Can an Unknowing Participant distinguish between Multi-Agent Designed and Human Designed Interiors?

Thomas O'Leary

***Abstract*—This dissertation is aimed towards answering the research question ”Can an Unknowing Participant distinguish between Multi-Agent Designed and Human Designed Interiors?”. Before reaching this question I discuss the current implementations of Procedural Interior Generation in both video games and in papers produced from those of academic backgrounds. To help answer my question, a research study was conducted with 56 participants where each participant took part in a 5 minute 2-staged A/B test. In each stage, participants were shown 5 pairs of room interiors - one room being human designed and the other being designed by the artefact. In the first stage, they were informed to select the room they prefer and in the second stage they were informed that 1 room in each pair was Artefact designed, they were then told to select which one they believe was not human-made.**

***Index Terms*—Procedural Generation (PCG), Procedural Interior Generation (PCIG), Multi-Agent (M-A)**

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 due to time constraints. 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 admirable 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, Chapter 2].

Through my literature review, I have found many implementations and techniques of Procedural Interior Generation (PCIG) but only a few of these implementations are compared with Human Designed interiors to check its authenticity. My research is aimed towards testing a PCIG tool against human designed rooms in a user study. More specifically a Multi-Agent system that creates a rooms' interior.

II. Literature Review

My literature review consists of two parts that I believe to be important to my research question. It first describes the use of PCIG in games and then explores the different implementations and techniques of PCIG that have been published.

A. Procedural Interior Generation in Games

Although PCG has a lot to show and offer in game development, 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 developer behind the game) wrote a case study about the level design in his game [10, Chapter 6]. When developing Catlateral Damage, Chung was undecided on how to design the levels and ultimately went for PCIG [10, Chapter 6]. 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, containing information such as furniture available and maximum type of each furniture. 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 [13] is used to place these objects within allocated surface areas on the floor and on top 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.

On 29th April 2021, Sony Interactive Entertainment published patent US20210121781, titled ”AI-Generated Internal Environments Based On External Geometry” [14]. 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 more commonly in the Games industry.

B. Implementations and Techniques of Procedural Interior Generation

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. I will explain the key elements of these implementations and will give my view on what does and doesn't work with these implementations.

1) Multi-Agent System: In 2009, *T. Germer, et al.* [15] sought out to procedurally arrange a rooms' furniture in real-time. The aim for this system was to allow the quick generation of a rooms' interior while a player is walking around a building - a live demonstration of this can be viewed on YouTube [16]. This system involves a Multi-Agent (M-A) based solution where each furniture object, in a given room, is seen as an individual agent that seeks a suitable parent furniture object. These agents have custom semantic descriptions to allow them to create different object layouts. Each agent has 3 states:

1) Search
   - All agents start in this 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.
2) Arrange
   - In the *Arrange* state, the agent attempts to position itself with the parent accordingly. Whilst doing so, the Separating Axis Theorem [17] is used to check for collisions - if no collisions are found the agents' state changes to *Rest*. If a collision does occur however, it must attempt to re-position itself with the parent.
3) Rest
   - In the *Rest* state, potential child agents are now able to seek this object as a parent. If the 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*.

Although a large proportion of the rooms furnishing is handled by the agents themselves, a big drawback is that the system requires a lot of user input before this can happen. Every room must have clear user defined data and every object type must have manually defined semantic descriptions. If starting from scratch this process can take a long time, but each object type only requires a singular semantic description. Some objects with matching behaviors can also share semantics - this creates a lot of flexibility when designing the agents as they are completely autonomous of one another [15]. Another issue with this implementation is that the system is never evaluated based upon how realistic or natural the furniture arrangements are based upon human designs.

2) Rule-Based Layout: A Rule-Based layout approach was proposed in 2009, users would be able to specify what objects can be placed within a layout - these would represent an instance of a class and contain certain rules on how it should be placed [18].

Rules can be defined in multiple ways - they can be associated specifically with an object class or defined in the layout planner. An example of defining a rule with an object class, as told by the authors, is by setting a rule for a sofa to always face an instance of a TV. The layout planner is responsible for sending objects to the solver. The planner can have custom rules to allow it to be applicable to different room layouts (living room, factory floor, waiting room). It also has a backtracking rule that is only triggered if an object of interest is not placeable. If this is the case, the planner would backtrack to place previous objects in different positions to allow this object to be placed.

The solver is given an object from this planner and the current layout. With this, it finds all possible locations for the new object - these locations are based upon the rules of this object and the rules already set in place for existing objects in the layout. The possible locations then take specific parameters into account, such as the amount of clearance an object requires or if an object's area is off limits (for example its bounding box) [18]. A Minkowski Sum [19] is carried out containing these inaccessible areas and removed from the list of possible locations. With this completed, the object is then given a list of all possible locations in the layout it can be placed.

A good approach taken for this implementation was the use of abstract classes - to allow easier expansion of new furniture. One issue I could see with this however is accidentally creating the same rules on multiple types of classes (perhaps on classes that do not derive from a similar parent). This implementation is also never tested for realism, to check if the layouts produced by the system are plausible and deemed natural (*of human design*).

3) Constraints: P. Henderson, et al. [20] presented a data driven system that learns from the SUNCG [21] database to generate furniture layouts. This database contains over 45000 apartment layouts that are designed by humans, from this database - 2500 models are categorised into 170 furniture object classes. Their system is presented in such a way that it can be left to be fully automated [20], but does allow flexibility with the user allowing them to change constraints within the layout.

These constraints include:

- Room size, shape & type
- Exclusion of Object classes
- Furniture clearance
- Locations of specified furniture
- Locations of doors & windows

A user study was carried out in their paper, where 1400 pairs of layouts are presented to 8 non-experts in an image format [20]. These participants are asked to identify the layout that has a more realistic/natural setting. In each pair one layout is from their system, the other being human designed - the order in which the images are shown is randomised per pair. In this study, both constrained and unconstrained layouts are put to the test. For unconstrained layouts, they were presented in 2 different styles; 1st person and an overhead view. Layouts that were presented in 1st person, were seen to be slightly preferred over the human designs and layouts presented in the overhead format were seen to be almost identical to human designs. Constrained layouts were only presented in the overhead format, but two sets of constraints were used:

i. Fixed room size & fixed placement of a singular object
ii. Fixed room size & fixed door/window locations

When referring to the results of (i), the constrained layouts were seen to be almost identical to that of human design. Whereas the results of (ii), showed that human designs were preferred over the constraints.

