Targeting Horror via Level and Soundscape Generation

Phil Lopes, Antonios Liapis and Georgios N. Yannakakis
Institute of Digital Games, University of Malta, 2080 Msida, Malta
{louis.p.lopes,antonios.liapis,georgios.yannakakis}@um.edu.mt

Abstract

Horror games form a peculiar niche within game design paradigms, as they entertain by eliciting negative emotions such as fear and unease to their audience during play. This genre often follows a specific progression of tension culminating at a metaphorical peak, which is defined by the designer. A player's tension is elicited by several facets of the game, including its mechanics, its sounds, and the placement of enemies in its levels. This paper investigates how designers can control and guide the automated generation of levels and their soundscapes by authoring the intended tension of a player traversing them.

Procedural content generation (PCG) is an extensive area of game research, and is often used as an effective method to reduce content creation costs and increase game longevity (Togelius et al. 2011). However, research in procedural audio is uncommon in the field (Collins 2013), likely due to the additional demands that sound often requires. Digital games are multi-faceted creative domains (Liapis, Yannakakis, and Togelius 2014), where facets such as audio, visuals, levels and game mechanics work in conjunction to create interactive digital experiences (Lopes and Yannakakis 2014). This paper investigates the interplay between level design and sound, where designers define the player experience while the system generates levels and their respective soundscapes to accommodate the designer's intentions.

The survival horror genre is unique in its heavy reliance on sound to convey negative affective states such as shock, disgust, ecstasy, fear and relief (Ekman and Lankoski 2009). It also focuses on exploration and hiding as players have limited combat ability (e.g. no weapons or limited ammunition). These complex characteristics of player affect raise important challenges for the generation of levels and soundscapes, where the focus is in evoking these types of emotions. For instance, an interesting question is how a level generator can anticipate and influence the affective state of a player, while consistently balancing feelings such as stress and relief; or how players navigate through a level under the effects of stress caused by previously encountered monsters. This paper tackles some of these challenges by exploring the generation of levels and soundscapes in the survival horror genre while also operating on a simplified player model, simulating horror gameplay during level traversal.

This paper presents an extension to the *Sonancia* prototype (Lopes, Liapis, and Yannakakis 2015), a system capable of generating multiple facets of horror games. Levels consist of rooms in a haunted mansion that are procedurally generated via genetic search, while audio snippets are distributed throughout the rooms (i.e. level sonification). Earlier work focused on generating the architecture of the level autonomously based on the distance of a singular path. This paper focuses on the generation of levels which include added gameplay elements (i.e. monsters, quest items), and on the generation of their soundscapes, where notions of *tension* and *suspense* are used to drive the level and soundscape generative process respectively.

Sonancia allows designers to define the flow of relaxation and tension through the generated level. To do this, the system requires two tension curves: the desired (designer-specified) tension curve (DTC) and the actual (level-based) tension curve (LTC). The system searches, via a genetic algorithm, for room setups where the actual tension curve more closely matches the desired one, so that generated levels and their respective soundscapes follow the designer specifications as closely as possible, while still maintaining a degree of variability.

Related Work

Digital games have been using PCG techniques for over 30 years, since games such as *Rogue* (Toy and Wichman 1980). More recently PCG has been a focus of academic interest, including the development of alternative PCG approaches (Togelius et al. 2011) and the adaptation of content to specific player experiences (Yannakakis and Togelius 2011). Often level generation is the focus of PCG research, with notable examples including the generation of 2D platform levels (Shaker et al. 2012), real-time strategy maps (Liapis, Yannakakis, and Togelius 2013), racing tracks (Togelius, De Nardi, and Lucas 2007), first-person shooter maps (Cardamone et al. 2011), among others (Shaker, Togelius, and Nelson 2015). The horror genre is no exception to PCG; games such as *Daylight* (Zombie Studios, 2014) procedurally generate levels and enemy positions, while the AI director of *Left 4 Dead* (Valve, 2008) procedurally spawns zombies according to a tension model and the progression of players in the level. *Sonancia* draws inspiration from the tension model of *Left 4 Dead*, as the level structure and monster placement are generated based on a designer-specified progression of tension.

