Case Study: A Small Step Toward Combining Procedural Content Approaches

Omitted for Anonymous Review

Omitted for Anonymous Review

Abstract. While generating content for video games is a growing field, many approaches focus on isolating single areas in games and inserting the new content into established game and game creation paradigms. This paper, however, introduces two new methods: one for creating audio and the other for weapon visuals, that combine to

1 Introduction

For most video games, the interplay between audio (e.g. character themes, sound effects, foreshadowing) and visuals (e.g. level design,) is crucial for the creative expression of the game designers and immersion of the players [6]. However, as *nonlinear games* (i.e. games with many different that give the player) surge in popularity (TODO stats), composers and designers are increasingly challenged to create meaningful TODOmoments/scenes for players who are now dynamically constructing their own game experiences.

As traditional composers and designers are adapting to this new medium [2], simultaneously approaches in procedurally generating assets for video games (called procedural content generation (PCG)) are emerging to enhance TODO-gameplay/replayability and alleviate some of the additional demands on designers, musicians, and artists. However, while these approaches in PCG are automating aspects game development, like asset creation, character behavior, and even developer tools, their focus ihas until recently been on inserting these assets and tools into the standard game development pipeline. (TODO- However, games based on PCG are interesting in their own right [**?**], emergence of playable PCG.) Instead, this paper not only explores two minimal PCG methods for generating music and weapon visuals and gameplay, but as a proof-of-concept also addresses how they can potentially influence each other to create a unique experience for each player.

(expressivity of the game, search in more than one facet).

(TODO-note incongruence between nonlinear games and the linear game we currently have - fix)

Add paragraph about multifaceted game creativity, see paper of liapis, distinguishes us from audiooverdrive, human for fitness, that it (limited human initivate) - explain different types

Mention domain of space shooter interactive evolution

As the field of procedurally generated content grows, so do the aims and scopes of what can be accomplished.

While procedurally generating content for video games may have begun as a way to relieve developers, the field has grown

The hope is that by exploring the interrelationships between two pcg systems within the same game, that a step can be made toward actualizing games with fully autonomous systems. Generating interesting soundscapes

Michael Cook (Liapis paper) - look specific to music, ICCC 2014 ICCC Traenor (Liapis paper) proof of concept

2 Background

While there are many approaches to procedurally generating content or facets for video games (e.g. visuals, audio, narrative, game design, level design and gameplay), the focus is often on the single facet rather than how they can combine to create a complete game experience. For instance, world maps in *Civilization V* (Firaxis 2010), dungeons in *Diablo* (Blizzard 1996), racing tracks [10], petal colors and shapes in Petalz [8], and weapons in *Borderlands* (Gearbox 2009) are all procedurally generated to provide the player with increased personalization and replayabity. Simiarly, *Audiosurf* [7] generates visual game play elements depending on the sound file provided by the player. While these generated elements may increase replayability and alleviate the art and design requirements on developers, these approaches serve as means to an end rather than celebrating the creativity of procedural content generation itself.

Other approaches aim to interweave procedurally generated elements to enhance the game experience. For instance, *Galactic Arms Race* [4] encourages players to interactively evolve weapon visuals and bullet trajectories to their aestetic and gameplay preferences, while *Game-o-matic* [11] merges human-authored narrative with procedurally generated rules and visuals to allow the player TODO. Similarly, *A Rogue Dream* [3] is about TODO and assigns game effects to word associations presented to the player in procedurally generated levels to creatively explore and interpret the designer's opinions and rhetoric XXX on topics such as religion or politics. While Proteus [**?**] is a completely procedurally generated pixel-art world where users can spacially and sonically explore their environments, the sonic output does not affect the previously generated landscapes. Even in these integrated environments, the output of one procedurally generated facet has little affect on the other.

While integrating procedurally generated game facets is a growing trend, few investigate bidirectional communication between elements. One notable exception is *AudioOverdrive*, which creates bidirectional communication between enemies, gameplay, and level visuals [5]. However, but the approach requires heavy human initiave by the developers.