The use of the user accessible constraints for this approach allows a lot of freedom when designing the layouts and the use of some of these constraints proves that it can be seen as human design [20].

4) Statistical Relationships: In 2011, a PCIG system using statistical relationships was proposed [22] [23]. This system uses 3 types of object relationships and utilizes these to calculate the *cost* of the current arrangement until one is of satisfactory price. The spatial relationship represents the objects distance and orientation to its nearest wall. The hierarchical relationship represents a child/parent relationship between objects - for example a candle (child) placed on a table (parent). The pairwise relationship represents the interaction (distance and orientation) between different pairs of objects (TV and a sofa). The cost function is used to quantify the realism or functionality of the state of the furniture arrangement. The higher the cost of an object, the higher the priority it takes. There are 5 stages to the cost function - each stage has an individual weighting and adds to the overall cost of the objects' arrangement.

- Accessibility
  - Every object must be accessible in the 3D space. Each object has a defined *Accessible Space* as well as *Bounding Box*. If another object enters an *Accessible Space*, the cost increases.
- Visibility
  - Certain objects must be viewed from a specific direction - these objects are given a *Viewing Frustum* (some objects include TV's and paintings). Similarly to the Accessibility cost function, whenever another object obstructs a *Viewing Frustum*, the frustums values are passed in and the cost increases.
- Pathways
  - Within the furniture arrangements, pathways are created using *Cubic Bézier curves*. These curves are represented as rectangles in the 3D space. The cost function is similar to that of Accessibility & Visibility, yet in this case the control points of the *Bézier curve* are used as the positional value and applied to the rectangles used to represent the pathway in the arrangement.
- Prior Spatial Relationships
  - The prior Spatial relationship (distance and orientation of the nearest wall) is subtracted from the objects current Spatial relationship.
- Pairwise Relationships
  - Similar to the Prior cost function, the pairwise cost function is defined to subtract the distance and orientation of the paired objects.

To create the arrangement with the cost function, a mixture of Simulated Annealing and the Metropolis-Hastings (M-H) algorithm is used. Simulated Annealing originates from the physical process Annealing used to heat objects to remove defects and slowly bring the object back down into a low-energy state [24] - in this system, it is used for the placement of the furniture. At first the objects are "*heated up*" to allow for more freedom whilst they are arranging until they "*cool down*", with each temperature decrement the cost function is called to evaluate the current iteration of the arrangement. With every iteration (or "*temperature decrement*"), the M-H algorithm [25] is used to compare the previous and the proposed new arrangement.

To see if the use of the cost function did produce furniture arrangements with a realistic/functional state, the system was put to the test in a perceptual study against human designed interiors. 25 volunteers (14 of which stated that they did not have any expertise in interior design) were used in this study and were unaware of its true purpose. Each participant viewed a total of 35 pairs, in each pair containing a synthesized and a human designed arrangement. The participants were told to select the furniture arrangement that they would prefer. The synthesized arrangement would only be considered to have been the victor if the human designs were not shown as the "clear" winners within the results. Of the 35 pairs shown to the volunteers, only 13 synthesized arrangements were seen to be the preferred choice.

The use of statistics to evaluate the relationships of each object in the layout is the most unique technique that I have read in my literature review. This would allow the system to work in full autonomy if all cost functions were to be set at a high weighting [22]. As the nature of the perceptual study was just to test the authenticity of the synthesized arrangements, one key factor that may have been overlooked is the plethora of potential experience of those who did have experience/expertise in interior design. I believe by showing different data of those with and without experience, the results produced from the study may have been more insightful.

III. Research Question & Artefact

Through my literature review, there is evidence to show that PCIG should be compared to human designs more frequently to test its authenticity. With this I propose the following research question; *Can an Unknowing Participant distinguish between Multi-Agent Designed and Human Designed Interiors?*
Given this, I also propose the following hypotheses to investigate in order to help me answer the research question.

A. Hypotheses

1) When unaware, the Artefact (*M-A System*) is picked more often than Human designed interiors by participants

2) When notified, the participant is not able to distinguish between Human and Artefact (*M-A System*) interiors

B. Artefact

The artefact proposed to help answer my research question is a Multi-Agent (M-A) system that is used to create a room's furniture arrangement where each piece of furniture is seen as an individual agent. These individual agent's use the Unity's [26] ScriptableObject class to contain all necessary information about themselves. This information includes data such as the type of furniture they are, the occupation of each of their sides and what other furniture types they can have as a potential parent - see Fig 1 for an example of an Agents ScriptableObject. The agents are represented in a scene using pre-fabricated assets from a free-to-use Unity asset pack by Brick Project Studio [27].

To generate a room's layout, the agents must have access to an empty room. In order for the furniture to be arranged, a user must place the furniture that they want to be placed in the Unity scene. Once the scene is running, the agents handle and place themselves accordingly within the room. All agents begin in a *SEARCH* state, within this state the agent aims to find a potential parent that is defined within its ScriptableObject. Once a parent is found, the agent begins its *ARRANGE* state where it places and orients itself appropriately. Once placed, the agents state changes to *REST*. The behaviour of a singular agent used in the Artefact is also demonstrated in Fig. 9. The behaviour behind the agents in the artefact is influenced by the work from *T. Germer, et al.* [15].

