**State of the Art:**

Comparison between manual playtesting and automated playtesting in the field of level design & prototyping.

Automated playtesting is not an industry standard per se. Concretely, this technique is usually used in simple puzzle games. Moreover, other techniques such as procedural level generation are more established. Nonetheless, the free to use video game engine *Unity* offers a possibility for developers to generate thousands of different playthroughs:

<https://blogs.unity3d.com/2020/12/11/automate-your-playtesting-create-virtual-players-for-game-simulation/?utm_source=linkedin&utm_medium=social&utm_campaign=ml_global_generalpromo_2020-12-11_virtual-player-game-simulation-blog>

Some commercial videogame examples in the link above

There are, however, some studies that offer innovative proposals to achieve automation in regards to level design and prototyping:

* Top-down image analysis, CNN...

<http://antoniosliapis.com/papers/learning_the_patterns_of_balance_in_a_multi-player_shooter_game.pdf>

* Genetic algorithms, CNN...

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

* Genetic algorithms, player personas, MCTS...

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

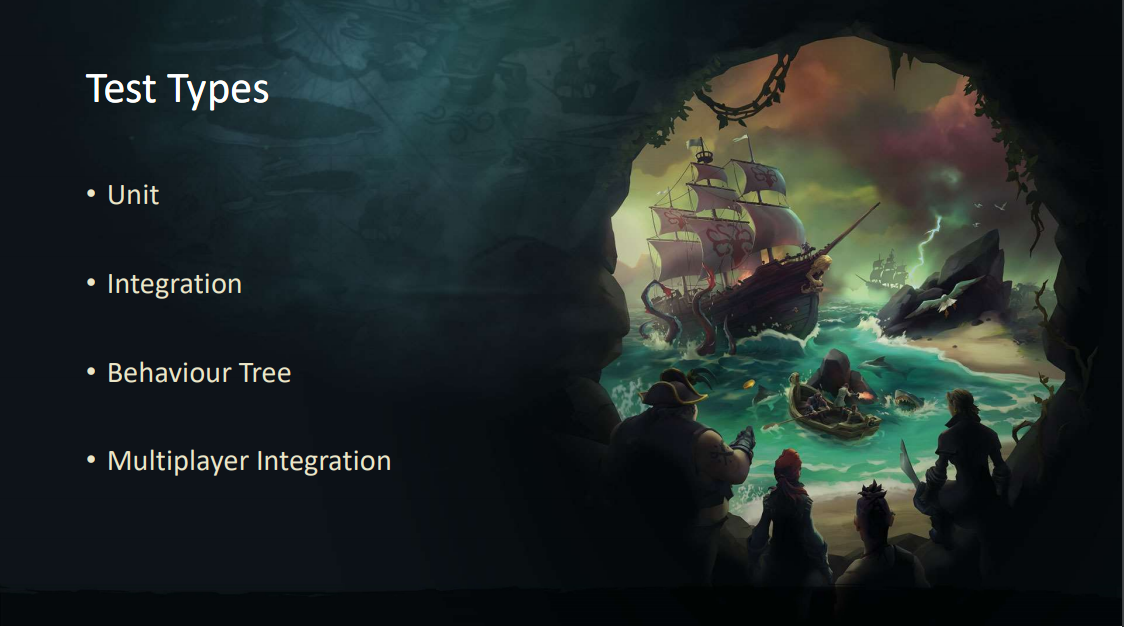
(see <https://docs.google.com/document/d/1WpLUIbCL8hgcdgnn96Lio83sYd74hQuyEbpNouuXwtw/edit>)

A real-world, triple A example, would be *Sea Of Thieves*:

<https://www.gdcvault.com/play/1026366/Automated-Testing-of-Gameplay-Features>

<http://eej.dk/gain/2017/slides/AUTOMATED_TESTING_FOR_MULTIPLAYER_GAME-AI_IN_SEA_OF_THIEVES.pdf>

<https://www.youtube.com/watch?v=KmaGxprTUfI>



* Unreal Engine
* Unit, Integration, BT, Multiplayer integration

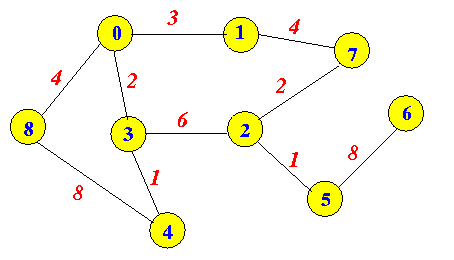
LOL, Riot Games, isolated mechanic testing:

<https://technology.riotgames.com/news/automated-testing-league-legends>

**Level Design Variables & Parametrization**

The idea in this block is to analyze different 3D Action video game genres and how each of them has a slightly different weighted graph of level design variables.