Digital games, especially those in the horror genre, rely on game audio as it enhances the player experience; the soundscapes created by game audio are capable of immersing players into the virtual world (Collins 2013; Gasselseder 2014). Although some arguments can be made that digital games already apply some form of procedural audio, such as the sounds of player actions in the background of multiplayer games (Garner and Grimshaw 2014), much more could be accomplished by orchestrating between virtual levels and the sounds played therein. Several professional tools such as the sound middleware of *UDK* (Epic Games, 2004) provide procedural sound components, albeit very simple (i.e. variations of notes in a specific scale). This shows an increasing commercial interest in sound as a procedurally generatable game facet.

On the other hand, games such as *Audio Surf* (Fitterer, 2008) and *Vib Ribbon* (Sony Entertainment, 2000) have previously focused on music-driven level generation, where the characteristics of the music influence the level generation. *Proteus* (Key and Kanaga, 2013) explored several ideas on how spatial positioning, visuals and player interaction affected and influenced sounds played in realtime. *AudioInSpace* (Hoover et al. 2015) is another example that combines both gameplay and audio within a side-scrolling space shooter that evolves its shooting mechanics based on the music playing in the background, which is pre-selected by the user or procedurally generated via artificial evolution. Scirea et al. (2014) have also investigated how music could be procedurally generated in order to convey narrative foreshadowing in digital games. This paper, instead, concentrates on developing methodologies capable of generating horror game levels and their corresponding sonification based on various tension models.

Methodology

Sonancia is a multi-faceted content generation tool for a horror game taking place in a procedurally generated haunted manor. The objective of the player is to collect an item deep within the manor, while at the same time outrunning or hiding from monsters that lurk inside it. Players do not have weapons and for this reason must avoid direct confrontation. In the current version of *Sonancia* players must only reach the main quest item to complete a level; in later versions of the game, players may be required to return to the entrance in order to complete the level. *Sonancia* is composed of two main modules: level generation and level sonification.

Generating Levels

Haunted manors consist of different rooms, separated by walls, and doors that interconnect them. Rooms are described by their room ID, and players start the game at the room with the lowest room ID. Levels are also populated by monsters and quest items (see Figure 1); the latter can be the main quest item (collecting it completes the level) or sidequest items which are optional. If more than one quest items exist in the level, the one furthest from the player's starting room becomes the main quest item.

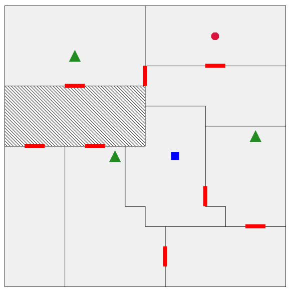


Figure 1: Generated manors consist of a set of rooms and interconnecting doors. Thin black lines represent walls, while thick red lines represent doors. Monsters are represented as green triangles, the main quest item is represented as a blue square, and sidequest items as pink circles. The darker room is where the player starts the level from.

To efficiently create levels that follow a designer-specified tension curve (DTC), a search-based PCG approach (Togelius et al. 2011) was chosen. *Sonancia* uses a mutation-based genetic algorithm (GA) using roulette wheel selection. The level architecture is represented by an array of integers. Each integer maps to a specific tile of the level, and represents which room occupies that tile. Doors are represented as tuple objects, describing which rooms are interconnected. Monsters and quest items are tuple objects describing their type (i.e. item or monster) and the room they are placed in.

Mutation can shift a room's walls, divide rooms, connect rooms with doors or remove doors (two rooms can only be connected with one door), move spawnpoints (ensuring one monster per room and one item per room) or add new monsters. Mutation chances depend on the operator; for instance, moving items affects the critical path and has a lower mutation chance as it is disruptive to the evolutionary progress. After mutation is applied, a flood fill algorithm ensures that rooms are not disconnected and that rooms are larger than 5 tiles; if not, the gene is repaired to assimilate small or disconnected rooms with adjacent ones, moving items or monsters as needed. Elitism is set to retain the fittest 3% of the population in the next generation.

To determine the quality of a level, the fitness function considers both the room layout (*structure fitness*) and the distribution of monsters between rooms (*tension fitness*). Both fitness dimensions are described below. When evolving levels, the two fitnesses are added to determine which individual will be selected for mutation.