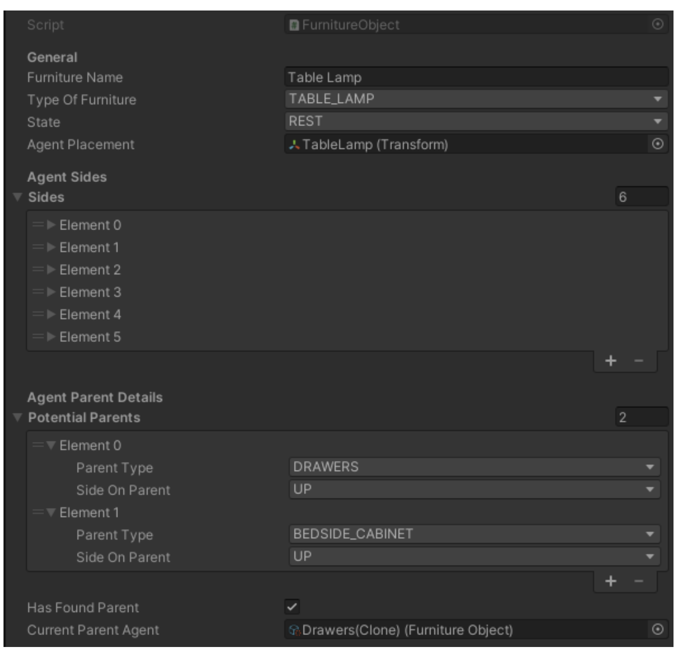

Fig. 1: Screenshot of an agent's ScriptableObject

C. Artefact Development & Quality Assurance

For the implementation of the Artefact, the Unity Game Engine [26] was used as I have used this engine throughout my time at University and have thorough experience with using this engine in creating both small and large projects. Furthermore, the knowledge I have in the C# language and Unity's base classes helped further in deciding what engine to use for the Artefact. Throughout the Artefact's development lifecycle, I also closely followed an Agile methodology and it's principles to help me slowly but iteratively develop a finished product. Using Agile to help with the artefact's development was a clear choice for me as it is commonly used in Software Engineering and Game Development environments alike [28], but I have also followed this methodology closely throughout my time at University, using it when working on other small or large projects. I worked in bi-weekly sprints and set myself achievable tasks in-order for my Artefact to progress further along in its development.
Throughout the Artefact's development, along with Agile, it was continuously tested using Unity's Test Framework [29] and C#'s *NUnit Framework* [30] and a pilot test was conducted to ensure the research methodology of this dissertation. (More details of Unit testing and the Pilot test can be found in Appendix D).

IV. Research Methodology

In order for me to test my hypotheses and answer the research question, the artefact was put to the test in a research study that involved 56 participants taking part in an estimated 5-minute test. The data collected in this short test was then analysed using the R programming language in a series of Z-Tests.

A. Experimental Design

For my experimental design, a participant was shown 5 pairs of room layouts in the form of a 2 Staged A/B test. In the first stage the participant was informed to pick, out of each pair, which room furniture layout they preferred. This preference may have been influenced by realism, authenticity or what environment they would rather be in. In each pair, one layout was that of human design and the other was designed by the Artefact (M-A System) - an example of this is shown in Fig 3. The order in which these layouts were shown were randomised per stage and per participant. Once 5 furniture arrangements were selected, the participant would move onto the second stage of the study - a notice would appear notifying the participant they have reached Stage 2, as shown in Fig 4. Just before beginning they were notified that in every pair within the first stage, 1 furniture layout was created by the Artefact (M-A System). Their challenge in the second stage of the study would be to select, out of the same 5 pairs they were initially shown, which one they believe was made by the Artefact (M-A System).

This experimental design was used as previous works by *P. Henderson, et al.* [20] and *L.-F. Yu et al.* [22] carried out similar perceptual studies requiring a participant to pick between one from their designed model and human designs. In 2010, Margaret Boden proposed a new variant of the Turing Test (TT) that is oriented around artistic creativity [31]. With her TT, for an art program to pass it would have to:

1) Be indistinguishable from human produced artwork And/Or
2) Be seen having similar aesthetic value to human produced artwork

I found this variant of the TT as a helpful source when designing the experimental design for my research study.

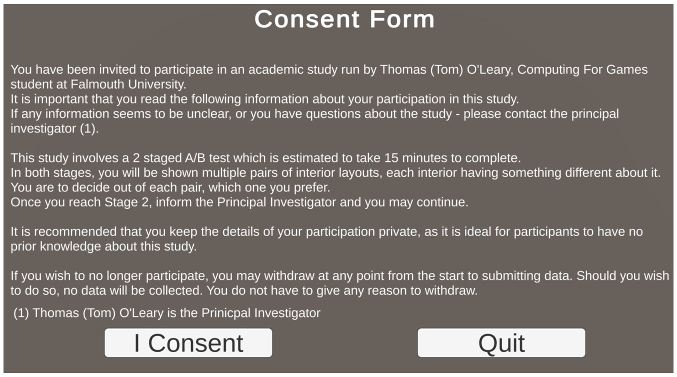

Fig. 2: The consent page before the start of the A/B test that the participant reads.

Fig. 3: A pair of furniture layouts used in the Research Study. One being human designed and the other being Artefact designed.

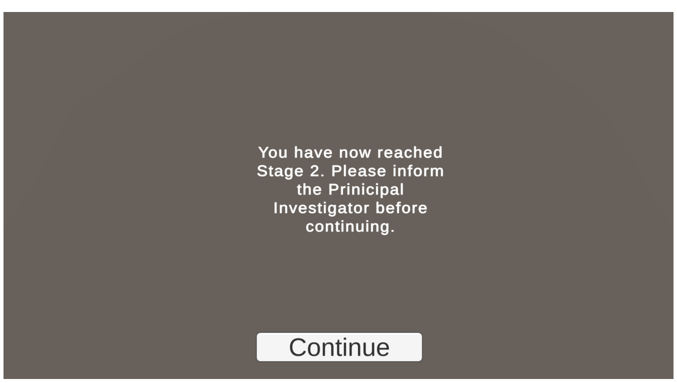

Fig. 4: Screenshot of the Stage 2 notice, where participants are notified of the involvement of the Artefact.

