

Somehow, for each of these activities, the proposed value must be researched for the provided games. Example:

DOOM Eternal has a practical average of 2 seconds between combat engagements and CSGO has 20 seconds.

The user would then think of an estimate of their level and adjust the slider on a range from 2 to 20 seconds, say 10 seconds for an average paced level.

Afterwards, the game would measure the actual average between combat engagements and would compare it to the desired result.

The more it deviates, the lower the score, and thus this value depends on the user itself. If any value is above or below the scale, it would be a major penalty since it results in an unbalanced and impractical situation.

Of course, the level and the players can also be tweaked to fit these requirements.

**How to relate all aspects in one formula?**

How do we calculate each variable’s score?

For the input variables, If there is a 0 → 10 scale for each variable and the user inputs a 6, the difference between the input and the actual result is calculated. Say there is a difference of 2, then the score for that variable is a 100% - 20% = 80%, unless the result goes beyond the scale (lower than the minimum or higher than the maximum) then a score penalty is applied depending on how much the value exceeds the limit (exponential).

For other variables like heatmaps… ??? An option would be how spread out or condensed are these heat values, do they correspond with objective location, etc

Regularly spread-out deathmaps should be the norm for a free-for-all level, whilst other game modes should paint a different picture.

Weapons are more straight-forward. The default interpretation would be for a map to be played almost as effectively with each weapon type, but on different strategic locations. We could ignore CSGO since it involves an economy system where certain weapons are much better than others.

How do we weigh each variable’s importance?

**More research on what makes an FPS level a great one. links, studies, talks...**