The approach in this paper is to extend the ideas of Audiooverdrive to create an environment where the visuals and audio are closely coupled with minimal

interference from the developers. CURRENTPROJECTTITLE attempts to orchestrate the generation of audio, visuals and gameplay: in the testbed space shooter game, the appearance of the spaceship's weapons (e.g. color) and their gameplay effects (e.g. whether it hit an enemy) affect the game's soundtrack; the soundtrack in turn affects the firing rate, color and movement patterns of the weapon's particles, thus ensuring a bidirectional communication across creative facets. While similar to AudioOverdrive in terms of goals, theme and creative game facets tackled, CURRENTPROJECTTITLE specifically uses information on the spaceship's gameplay behavior and on-screen visuals rather than treating gameplay events such as firing a weapon solely on account of its sound effect.

3 Approach

While many approaches to generating content create singular game facets to augment the developers' creativity, the approach in this paper is to explore mixed initiative co-creativity, where the machine and player combine to create an augment the other's experience. The machine's creativity is enhanced through a *conceptual blending* of the audio and visual modules in figure 1. Here, audio inputs are sent to the visual module, which are looped back to inputs for the audio. Players exercise their own creativity when deciding which weapons and audio are sutible for their purpsoses. Because it is fundamental for the individual audio and visual domains to interact, each PCG module bases decisions on information gathered from both domains.

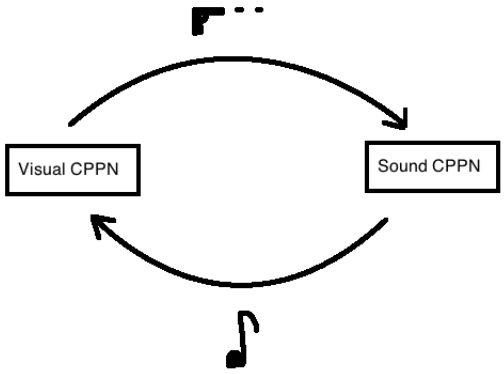


Fig. 1. In TODOmusicGame, audio and weapon visuals are represented by two separate CPPNs that each provide information to the other.

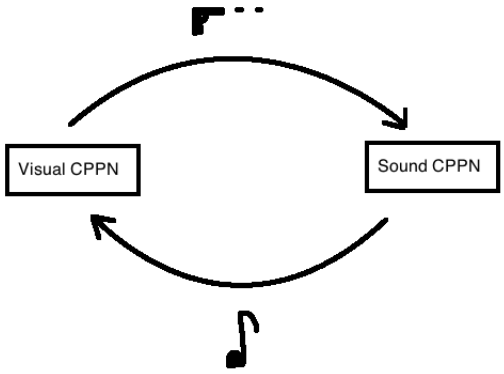


Fig. 2. TODO-genericimage-Compositional pattern producing networks (CPPNs; 9) are a special type of artificial neural network where activat. Like traditional ANNs, CPPNs are an interconnected network of nodes with bias outputs toward certain types of regularities. TODO:brief description of CPPNs

These relationships between the domains and generated outputs are represented by a special type of artificial neural network (ANN) called a compositional pattern producing network (CPPN; shown in figure 2) [9]. Like traditional ANNs, each CPPN is an interconnected network of nodes and connection weights that when provided input, calculate an output value. However, unlike traditional ANNs that only compute sigmoid functions at the hidden nodes, CPPNs can compute any function (e.g. Gaussian, sigmoid, sine, ramp etc.), in effect making the CPPN a patten generator biased toward certain regularities. Through this representation, the aim is that each By representing both the generated audio and visual/gameplay aspects through patterns of each, the aim is that TODO-somekindofcoolrelationship

Through the NeuroEvolution of Augmenting Topologies (NEAT) (NEAT; **?**) algorithm, each CPPN can start minimally and complexify as necessary over evolutionary time. By adding hidden nodes and connections, and changing activation functions and weight values, each new individual can expand the relationship between inputs and hidden nodes.

Each PCG module is represented by a separate CPPN that inputs domain information and outputs instructions for generating either audio or visual patterns that not only affect player immersion but also gameplay. Because each CPPN - Domain patterns are sent to the CP- Explain how they interact, both take each other as input: Game loop

To personalize the relationship between music, visuals, and gameplay, these CPPNs can be evolved through a process similar to animal breeding called inter-

active evolutionary computation (IEC), wherein the human use rather than an explicit fitness function rates candidate individuals. CPPNs are evolved through the NeuroEvolution of Augmenting Topologies (NEAT; **?**) algorithm which was originally developed to solve control and decision tasks but generates sound and visuals in this paper.