B. Participants

Using G*Power to calculate a sample size for my research study, it was estimated that in order for my data to be significant with 95% confidence - I required 54 participants to be involved in my research study (with an effect size of 0.5). The participants involved in my research study were primarily students from within the Games Academy at Falmouth University - as this is where the Artefact had under-gone its development and was the easiest solution to gain participants due to the time limitations of this research.

C. Data Collection & Management

For my data analysis, the main data required was whether or not the Artefact was picked in each pair and stage. In order for me to obtain this, each time a participant selected a room in a pair, a method would be called to write to a CSV file I had set up for my data collection. This method would write the participant ID, what stage of the study they were currently in, the pair ID, the name of the 2 rooms and a boolean that returned appropriately if the Artefact was picked in that pair. As explained in *Ethical Considerations*, no personal information is collected in the research study, nor can any information collected in the research study can be used to identify participants - signifying that General Data Protection Regulation (GDPR) [32] does not need to be followed. Results collected from the study however were exported and stored in a CSV file - see Fig 5. This file was password encrypted to prevent any third party intervention.

PARTICIPANTID	STAGE	PAIRID	ARTEFACTNAME	HUMANNAME	ARTEFACTPICKED
1	1	5	aRoom5	hRoom5	True
1	1	2	aRoom2	hRoom2	False
1	1	1	aRoom1	hRoom1	True
1	1	4	aRoom4	hRoom4	False
1	1	3	aRoom3	hRoom3	True

Fig. 5: CSV file used to store the data collected from the Research Study

D. Ethical Considerations

As the research study requires human participants this creates a medium ethics risk according to the Falmouth University Ethics Board. To facilitate this risk, a Falmouth University ethics form was completed and signed off by the project Supervisor in consultation with the Head of Subject. The artefact itself is of low risk/concern as it is not used for militarization and participants are able to opt out at any point during their participation. No personal information about the participants involved is collected nor can any data collected in the research study be used to identify participants - the EU's General Data Protection Regulation (GDPR) [32] does not need to be followed due to the nature of the research study. However, to protect the participants rights, the Nuremberg Code will be followed to keep and ensure this research study is ethically sound [33]. All participants will be handed a Participant Information Sheet that details the key information they must know before the study, a consent form is also supplied to ensure they have agreed to participate. Participants are still able to withdraw at any point until submitting data.

COVID-19: At the current state of the pandemic and following the latest Government Guidelines in England [34], participants were not required to wear a face covering however they would not be questioned if they preferred to do so. All surfaces were sanitised between participants.

Computer Related Injuries: Due to the nature of the study taking place on a Computer, all forms of computer related injuries were taken into account. Participants had the opportunity to appropriately set the computers screen and seat positioning to the best ergonomic setting for themselves before starting the study. Participants were also informed that they could take a short break whenever they pleased.

V. Data Analysis

With the data I collected, a total of 56 participants were involved in my research study - giving me a total of 560 results to analyse (280 per stage). This data collected was then split into two separate CSV files each holding the appropriate data regarding each stage. The data was used in a series of Z-Tests in R (See *Appendix E*) and appropriate charts were generated to visualise the data in each stage (See Fig 6 and Fig 7). Due to the large sample size from the data, the Binomial distribution is approximated with the population mean $\mu = np$ (where n is the total sample size and p is the probability) and standard deviation $\sigma = \sqrt{np(1-p)}$.
With this, the Z-Test is calculated like so (where x is the amount of successful trials in the Binomial Test) [35, Chapter 6]

$$z = \frac{(x-\mu)}{\sigma}$$

A. Hypothesis Results

1) Hypothesis 1: When unaware, the Artefact (M-A System) is picked more often than Human designed interiors by participants.
In this stage participants were required to select, out of each pair, which room they preferred - to test this hypothesis an upper-tailed Z-Test was used. This test was used to check to see if the Artefact interiors were selected significantly more than human designs. The Z-Test returned a value of 0.84, when calculating the P value it returns as 0.2005. With this we have failed to reject the null hypothesis - as the value is not statistically significant enough at $P < 0.05$.

2) Hypothesis 2: When notified, the participant is not able to distinguish between Human and Artefact (M-A System) interiors.
In this stage, participants were notified of the Artefact and were told to select, out of the same 5 pairs they saw in Stage 1, which room they believe was made by the Artefact. By completing another upper-tailed Z-Test to check if human designs were selected more than the Artefacts, we return a Z score of -1.67 and can calculate a P value of 0.9525. With this, we have again failed to reject the null hypothesis as it is not significant enough at $P < 0.05$.

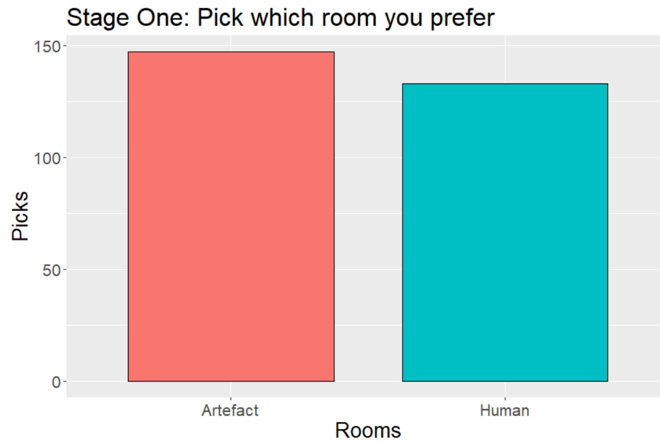

Fig. 6: Bar chart representing the data from Stage 1

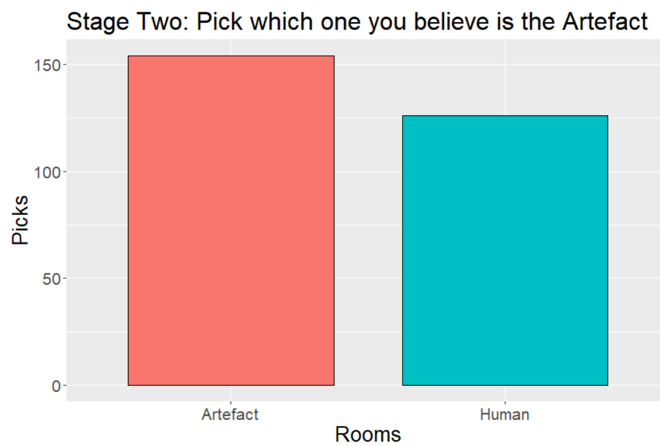

Fig. 7: Bar chart representing the data from Stage 2

VI. Discussion

Regardless of the upsetting results of both Hypotheses failing to reject the null, the information collected from my research study can still prove to be of some use as a preliminary test to help towards future studies regarding comparisons between Procedural Interior Generation and Human design.