* Said variables would belong both to the level and to the AI that navigates it.
* Case Studies



* Each node in the graph would be a “level design variable”. Since there are connections between nodes, some variables influence others, but not all of them per se.
* *Example: “cover” can be a variable with a high value in some shooters and a low value in beat ‘em ups. In this case, “mortality” would be another variable that directly depends from cover, but not exclusively, since in a shooter there can be grenades that bypass a high cover value. Particularly, cover would influence mortality but not the other way around.*
* By representing different action genres by means of a weighted graph, one can create a model where it is plausible to detect patterns between genres and therefore discover if they follow a similar or polarized topology.
* The ultimate purpose in an automated playtest is to interpret these values as end products of evaluations inside the code, so that the evaluated level has a “deviation score” for each value (positive or negative), that is to say, *eg. a “cover” score is calculated and compared to the model for shooter games to then calculate how much it differs.*
* At the same time, to evaluate these variables, there is a need for another set of values, the ones pertaining to the logic (AI) that will evaluate the level, since depending on the player, a level can be perceived subjectively differently.

**… Rational Level Design (RLD)**

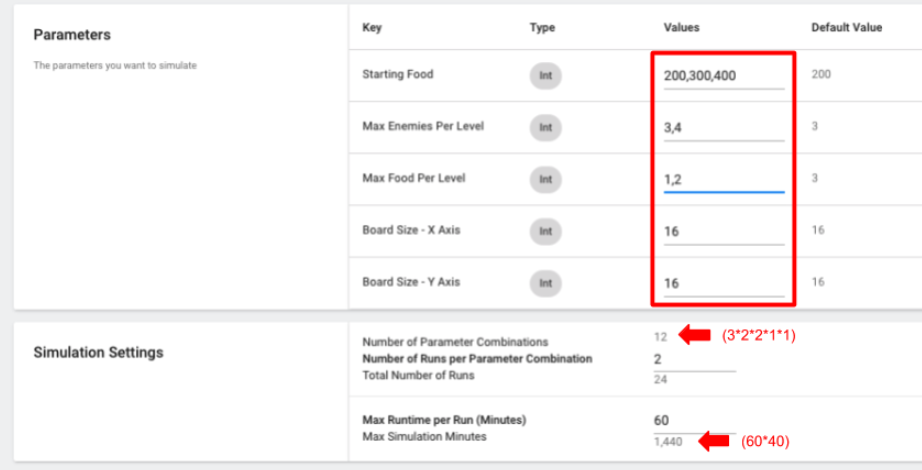
<https://www.gamasutra.com/blogs/LukeMcMillan/20130806/197147/The_Rational_Design_Handbook_An_Intro_to_RLD.php>

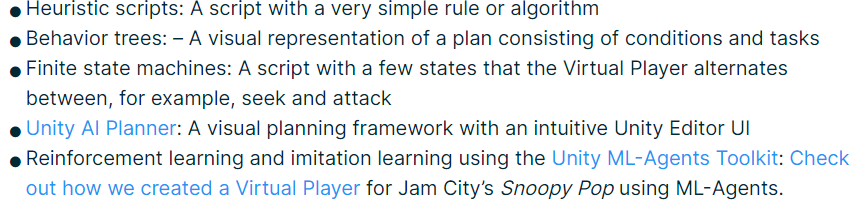
**AI**

* Therefore, multiple player personas, with different objectives, come into play. (<https://arxiv.org/pdf/1802.06881.pdf>**)**

This AI can be achieved with multiple techniques, although the common denominator is the fact that the AI’s variables are the ones that need to be interpreted as input in the automated playtest model, eg by using unity’s automation.

* In conclusion, level design variables would be the metrics and the AI variables would be input paramters.





Personalities:

<https://www.gamasutra.com/view/feature/6474/personality_and_play_styles_a_.php?print=1>

And how to create personas & evolve them:

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

**Data Validation (idea)**

Compare the automated testing prototype with human playtesting

**My game plan:**

* Study parametrization in different 3D action genres. Come-up with weighted graph models (via case studies, concrete data)
* Translate this graphs into a function (how “fun” or “ideal” the level is)
* Gather or create a map based on RLD (Rational level design).
* Gameplay: AI that can navigate the level & fight, basic items (coins, health…) and enemies, as well as basic logic (exit door, winning when all enemies dead…)
* Create the tests for the level.

<https://blogs.unity3d.com/2020/12/11/automate-your-playtesting-create-virtual-players-for-game-simulation/?utm_source=linkedin&utm_medium=social&utm_campaign=ml_global_generalpromo_2020-12-11_virtual-player-game-simulation-blog>

1. Parameters = AI variables (could also try with “can use certain type of weapon” to evaluate weapons, if there are many)
2. Metrics = the model’s node values (time to complete, mortality, etc)

*“The framework we’ve been developing focuses on three key elements of a game – metrics, test cases, and parameters. Metrics are the results of a playthrough that are important to the balance or health of the game. Test cases are the different scenarios that you’d like to measure within a game. Finally, parameters are the configurations that can change your game, thus directly impacting your metrics.”*

* Bonus: gather data that cannot be processed via the testing package, eg maps: heatmap, deathmap...
* Evolve the AI via procedural personas, so that AI can follow different player types and therefore the tests have varied input parameters:

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

* Compare results vs manual playtesting. For the manual playtesting, collect the data the same way (the code for gathering data is the same but instead of an intelligent bot, there is a human player controlling it).
* Validation: Since the level was created via RLD, tweak some props, enemies, etc., we can analyze the test results (how much do they differ from expected metrics) to try to manually change some of those elements. Re-test. After each iteration, the results should get closer to the expected model metrics.