A level's structure is evaluated based on the shortest path from the player's starting room to the main quest item; this path is the *critical path* of the level (see Figure 2). Levels with a critical path going through as many unique rooms as possible receive higher scores. Structure fitness (f_s) is calculated as $f_s = R_s - P_s$, where R_s is the number of rooms that are uniquely traversed by each path from the start to the main quest item as well as sidequest items, which discourages linear levels; P_s is the number of rooms with no doors,

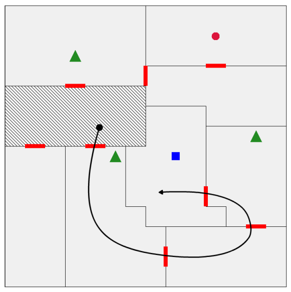


Figure 2: The critical path of a level, from the starting room to the main quest item.

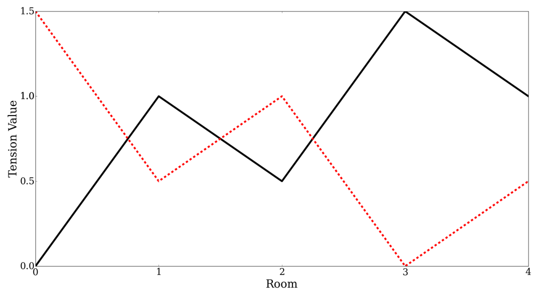


Figure 3: The LTC of the path of Figure 2 (black solid line) and its corresponding suspense curve (red dotted line); the latter is obtained by inverting the LTC.

which penalizes disconnected, unusable rooms.

Unlike other game genres, horror games attempt to create a sense of unease that slowly builds up over time (Cheong and Young 2008). Tension is visualized as a tension curve, allowing designers to define the intended rise and fall of tension in the level (DTC). The x-axis of a tension curve is the number of rooms visited along the critical path, while the y-axis is the tension value of each room. A generated level creates its own tension curve (LTC) by following the critical path, increasing tension by 1 when a room contains a monster while decreasing tension by 0.5 when it does not (see Figure 3). Matching DTC with LTC is done by fitting the intended tension curve to the number of rooms available, then evaluating their similarity via the tension fitness (f_t):

$$f_t = \sum_{i=0}^{r} 1 - \mid L_i - T_i \mid \tag{1}$$

where L_i and T_i are the level-based and designer-specified tension values of room i, respectively, and r is the total number of rooms on the critical path.

Note that DTC also acts as a designer constraint for the maximum number of rooms on the critical path: levels with more rooms than the x-axis of DTC receive a f_t value of 0.

Sonifying Levels

Level sonification consists of both the selection and allocation (i.e. before play begins) and the mixing (i.e. during play) of sound assets. When selecting sounds, *Sonancia* specifically takes into account the LTC, so that a level's soundscape is influenced by its structure; this ensures coherence between the level and audio facets. *Sonancia* has access to a soundbank of approximately 50 human-authored recordings with an average length of 4 seconds, where all assets are tagged according to *instrument*, *note*, *octave* and *suspense value*; the suspense value is an empirical measure of how dramatic that particular sound is perceived by human listeners. These tags inform *Sonancia* about the characteristics of each asset, allowing it to select sounds according to minor, major or dissonant scales.

Audio Allocation: To increase audio fidelity and avoid breaking player immersion, it is critical to select sounds in an efficient manner and distribute them throughout the generated level appropriately. *Sonancia's* audio allocation is a two-stage process consisting of *instrument-based* and *suspense-based* audio selections.

The **Instrument-based** audio selection phase consists of choosing an asset via roulette wheel selection based on its instrument tag. Each instrument is given the same probability of being selected at the start of the process, which is halved each time that instrument is chosen. This selection algorithm also ensures that no recording is chosen twice, to avoid repetition, and provides instrument variability.

The **Suspense-based** audio selection phase consists of distributing the assets chosen during the previous phase throughout the level based on their suspense tag and the *level suspense curve*, which is the inverted LTC (see Figure 3).