Because CPPNs produce patterns of the inputs, it is important to choose meaninful inputs Because patterns are important, generating desirable melodic and visual patterns is dependent on the types of inputs sent through the CPPN.

Relationships in TODO music-game are represented as compositional pattern producing networks evolved with neat. Each game asset has a different set of inputs and outputs.

3.1 Game

This mixed initiave design is presented through a TODO-typeofgame(shoot-them-up) space shooter, called TODO-name [1], which extends **?**] approach for weapon particle generation by adding musical inputs to the CPPN. Shown in figure 3, the player moves through levels in an outerspace environment attempting to avoid obstacles and shoot enemies while simultaneously evolving weapon trajectories and visuals [**?**] with the generated accompanying audio. TODO-motivationsentence-loop construct between the two networks where one network tells the in-game bullets how to travel whereas the other tells the game what sound to produce.

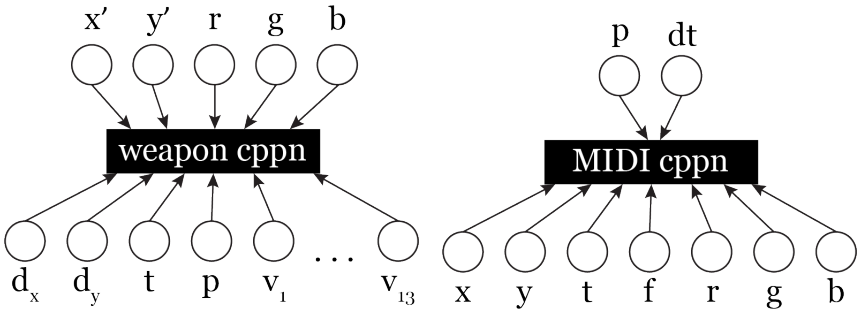
Upon launch, the information for the first bullet is gathered when a random note from the C Major pentatonic scale is struck and sent to the visual CPPN in figure **??**a. Together with the pitch information from the initially random MIDI note, this CPPN also inputs the $(\delta x, \delta y)$ position between where the bullet was fired and where it is currently, and the time t since firing. Note that before the bullet is fired, the initial $(\delta x, \delta y) = (0, 0)$ and $t = 0$. The ouputs determine the red, green, and blue (RGB) color values for the bullet, and the (x', y') its new position.

New notes are fired when the bullet hits an enemy or otherwise at the end of the current note's duration. Shown in figure **??**b, the music CPPN determines pitch and a notes duration through the respective Pitch and DeltaTime outputs. These values are based on the (x, y) position of where the last bullet was fired, time since firing t, whether the bullet struck an enemey h, and its RGB values.

As the game progresseses, different levels are displayed. Players encounter hostile enemies who have their own weapon trajectories, large rocks to avoid, and small rocks to avoid and shoot. The longer each weapon and audio network are in play, the higher the prefer Through IEC, players evolve the visual and audio networks toward their aestetic preferences.

The networks both start off as Single-Layer Perceptrons, allowing for an observation from a simple network to a more complex one as more Hidden Nodes are added to the network. The fitness function determined for each network is different between the bullet network and the MIDI network. The bullet CPPN is evaluated based on how many times a shot was fired using that network.

Fig. 3. In TODO-gameName, the user controls the space ship on the left side of the screen while moving through each space level. The health of the ship is shown at the top of the screen as "hull strength" while the weapon score is displayed at the bottom.



(a) Weapon CPPN (b) Audio CPPN

Fig. 4. fig:cppnspecific

The MIDI CPPN is evaluated based on how long the player remained with that network.

4 Experiments

Experiments in this section illustrate the potential power of this proof-of-concept through exploring the affect different inputs have on generated output. The first experiment explores the relationship between the sound CPPN and the visual by systematically restricting inputs to both and exploring the affects on the other. First, the MIDI input to the visual network is restricted to two pitches, middle C of MIDI value 60 and high C of MIDI value 72 to explore how these notes can effect the visual trajectories and colors. Then, In this way, it is clear how much, influence the MIDI CPPN exerts on the visuals.

The second experiment builds on previous work by Cachia et al. [1] explores how starting with a user provided random note from the C major pentatonic scale affects the visuals and vice versa.