Looking at Fig 6, we can see that Artefact interiors are chosen more than Human interiors. In this stage, the Artefact was selected 147 times whereas the Human interiors were selected 133. Although not statistically significant, this does show some evidence to say that the artefact was preferred over human designs in the research study.

When looking at Fig 7, we can quite clearly see that participants selected Artefact interiors a lot more than human. In this stage, the Artefact was selected 154 times whereas the human rooms were selected 126. If *Hypothesis 2* were to be changed to that of *"When notified, the participant IS able to distinguish between Human and Artefact (M-A System) interiors"*, this would have lead to a lower-tailed Z-Test checking if the Human designs were picked less than

Artefact designs. Resulting in a Z-Score of 1.67 and a P value of 0.0475, at $P < 0.05$ the null hypothesis would have been rejected thus accepting the alternate.

VII. Issues & Limitations

A. Viewing format in Study

Although no points were made during the Pilot Study of this research regarding the viewing format of the room interiors - many participants during the research study verbally stated that they found the viewing format "off-putting". By this they were referring to the way the rooms were represented on screen (See Fig 3). Some believed that they couldn't see enough of the room to justify picking one or the other and ultimately struggled in deciding what rooms to pick in both stages of the study.

Mentioned earlier in justifying my experimental design, previous works by *P. Henderson, et al.* [20] and *L.-F. Yu et al.* [22] also carried out perceptual studies requiring participants to pick between human designed interiors and those designed from their own models.

In the perceptual study ran by *P. Henderson, et al.* [20] (Constraint based approach), multiple viewing formats were applied to represent the room interiors depending on what type of room was generated. When displaying unconstrained room types to participants, both overhead and first person view formats were used. Regarding results for these layouts, an overhead viewing format was seen to be almost identical to human and a first person viewing format was slightly preferred.

However, in the perceptual study ran by *L.-F. Yu et al.* [22] (Statistical approach) 3 viewing formats were used to represent the layouts - an overhead and 2 different views from the corners of the room. The results showed that in 3 of the paired comparisons, the human designs were not clearly preferred over the synthesized arrangements.

Perhaps by showing multiple viewing formats of both Artefact and Human designed interiors - it may have made for a much more comfortable experience for some participants, and we may have seen different choices made in both stages of the study.

B. Participants

Out of the 56 participants involved in the research study, 53 of these were students from the Games Academy (Falmouth University). With a good sense of game development/design and a general census of what is achieved by others within the Games Academy, some participants may have been able to predict the involvement of a non-human element to the research study before being notified of the Artefact in Stage 2. This possible prediction skews the expectation of a participant not having prior knowledge of the Artefact's involvement, possibly further skewing results. To help with this, it would have possibly been more beneficial for the study to have a wider range of participant knowledge surrounding digital games and Artificial Intelligence (AI) - branching outside the Games Academy for participants. But due to the strict time schedule I had set myself for this research project, asking students within the Games Academy to be participants in order to reach my sample size was the easiest option available.

VIII. Conclusion & Future Work

As discussed in Section VII, particular elements of the research study can be adjusted to possibly administrate more beneficial data in whether participants can distinguish between Multi-Agent designed and Human designed Interiors.

Inspired by how *P. Henderson, et al.* [20] and *L.-F. Yu et al.* [22] handle it in their work - implementing different viewing frustums for the layouts in the experimental design, could potentially open an alley for more thoughtful decisions made by participants as they have more information to take in regarding each room layout in both stages of the study. It could allow them to analyse the room in more depth and perhaps visualise the room with much more ease with the multiple perspectives.

Another aspect of the study that could be adjusted is the selection of participants. By expanding the participation selection pool, a much varied range of data could have been collected due to the varied knowledge of participants - potentially having participants with varied knowledge in digital games, varied knowledge of what AI is and potentially varied knowledge in interior design which could have a much bigger impact in the first stage of the study.

To conclude, this research paper sought out to see if an unknowing participant was able to distinguish between Human and Artificial Intelligence designed (through the use of a Multi-Agent system) interiors. 5 pairs of both types of interiors were held to the challenge in a 2 staged A/B test that gained the data from 56 willing participants. In the first stage, participants were informed to select which room they preferred in each pair and in the second, participants were informed of the involvement of the Artefact and to proceed in picking which interior layout was made by this Artefact. Although both hypotheses failed to reject their respective nulls, the data collected from this research study can be used as preliminary results going forward to help any future studies surrounding Procedural Interior Generation and its comparison to that of Human design.

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] E. Huang and R. E. Korf, "Optimal Rectangle Packing: An Absolute Placement Approach," *Journal of Artificial Intelligence Research*, vol. 46, pp. 47–87, 2013.

[14] 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

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

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

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

[18] 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.

[19] "CGAL 5.3 - 2D Minkowski Sums." [Online]. Available: https://doc.cgal.org/latest/Minkowski_sum_2/index.html

[20] P. Henderson and K. Subr and V. Ferrari, "Automatic Generation of Constrained Furniture Layouts," *Computer Vision and Pattern Recognition*, 2017.

[21] 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.