Each audio asset is placed in a room within the generated level, and each room can only have one audio asset. The algorithm starts by creating two arrays: one with the selected audio assets and another with the rooms on the critical path. The audio asset array is sorted based on the value of each sound asset's suspense tag, while the room array is sorted based on the level suspense curve. Assets and rooms in the same position in their respective arrays are associated, i.e. the sound asset is placed in that room. This ensures that a consistency between both the audio assets and the rooms are kept (i.e. high suspense audio assets are placed in rooms with a high value in the level suspense curve).

Audio Mixing: The mixing algorithm controls how the sounds selected by the process of audio allocation are played during the game. To produce in-game relevant results, audio mixing requires real-time information of the current game state (e.g. where the player is at the time). The current version of *Sonancia* employs a mixing rule which controls the volume of sound depending on the player's position. The audio volume increases (or decreases) exponentially the closer (or furthest) the player is from a neighbouring room. This mixing rule allows players to hear the sound from a room they are headed to, offering a sense of foreshadowing.

Experiments

For the purpose of testing the efficiency of the *Sonancia* system, experiments with different ad-hoc designed tension curves (DTCs) and map sizes are reported in this section.

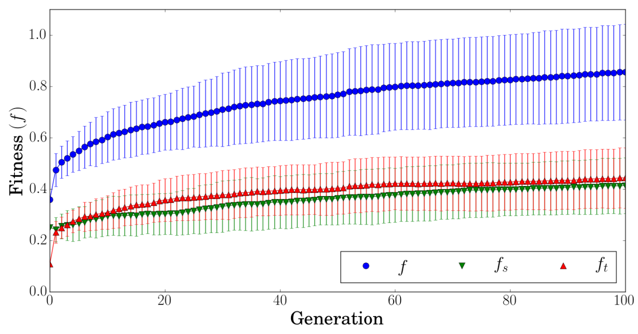


Figure 4: Evolution of the total fitness (f) and its components f_s and f_t for the tension curve depicted in Figure 5(e). Values are averaged across 100 GA trials; error bars show standard error.

Tension Curve Variations

To test how evolved levels match a desired tension curve, four experiments were performed using four different curves (see Figure 5). The GA runs for 100 generations in 100 independent trials for each experiment. Indicatively, Figure 4 shows the best individuals' fitness values obtained for the curve depicted in Figure 5(e). Similar trends of fitness convergence were found in all experiments with tension curves of Figure 5, but are not presented here for brevity.

Figures 5(a)–5(d) show different levels generated by *Sonancia* for a corresponding desired tension curve. The level of Figure 5(a) attempts to match a V-shaped tension curve (Figure 5(e)). The level contains 7 rooms on the critical path, and therefore the desired tension curve is scaled along the x-axis from 10 to 7 rooms. As the tension fitness attempts to balance the placement of monsters to fit the DTC, monsters are gathered towards the start and the end of the critical path. However, a monster is also placed in the midst of the critical path (bottom of the level) in order for LTC to be non-zero in the next few rooms due to tension decay (and thus match the non-zero tension of the DTC in those rooms).

The level of Figure 5(b) uses the inverse tension curve (see Figure 5(f)), and contains 9 rooms in its critical path. As expected from the DTC, the first two and the last three rooms on the critical path do not contain monsters; instead monsters are concentrated towards the middle of the path, in the attempt to match the inverted V-shaped tension curve.

The level of Figure 5(c) uses a "wave" tension curve with two peaks (see Figure 5(g)). This level has 9 rooms on its critical path; monsters are more evenly distributed along it. The GA clearly attempts to balance monster-based tension escalations and de-escalations to match the DTC as closely as possible. A particularity of the level of Figure 5(c) is the room in the upper right corner which is inaccessible. This indicates that the penalty value in the structure fitness is not high enough to prevent the generation of such levels.

