# Can an Unknowing Participant distinguish between Procedurally Generated and Human Designed Interiors?

Thomas O'Leary

***Abstract*—What's the problem? What am I looking at? How does that help solve the problem?**

**Opening, Challenge, Action, Resolution**

**Attempt to see if procedural generated interiors can be percieved as human designed. Comparing the two together and see if participants prefer the procedurally generated designs.**

## I. Introduction

Urban open world games such as Grand Theft Auto V [1], The Division [2] and Batman: Arkham Knight [3] have such large built-up areas for players to venture in. However, only a few handpicked buildings in these large cities are accessible and have modelled interiors leaving the others to be blocked off for decorative purposes. This could be resolved by modelling and designing each room in these cities, but this would become incredibly impractical. Other issues with this can lean towards rendering and the storage of such heavily dense areas.

### *Procedural Generation*

Procedural Generation (PCG) refers to automatically creating content using algorithms [4]. PCG has many applications in video games, some notorious examples being the world/cave generation in Minecraft [5], the texture [6] and world generation [7] in Spore [8] and the procedural texture and music generation in .kkrieger [9].

Using PCG, this largely time-consuming task of designing room interiors can be automated. **And can possibly help maintain a player's immersion within the game.** An issue with this however is that PCG tool's can be seen as boring and repetitive [10].

Through my literature review though I have found many implementations and techniques of Procedural Interior Generation (PCIG), none of these get compared to Human designed interiors.

This study looks to see if a participant is able to tell the difference between Human designed and AI generated interiors.

## II. Literature Review

My literature review consists of two parts that I believe to be important to my research question. It first describes different implementations of Procedural Interior Generation (PCIG) then explores ways in which Artificial Intelligence (AI) is compared to Humans.

### *A. Implementations of Procedural Interior Generation*

Although PCG has a lot to show and offer in game development - being used for characters, terrain, weapons and textures - the use of Procedural Interior Generation (PCIG) in games however is scarcely come by.
A game that does use PCIG is Catlateral Damage [11], a small indie game developed by Manekoware where you play as a cat on a destructive rampage in its own house. In 2017, Chris Chung (the indie developer behind the game) wrote a case study about the level design in his game [4]. When developing Catlateral Damage, Chung was undecided on how to design the levels and ultimately went for PCIG [10]. Before the interior decoration can take place, a Squarified Treemap algorithm is used to generate the room layouts and floor plans within the level [12]. Each room generated from this algorithm has an associated data file, this file contains the type of furniture available, maximum type of each furniture, spawn probabilities, if the furniture is placeable on walls and if the player will spawn there. The furniture objects that can be placed have physics components attached, to allow them to be accurately placed within the level - for this, a Rectangle Packing algorithm is used to place these objects within allocated surface areas on the floor and other furniture objects. Concluding the case study, Chung states that most players could not notice that the levels were procedurally generated - **although this is a promising statement, Chung has not shown any evidence to back this claim.**

Despite there not being many implementations of PCIG in games, there are however a handful of published papers that have used their own techniques to emulate room interiors.

*Multi-Agent System:* In 2009, T. Germer and M.Schwarz sought out to procedurally arrange a rooms' furniture in real-time. [13]. They did this by producing a Multi-Agent based solution where each furniture object, in a given room, is seen as an individual agent that seeks a suitable parent furniture object. The tool has 3 main requirements: [13]

- Generate furniture arrangements rapidly
- Arrangements must be persistent
- Must be plausible and interesting

As stated earlier, each agent seeks a suitable parent in a given room - once found it would place and orient itself accordingly. These agents have custom semantic descriptions to allow them to create different object layouts, an example listed by the authors is a chair - a chair can either be set next to a table/desk but can also be isolated in its own surroundings leading it to have many possible parent objects [13].
Each agent has only 3 states:

1) Search
   - All agents start in the *Search* state, they begin by searching for possible parent objects - if a parent is found that suits its semantics the agents' state changes to *Arrange*, if a parent can't be found at all the agent is deleted. Different parts of the room are set as the root parent for agents - floor, walls, windows and doors.
2) Arrange
   - In the *Arrange* state, the agent attempts to place and orient itself with the parent accordingly. Whilst doing so, it has to check for collisions with other agents in which the Separating Axis Thereom is used [14] - if no collisions are found the agents' state changes to *Rest*.
3) Rest
   - In the *Rest* state, potential child agents are now able to seek this object as a parent. If the resting agents parent moves, the resting agent will move along with it - however if this move results in a collision, its parent is lost and the agents state is changed back to *Search*.

The rooms are responsible for the agents. They manage all existing agents in the room by keeping a list with their corresponding type - allowing the agents to easily identify potential parents. The room is only created if a player comes within a specified range, when the player is no longer in this range it gets removed from memory. The issue with this is that if a player were to re-enter said room, it will be furnished completely differently. The authors fixed this issue by saving a numerical value (also known as a seed) to the furnished layout - so if a player were to re-enter, the numerical value would be applied to furnish the room exactly as it was.

YouTube video demonstrating the implementation [15]
*Rule-Based Layout:* Paper [16]
*Statistical Relationships:* Paper [17]
*Constraints:* Paper: [18], uses SUNCG dataset [19]

On 29th April 2021, Sony Interactive Entertainment published patent US20210121781, titled "AI-Generated Internal Environments Based On External Geometry" [20]. The patents' description goes onto explain a Machine Learning (ML) tool that takes in data from the external structure of a virtual building and generates an interior environment just from this data. Although this is just a patent for an ML tool, this could be the start of PCIG being used in AAA Titles.