[22] 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

[23] HKUST VGD, "SIGGRAPH 2011 - Make it Home: Automatic Optimization of Furniture Arrangement," YouTube, April 2011. [Online]. Available: https://youtu.be/vlDoSv6uDKQ

[24] A. Zhigljavsky and A. Zilinskas, *Stochastic Global Optimization*, P M. Pardalos and Ding-Zhu Du, Ed. New York Springer, 2008.

[25] S. Chib and E. Greenberg, "Understanding the Metropolis-Hastings Algorithm," *The American Statistician*, vol. 49, no. 4, pp. 327–335, 1995.

[26] Unity Technologies, "Unity," 2021. [Online]. Available: https://unity3d.com/unity/whats-new/2020.3.12

[27] Brick Project Studio, "Apartment Kit," 2022. [Online]. Available: https://assetstore.unity.com/packages/3d/props/apartment-kit-124055

[28] C. Keith, *Agile game development with Scrum*. Upper Saddle River, NJ : Addison-Wesley, 2010.

[29] Unity Technologies, "Unity Test Framework," 2021. [Online]. Available: https://docs.unity3d.com/Packages/com.unity.test-framework@1.1/manual/index.html

[30] Microsoft, "Unit testing C# with NUnit and .NET Core," 2021. [Online]. Available: https://docs.microsoft.com/en-us/dotnet/core/testing/unit-testing-with-nunit

[31] M. A. Boden, "The Turing Test and Artistic Creativity," in *Turing Test*, M. Bishop, Ed. Bradford: Emerald Group Publishing Limited, 2010, ch. 1, pp. 409–413.

[32] "General Data Protection Regulation (GDPR)," 2016. [Online]. Available: https://gdpr-info.eu/

[33] "The Nuremberg Code," *Law, Medicine and Health Care*, vol. 19, no. 3-4, pp. 266–266, 1991.

[34] Cabinet Office, "Coronavirus: How to stay safe and help prevent the spread," Dec 2021. [Online]. Available: https://www.gov.uk/guidance/covid-19-coronavirus-restrictions-what-you-can-and-cannot-do

[35] J. Zar, *Biostatistical Analysis*, 4th ed. Prentice Hall, 01 1999.

APPENDIX A - GENERAL

Link to the Ethics form and Participant Information Sheet: https://falmouthac-my.sharepoint.com/:f:/g/personal/to231922_falmouth_ac_uk/EiE3vOcxqLlLuU0_BN7m1NoBCja231kbqKHAxOlqX-sRfw?e=4dc65R

Link to the Computing Artefact GitHub repository: https://github.falmouth.ac.uk/TO231922/computing-artefact

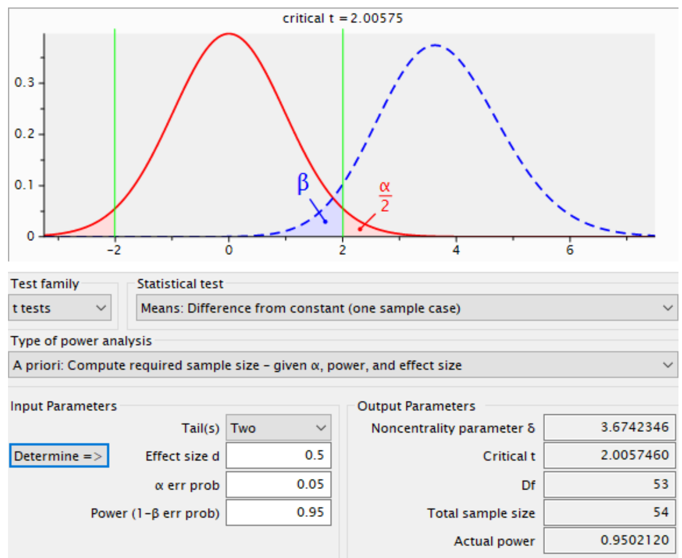

Fig. 8: G*Power Sample Size

Appendix B - Reflective Addendum

Not-so-SMART Objective: Something

Key Component	**Objective**
Specific	Something something something.
Measurable	Something something something.
Achievable	Something something something.
Realistic	Something something something.
Time-Bound	Something something something.

SMART Object: Something smart.