Finally, the level of Figure 5(d) is generated through a linear tension curve (as depicted in Figure 5(h)) and contains 8 rooms on its critical path. Monsters are concentrated at the end of the critical path, although monsters are also placed in other parts of the level in order to provide non-zero tension

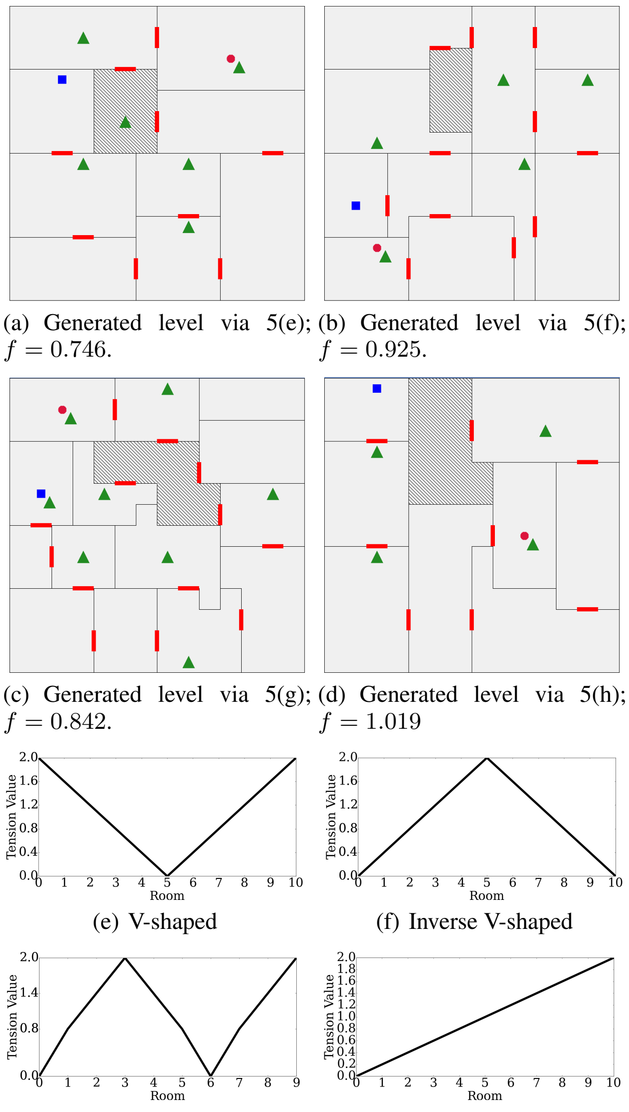

(a) Generated level via 5(e); $f = 0.746$.
(b) Generated level via 5(f); $f = 0.925$.
(c) Generated level via 5(g); $f = 0.842$.
(d) Generated level via 5(h); $f = 1.019$
Tension Value
Room
(e) V-shaped
(f) Inverse V-shaped
(g) Inverse V-wave-shaped
(h) Linear

Figure 5: Generated levels (a, b, c, d) using different designer-authored tension curves (e, f, g, h). Darker rooms represent the players' starting room; green triangles represent monsters; pink circles represent sidequest items and blue squares the main quest item.

values to the rooms after it (due to tension decay). The last room on the critical path contains no monsters, as the two monsters in previous rooms result in a high tension in the last room, despite the tension decay.

Suspense Sonification: For the interested reader the sonification samples based on the levels depicted in Figure 5 are available online[1].

[1]`https://goo.gl/cBPiMk`

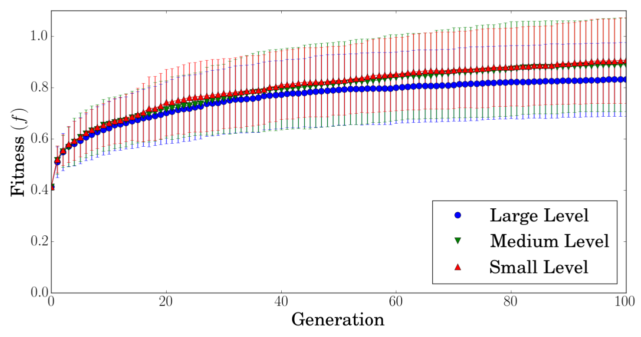


Figure 6: Evolution of the total fitness (f) across three different level sizes using the linear DTC of Figure 5(h). Values are averaged across 100 GA trials; error bars show standard error.