In each of the following experiments, tournament selection occurs amongst a population of fifty individuals, ten of which are rated by the player through IEC. Players can switch the current weapon or audio reprentation by pressing a button to activate a different CPPN chromosome whose effects are immediately heard and seen in TODO-game. The evolutionary parameters are set through preliminary testing, and the probability of adding a new node or connection is 30% and 15% respectively. Similarly, the activation functions at any node can be replaced by one of the following with an even chance: Hidden Node Functions: Sigmoid, Hyperbolic Tangent, Sine, Cosine, Bipolar Sigmoid, Gaussian, Ramp, Step, Spike [**?**]. However, the activation function of any particular node is only changed with a 20% chance. Weights mutated with a 90% chance real values between $[-10, 10]$. Ocassionally (1% chance), connections are disabled.

number of generations

likes for each gun

5 Results

time results

initial feed for music, constant on either side, see which generate what

Because the CPPNs can potentially represent any relationship, sometimes an inverse relationship Note trajectory outputs can go backwards, may be better for different types of levels

pitch helps determine speed

6 Discussion and Future Work

While this paper is a proof-of-concept about fusing game and player creativity, it is important that players enjoy this new mixed-initiave game. Future work aims

to explore how much users enjoy this fused concept, and whether they prefer the described interaction between the two networks.

the resulting videos illustrate that relationships between the audio and visuals are TODO, which implies that this approach could be more enjoyable for players. Future work includes user testing which should indicate to what extent the game interactions are appreciated by the players.

Evaluation: studies with designers, humans, player study, listener's study, control - potential of the fusion on creativity/human, GAR fed with number of weapons, see which weapon music combination that they use, what they like/preference comes directly from the use, what they like about the weapons why do they like the combo

Furthermore,

Monitors on the music to trigger events in game - appearance of enemies, frequencies, boss battles

Narrative?Level design

Game about learning weapon patters, evolving weapon patterns

7 Conclusion

This paper presented a mixed initiative co-creative goal for developers and players. Through TODOgame

However,

Extensibility to other games, other disciplines, out of the box

8 Acknowledgements

[Omitted for anonymous review]

References

[1] William Cachia, Luke Aquilina, Hector P. Martinez, and Georgios N. Yannakakis. Procedural generation of music-guided weapons. In *Proceedings of the IEEE Conference on Computational Intelligence and Games (CIG)*, page TODO.

[2] Karen Collins. An introduction to procedural music in video games. *Contemporary Music Review*, 28(1):5–15, 2009.

[3] Michael Cook and Simon Colton. A rogue dream: Automatically generating meaningful content for games. In *Proceedings of the AIIDE Workshop on Experimental AI and Games*, 2014.

[4] Erin J. Hastings, Ratan K. Guha, and Kenneth O. Stanley. Automatic content generation in the galactic arms race video game. *IEEE Transactions on Computational Intelligence and AI in Games*, 1(4):245–263, 2009.

[5] Nils Iver Holtar, Mark J. Nelson, and Julian Togelius. Audioverdrive: Exploring bidirectional communication between music and gameplay. In *Proceedings of the 2013 International Computer Music Conference*, 2013.

[6] Antonios Liapis, Georgios N. Yannakakis, and Julian Togelius. Computational game creativity. In *Proceedings of the Fifth International Conference on Computational Creativity*, 2014.

[7] Audio Surf LLC. Audiosurf. www.audio-surf.com, 2011.

[8] Sebastian Risi, Joel Lehman, David D'Ambrosio, Ryan Hall, and Kenneth O. Stanley. Combining search-based procedural content generation and social gaming in the petalz video game. In *Proceedings of Artificial Intelligence and Interactive Digital Entertainment Conference*, 2012.

[9] Kenneth O. Stanley. Compositional pattern producing networks: A novel abstraction of development. *Genetic Programming and Evolvable Machines Special Issue on Developmental Systems*, 8(2):131–162, 2007.

[10] Julian Togelius, R De Nardi, and Simon M. Lucas. Towards automatic personalised content creation for racing games. In *Proceedings of IEEE Symposium on Computational Intelligence and Games*, pages 252–259. IEEE, 2007.

[11] Mike Treanor, Bryan Blackford, Michael Mateas, and Ian Bogost. Game-o-matic: Generating videogames that represent ideas. In *Proceedings of the FDG Workshop on Procedural Content Generation*, 2012.