Appendix C - Software Architecture

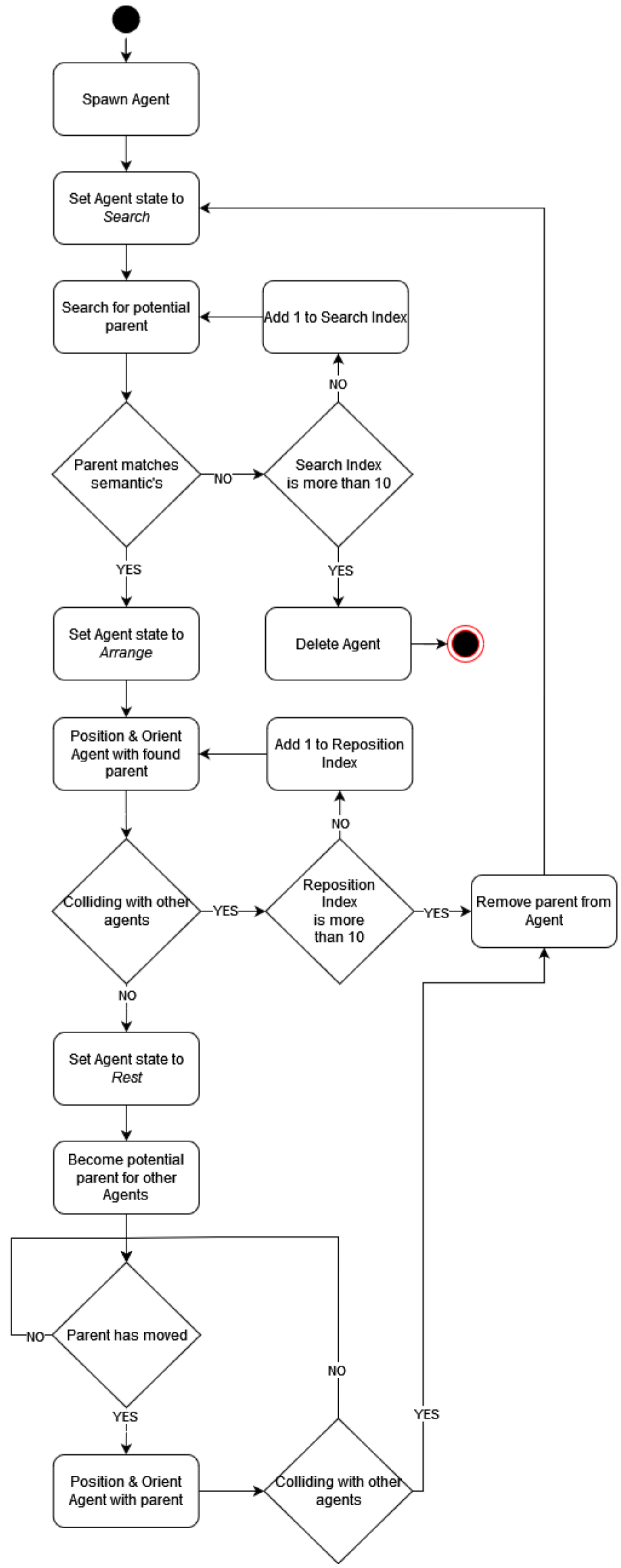

Fig. 9: Agent Behaviour represented in an Activity Diagram

APPENDIX D - TESTING

The Unit tests for my artefact were created using Unity's Test Framework and were used throughout the Artefact's development to validate the code and to ensure the Artefact worked as intended. Unit testing code can be seen in Appendix F and passed Unit Tests can be seen in Fig 10.

A small pilot study was conducted with the help of some BSc peers. This pilot study was conducted to ensure the validity of my experimental design. It consisted of them taking part in my 2-staged A/B test, helping in testing the effectiveness of how information was being represented - any feedback received from this small study was implemented into my methodology and no data from this pilot study was used within my data analysis.

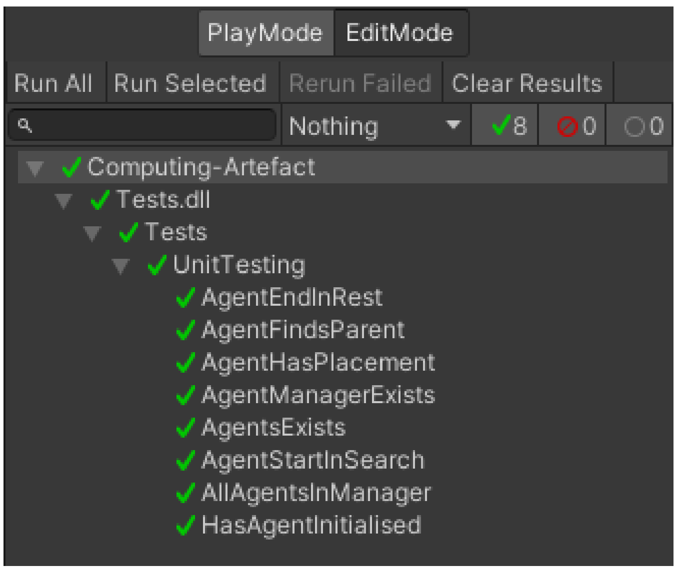

Fig. 10: Unit Tests in the Unity Test Runner

APPENDIX E - R CODE

```r
# Install packages - only needs to be run once
#install.packages("readr")
#install.packages("psych")
#install.packages("pwr")
#install.packages("ggplot2", dependencies = TRUE)

# Load Libraries
library(readr)
library(pwr)
library(ggplot2)

# Load data from Stage One from CSV
stageOneData <- read.csv("D:/R/testr/StageOneData.csv")

# Create empty vector for all Pair picks to go into
s1Picks <- vector()

# Add either Artefact or Human to the vector depending on pick
for (x in stageOneData$ARTEFACTPICKED){
  if (x == " True"){
    s1Picks <- append(s1Picks, "Artefact")
  }
  if (x == " False"){
    s1Picks <- append(s1Picks, "Human")
  }
}

# Turn vector to Factor
s1PicksFactor <- factor(s1Picks)

# Turn Factor to table object
s1PicksTable <- table(s1PicksFactor)

# Apply table values to integers
artefactPicks <- s1PicksTable[1]
humanPicks <- s1PicksTable[2]

# Get proportions from results
s1Prop <- s1PicksTable / sum(s1PicksTable)
s1Prop

# This is our expected proportion (as there are only 2 options)
expProp <- 0.5

# Cohens H to calculate effect size
cohensH <- ES.h(s1Prop[1], expProp)

# For interpretation
h <- abs(cohensH * sqrt(2))

if (h < 0.2){
  print("Negligible Effect Size")
} else if (h < 0.5){
  print("Small Effect Size")
} else if (h < 0.8){
  print("Medium Effect Size")
} else{
  print("Large Effect Size")
}
paste("Effect Size:",h)

##### Z TEST APPROXIMATION FROM A BINOMIAL TEST #####

# Calculating the Standard Deviation
sD <- sqrt((artefactPicks + humanPicks) * expProp * (1 - expProp))

# Calculating the population mean for a large sample size
populationMean <- (artefactPicks + humanPicks) / 2

```

```r
# Calculating the Z score
zScore <- (artefactPicks - populationMean) / sD
paste("Z Score:",zScore)

# Rounding Z score to 2 decimal places (standard)
zScore <- round(zScore, 2)

# Calculating P Value from Z Score
pValue <- pnorm(zScore, lower.tail = FALSE)
# Rounding P Value to 4 decimal places
pValue <- round(pValue, 4)
paste("P Value:",pValue)


##### BAR CHART #####

# Create Bar Chart
df <- as.data.frame(s1PicksTable)
chartColours <- c("Artefact","Human")
df <- cbind(df, chartColours)

barChart <- ggplot(data=df, aes(x=s1PicksFactor, y=Freq, fill=chartColours)) +
  geom_bar(colour="black", stat="identity", width = 0.75) +
  theme(text=element_text(size=20)) +
  guides(fill=FALSE) +
  xlab("Rooms") + ylab("Picks") +
  ggtitle("Stage One: Pick which room you prefer")

barChart
```