Map Size Variations

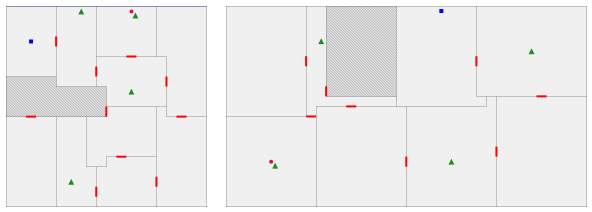
To test the efficiency of *Sonancia* across different level shapes and sizes, three types of level are tested: a small level (14x14 tiles) as per previous experiments, a medium level (20x20 tiles), and a large level (20x24 tiles). For all level types we run *Sonancia* for 100 generations and 100 independent trials; all experiments reported in this section use the linear tension curve as depicted in Figure 5(h).

Figure 6 shows the evolution of the best map's fitness score across the three level sizes (averaged from 100 independent runs). Based on the optimization behaviours of the different map sizes, level size does not seem to affect the performance and convergence of the level generation process.

Two examples of medium and large levels are shown in Figure 7. Figure 7(a) depicts the fittest medium level with a critical path of 8 rooms. Medium levels, in general, did not present any structural or tension-relevant differences compared to generated small rooms. Figure 7(b) shows the fittest larger level with a critical path of 7 rooms. Larger levels achieve lower fitnesses (although not significantly); they contain fewer rooms on the critical path, which consequently makes the tension fitness under-perform. The many tiles in large levels allow the GA to create more rooms but at the same time make it difficult to satisfy the critical path constraint (i.e. the upper limit on rooms specified in the DTC). Moreover, the wall shift mutation has a minimal effect on the layout of rooms in the large level, as it shifts only one row of tiles back and forth; this explains why rooms in larger levels are more symmetrical and rectangular.

Discussion

Initial results of *Sonancia* show the potential of generating and sonifying levels based on a designer-defined progression of tension. In all situations the GA attempted to balance monster distribution and decay effects to approximate the desired tension curve as closely as possible. Interesting and unexpected patterns also occurred, due to the inability of perfectly matching the LTC to the DTC; this allowed for a degree of control while still providing variability. Future experiments will explore more granular approaches, by adding



(a) Medium level; $f = 0.907$

(b) Large level; $f = 0.789$

Figure 7: The fittest levels of different sizes generated using the linear tension curve of Figure 5(h)

and modifying parameters for tension and decay specifically for levels. One idea includes the placement of "soundspawn" game objects with different tension values (depending on the sound), which would be an addition to the current ones (1 and 0.5). This could potentially allow the GA to follow the designer's tension curve more closely, while adding an extra sonification layer obtained through evolution.

Although initial results seem promising from an algorithmic standpoint, the current version of the system was specifically created to test the feasibility of designer influenced facet blending. However, thorough experiments with users (i.e. players) and eventually designers will be required to test the accuracy (with players) and usefulness (with designers) of the system and the quality of generated content. Players' level traversals and experienced affective states (via self reports) will be collected during play so that a comparison can be made between their personal and the designer-intended experience. An example will be to compare levels with audio assets selected through our methodology and at random. Experiments with designers will also help in evaluating both usability and the quality of the generated content.

While levels created by *Sonancia* (see Figure 7) show interesting level variations, some weaknesses limit the potential effects of tension curves on generated levels. One of the main downsides of evolutionary algorithms is the delicate balance between giving a reward or a penalty to generated artefacts (Michalewicz 1995). *Sonancia's* evolutionary process still struggles in creating levels that reach the total number of rooms defined by the designer. This may be caused by the aggressive death penalty that evolved levels incur if the total number of rooms goes over the room threshold, which can potentially eliminate good candidates. This is also true for inaccessible rooms, as the GA can produce highly fit levels containing these due to the low penalty such rooms incur.

The biggest limitation of the sonification methods in *Sonancia* is the use of specific sound recordings. While audio fidelity is greatly enhanced by using human-authored recordings, there is a degree of control that is lost when manipulating real-time sound. While the audio assets in *Sonancia* are still approximately 4 seconds long, the system has less flexibility in influencing the sound signal in such a way that it does not sound broken to the end-user. In sound design this is referred to as *sound granularity*; more granular, very short sound snippets makes sound easier to manipulate (Stevens and Raybould 2013). As playing sounds in