### B. Artificial Intelligence Compared to Humans

This is going to be a little more difficult to right about, as I haven't read a paper on this so far. And I have only managed to find 3 papers that talk about this, but I am not sure that they could be entirely relevant.

## III. Research Question

From the above sources, I have formed **actual research question that's totally not wip anymore**

### A. hypothesis & null hypothesis

**Hypothesis stuff...**

## IV. Artefact

### A. What will be made

AI that procedurally generates interior at runtime in a pre-defined room size and access to pre-made furniture assets. Will be later compared with human designed interiors (being given the same room size and assets)

### B. How will I ensure Quality

Quality control. Roadmap? Unit Testing? Integration testing?

### C. How will I create it

The AI will be made in the Unity game engine (Version 2020.3.12f1) **cite unity engine**

### D. Why will this answer the questions

The artefact itself will contribute towards answering the research question.

## V. Research Methodology

### A. Experimental Design

A/B test. Paper uses something similar [18].

### B. Limitations

Time, resources

### C. Sampling Plan

Sample size, sampling method

### D. Data management plan

Managing, collecting, & storing data

### E. Data Analysis

Something to do with R

### F. Ethical Considerations

I plan to not commit war crimes I promise

## VI. Appendix

Data analysis code, supporting screenshots, system development life-cycle, list of unit tests & testing plan

## REFERENCES


[1] Rockstar North, "Grand Theft Auto V," [PC Game] Rockstar Games, Steam, 2013.

[2] Massive Entertainment, "The Division," [PC Game] Ubisoft, Steam, 2016.

[3] Rocksteady Studios, "Batman: Arkham Knight," [PC Game] Warner Bros. Interactive Entertainment, Steam, 2015.

[4] J. Togelius, E. Kastbjerg, D. Schedl, and G. N. Yannakakis, "What is procedural content generation? mario on the borderline," in *Proceedings of the 2nd International Workshop on Procedural Content Generation in Games*, ser. PCGames '11. New York, NY, USA: Association for Computing Machinery, 2011. [Online]. Available: https://doi.org/10.1145/2000919.2000922

[5] Mojang, "Minecraft," [PC], 2011.

[6] D. g. DeBry, H. Goffin, C. Hecker, O. Quigley, S. Shodhan, and A. Willmott, "Player-driven procedural texturing," in *ACM SIGGRAPH 2007 Sketches*, ser. SIGGRAPH '07. New York, NY, USA: Association for Computing Machinery, 2007, p. 81–es. [Online]. Available: https://doi.org/10.1145/1278780.1278878

[7] K. Compton, J. Grieve, E. Goldman, O. Quigley, C. Stratton, E. Todd, and A. Willmott, "Creating spherical worlds," in *ACM SIGGRAPH 2007 Sketches*, ser. SIGGRAPH '07. New York, NY, USA: Association for Computing Machinery, 2007, p. 82–es. [Online]. Available: https://doi.org/10.1145/1278780.1278879

[8] Maxis, "Spore," [PC Game] Electronic Arts, Steam, 2008.

[9] Farbrausch, ".kkreiger," [PC Game] Farbrausch, 2004.

[10] T. Short and T. Adams, *Procedural Generation in Game Design*. CRC Press, 2017.

[11] C. Chung, "Catlateral Damage," [PC Game] Manekoware, 2015.

[12] F. Marson and S. Musse, "Automatic real-time generation of floor plans based on squarified treemaps algorithm," *International Journal of Computer Games Technology*, vol. 2010, 09 2010.

[13] T. Germer and M. Schwarz, "Procedural Arrangement of Furniture for Real-Time Walkthroughs," *Computer Graphics Forum*, vol. 28, no. 8, pp. 2068 – 2078, 2009.

[14] S. Gottschalk, M. Lin, and D. Manocha, "OBBTree: A Hierarchical Structure for Rapid Interference Detection," *Computer Graphics*, vol. 30, 10 1997.

[15] T. Germer, "Procedural Arrangement of Furniture for Real-Time Walkthroughs," [Online] Youtube, June 2009. [Online]. Available: hhttps://youtu.be/xwVUknGeycQ

[16] T. Tutenel, R. Bidarra, R. Smelik, and K. J. de Kraker, "Rule-based layout solving and its application to procedural interior generation," in *Proceedings of the CASA Workshop on 3D Advanced Media in Gaming and Simulation (3AMIGAS)*, 01 2009.

[17] L.-F. Yu, S.-K. Yeung, C.-K. Tang, D. Terzopoulos, T. F. Chan, and S. J. Osher, "Make it home: Automatic optimization of furniture arrangement," in *ACM SIGGRAPH 2011 Papers*, ser. SIGGRAPH '11. New York, NY, USA: Association for Computing Machinery, 2011. [Online]. Available: https://doi.org/10.1145/1964921.1964981

[18] P. Henderson, K. Subr, and V. Ferrari, "Automatic generation of constrained furniture layouts," *arXiv: Computer Vision and Pattern Recognition*, 2017.

[19] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser, "Semantic Scene Completion from a Single Depth Image," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 190–198.

[20] W. Benedetto, Sony Interactive Entertainment Inc., "AI-Generated Internal Environments Based On External Geometry," Patent Number: US20210121781, April 29 2021. [Online]. Available: https://patentscope.wipo.int/search/en/detail.jsf?docId=US322909393&_fid=WO2021081415