Listing 1: R Code for Stage One data.

```r
# Install packages - only needs to be run once
#install.packages("readr")
#install.packages("psych")
#install.packages("pwr")
#install.packages("ggplot2", dependencies = TRUE)

# Load Libraries
library(readr)
library(pwr)
library(ggplot2)

# Load data from Stage Two from CSV
stageTwoData <- read.csv("D:/R/testr/StageTwoData.csv")

# Create empty vector for all Pair picks to go into
s2Picks <- vector()

# Add either Artefact or Human to the vector depending on pick
for (x in stageTwoData$ARTEFACTPICKED){
  if (x == " True"){
    s2Picks <- append(s2Picks, "Artefact")
  }
  if (x == " False"){
    s2Picks <- append(s2Picks, "Human")
  }
}

# Turn vector to Factor
s2PicksFactor <- factor(s2Picks)

# Turn Factor to table object
s2PicksTable <- table(s2PicksFactor)

# Apply table values to integers
artefactPicks <- s2PicksTable[1]
humanPicks <- s2PicksTable[2]

# Get proportions from results
s2Prop <- s2PicksTable / sum(s2PicksTable)

# This is our expected proportion (as there are only 2 options)
expProp <- 0.5

# Cohens H to calculate effect size
cohensH <- ES.h(s2Prop[2], expProp)

# For interpretation
h <- abs(cohensH * sqrt(2))

if (h < 0.2){
  print("Negligible Effect Size")
} else if (h < 0.5){
  print("Small Effect Size")
} else if (h < 0.8){
  print("Medium Effect Size")
} else{
  print("Large Effect Size")
}
paste("Effect Size:",h)


##### Z TEST APPROXIMATION FROM A BINOMIAL TEST #####

# Calculating the Standard Deviation
sD <- sqrt((artefactPicks + humanPicks) * expProp * (1 - expProp))

# Calculating the population mean for a large sample size
populationMean <- (artefactPicks + humanPicks) / 2

# Calculating the Z score
zScore <- (humanPicks - populationMean) / sD
paste("Z Score:",zScore)

# Rounding Z score to 2 decimal places (standard)
zScore <- round(zScore, 2)

# Calculating P Value from Z Score
pValue <- pnorm(zScore, lower.tail = FALSE)
# Rounding P Value to 4 decimal places
pValue <- round(pValue, 4)
paste("P Value:",pValue)



##### BAR CHART #####

# Create Bar Chart
df <- as.data.frame(s2PicksTable)
chartColours <- c("Artefact","Human")
df <- cbind(df, chartColours)

barChart <- ggplot(data=df, aes(x=s2PicksFactor, y=Freq, fill=chartColours)) +
  geom_bar(colour="black", stat="identity", width = 0.75) +
  theme(text=element_text(size=20)) +
  guides(fill=FALSE) +
  xlab("Rooms") + ylab("Picks") +
  ggtitle("Stage Two: Pick which one you believe is the Artefact")

barChart
```

Listing 2: R Code for Stage Two data.

APPENDIX F - UNIT TESTING CODE

```
using System.Collections;
using System.Collections.Generic;
using NUnit.Framework;
using UnityEngine;
using UnityEngine.TestTools;

namespace Tests
{
    public class UnitTesting
    {
        private AgentManager agentManager;
        private GameObject[] agents;
        private int agentAmount;

        [SetUp]
        public void SetUp()
        {
            agentManager = GameObject.Find("Room").GetComponent<AgentManager>();
            agents = GameObject.FindGameObjectsWithTag("agent");

            agentAmount = agents.GetLength(0);
        }


        // Checks if the AgentManager exists in the scene
        [Test]
        public void AgentManagerExists()
        {
            Assert.IsNotNull(agentManager);
        }

        // Checks if all Agents are in the AgentManager
        [Test]
        public void AllAgentsInManager()
        {
            var tempManager = new GameObject().AddComponent<AgentManager>();

            if (tempManager.furnitureInScene.Count == agentManager.furnitureInScene.Count)
            {
                Assert.IsNotNull(tempManager);
            }
        }


        // Checks if there are Agents in the scene
        [Test]
        public void AgentsExists()
        {
            Assert.IsNotNull(agents);
        }

        // Checks to see if the Agents are initialised
        [Test]
        public void HasAgentInitialised()
        {
            foreach (var agent in agents)
            {
                if (agent.GetComponent<Agent>().agentSO.isInitialised)
                {
                    Assert.IsNotNull(agent.GetComponent<Agent>().agentSO.isInitialised);
                }
            }
        }

        // Checks to see if all Agents have placement transform set
        [Test]
        public void AgentHasPlacement()
        {
            foreach(var agent in agents)
            {
                if (agent.GetComponent<Agent>().agentSO.agentPlacement)
                {
                    Assert.IsNotNull(agent.GetComponent<Agent>().agentSO.agentPlacement);
                }
            }
        }


        // Checks to see if Agent is in SEARCH
        [Test]
        public void AgentStartInSearch()
        {
            foreach(var agent in agents)
            {
                if (agent.GetComponent<Agent>().agentSO.state == AgentState.SEARCH)
                {
                    Assert.IsNotNull(agent);
                }
            }
        }

        // Checks to see if Agent finds a parent
        [Test]
        public void AgentFindsParent()
        {
            foreach(var agent in agents)
            {
                if(agent.GetComponent<Agent>().agentSO.hasFoundParent)
                {
                    Assert.IsNotNull(agent);
                }
            }
        }

        // Checks to see if Agent is in REST
        [Test]
        public void AgentEndInRest()
        {
            foreach(var agent in agents)
            {
                if (agent.GetComponent<Agent>().agentSO.state == AgentState.REST)
                {
                    Assert.IsNotNull(agent);
                }
            }
        }
    }

}
```

Listing 3: Unit Test Code used to validate the Artefact's code.