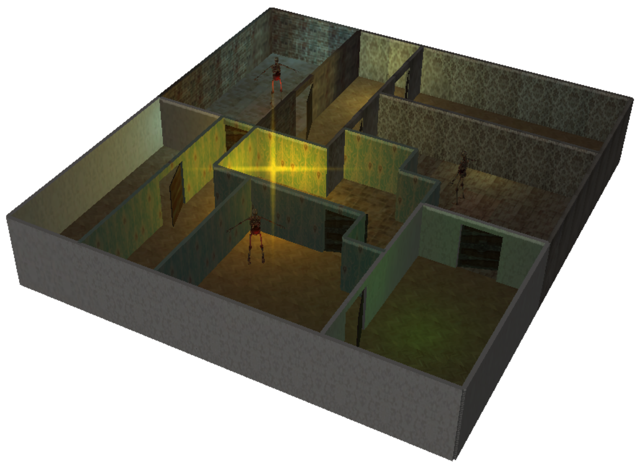


Figure 8: 3D render of Figure 1 in the *Unity3D* game engine.

real-time is memory intensive, a balance must be met and thus the fidelity of granular sound has to be much lower compared to studio recorded renders. A solution is the use of proprietary tools which produce higher quality and granular sounds, but are unusable for freely distributed games as they must run in real-time along the system. *MIDI* compositions are another option, with sonification algorithms requesting an external service to produce musical renders of MIDI files. Finally, background audio can be decoupled from foreground audio (i.e. sound reacting to player actions), as the latter does not necessarily require the quality of background audio and could be manipulated through systems such as *Pure Data* for more interesting soundscapes.

Another problem with the current sonification implementation is that it only accounts for the critical path. The survival horror genre often relies on players getting lost in order to enhance suspense, rather than finding the quickest way to the exit. Future work will adapt the current methodology for rooms outside of the critical path, by creating a list of alternative paths (i.e. rooms that branch off the critical path to a "dead-end room") and creating a suspense curve for each alternative path within the generated level, based on the monster distribution of these paths. This way the soundscape can stay consistent even in rooms outside the critical path.

Further work on the audio allocation algorithms will take into account different suspense models besides an inverted tension level. For instance, the sonification of rooms outside the critical path must be improved, as monsters can still be present in these sub-paths. Mixing algorithms will also be substantially polished: the combination of dissonance, scales, and sound effects in certain situations are current ideas being explored. Another promising approach is audio signal modification, for instance modifying audio signals to add reverberation based on room size, or using low-pass filters to simulate sounds coming from adjacent rooms which contain monsters. Volume mixing additions are also being explored to account for monsters in neighbouring rooms and adapt the volume until a monster is in a player's line of sight.

Sonancia is currently being imported to the *Unity3D* game engine (see Figure 8), which offers interesting out-of-the-box tools for enhancing the current system.

The suspense value parameters of audio assets can be reworked in order to provide a more accurate value mapping to the valence/arousal circumplex model (Russell 1980). To accomplish this, a classifier (such as an artificial neural network) can be trained via crowdsourcing, using human listeners to accurately classify "suspense". This will be incorporated in *Sonancia* so that sound designers can apply their own custom recordings and have them classified, similarly to the DARCI system (Norton, Heath, and Ventura 2010).

Finally, we intend to explore different types of interactions between the generated levels and soundscape generation so that consistency between the two is further improved. Once the system is fully imported into 3D, one idea is to dynamically arrange the lighting of the level so that it also follows the designer's intended tension curve (e.g. as the player progresses it gets darker). Another idea is to also add simple scripted events that can potentially occur during gameplay (e.g. a scream), which would have a higher probability of occurring in tenser situations.

Conclusion

This paper presented improvements to the *Sonancia* system, a multi-faceted level generator for the horror genre. The additions include a level generation system that optimizes towards a designer-defined tension curve, while still providing a degree of variability. The paper also presented some initial methodologies for creating soundscapes of generated levels by directly using the distribution of monsters in the level's path from the starting player position to the goal. Several experiments studied the impact of designer tension curves on level generation and sonification, as well as the efficiency of the GA in generating larger maps. Early results show that levels follow patterns consistent with the designer's tension curves, while still maintaining slight variations.

Acknowledgments

The research was supported, in part, by the FP7 ICT projects C2Learn (project no: 318480) and ILearnRW (project no: 318803), and by the FP7 Marie Curie CIG project AutoGameDesign (project no: 630665).

References

Cardamone, L.; Yannakakis, G. N.; Togelius, J.; and Lanzi, P. L. 2011. Evolving interesting maps for a first person shooter. In *Applications of Evolutionary Computation*. Springer. 63–72.

Cheong, Y.-G., and Young, R. M. 2008. Narrative generation for suspense: Modeling and evaluation. In *Interactive Storytelling*. Springer. 144–155.

Collins, K. 2013. *Playing with sound: a theory of interacting with sound and music in video games*. MIT Press.

Ekman, I., and Lankoski, P. 2009. Hair-raising entertainment: Emotions, sound, and structure in silent hill 2 and fatal frame. *Horror video games: Essays on the fusion of fear and play* 181–199.

Garner, T., and Grimshaw, M. 2014. Sonic virtuality: Understanding audio in a virtual world. *The Oxford Handbook of Virtuality* 364.

Gasselseder, H.-P. 2014. Re-scoring the games score: Dynamic music and immersion in the ludonarrative. *Proceedings of the Intelligent Human Computer Interaction conference* 1–8.

Hoover, A. K.; Cachia, W.; Liapis, A.; and Yannakakis, G. N. 2015. Audioinspace: Exploring the creative fusion of generative audio, visuals and gameplay. In *Proceedings of the EvoMusArt conference*. Springer. 101–112.

Liapis, A.; Yannakakis, G. N.; and Togelius, J. 2013. Generating map sketches for strategy games. In *Applications of Evolutionary Computation*. Springer Berlin Heidelberg. 264–273.

Liapis, A.; Yannakakis, G. N.; and Togelius, J. 2014. Computational game creativity. In *Proceedings of the International Conference of Computational Creativity*, volume 4, 71–78. Springer.

Lopes, P., and Yannakakis, G. N. 2014. Investigating collaborative creativity via machine-mediated game blending. In *Proceedings of the Artificial Intelligence and Interactive Digital Entertainment conference*.

Lopes, P.; Liapis, A.; and Yannakakis, G. N. 2015. Sonancia: Sonification of procedurally generated game levels. In *Proceedings of the 1st Computational Creativity and Games Workshop*.

Michalewicz, Z. 1995. A survey of constraint handling techniques in evolutionary computation methods. In *Proceedings of the 4th Annual Conference on Evolutionary Programming*. 135–155.

Norton, D.; Heath, D.; and Ventura, D. 2010. Establishing appreciation in a creative system. In *Proceedings of the International Conference of Computational Creativity*, 26–35.

Russell, J. A. 1980. A circumplex model of affect. *Journal of personality and social psychology* 39(6):1161.

Scirea, M.; Cheong, Y.-G.; Nelson, M. J.; and Bae, B.-C. 2014. Evaluating musical foreshadowing of videogame narrative experiences. In *Proceedings of the Audio Mostly: A Conference on Interaction With Sound*. ACM.

Shaker, N.; Nicolau, M.; Yannakakis, G. N.; Togelius, J.; and O'Neill, M. 2012. Evolving levels for super mario bros using grammatical evolution. In *Conference on Computational Intelligence and Games*, 304–311. IEEE.

Shaker, N.; Togelius, J.; and Nelson, M. J. 2015. *Procedural Content Generation in Games: A Textbook and an Overview of Current Research*. Springer.

Stevens, R., and Raybould, D. 2013. *The Game Audio Tutorial: A Practical Guide to Creating and Implementing Sound and Music for Interactive Games*. Taylor & Francis.

Togelius, J.; Yannakakis, G. N.; Stanley, K. O.; and Browne, C. 2011. Search-based procedural content generation: A taxonomy and survey. *Conference on Computational Intelligence and Games* 3(3):172–186.

Togelius, J.; De Nardi, R.; and Lucas, S. M. 2007. Towards automatic personalised content creation for racing games. In *IEEE Symposium on Computational Intelligence and Games.*, 252–259. IEEE.

Yannakakis, G. N., and Togelius, J. 2011. Experience-driven procedural content generation. *IEEE Transactions on Affective Computing* 2(3):147–161.