This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2016.2520305, IEEE Transactions on Computational Intelligence and AI in Games

The ANGELINA Videogame Design System, Part II

Michael Cook, Simon Colton, and Jeremy Gow
ccg.doc.gold.ac.uk

***Abstract*—Procedural content generation is generally viewed as a means to an end – a tool employed by designers to overcome technical problems or achieve a particular design goal. When we move from generating single parts of games to automating the entirety of their design, however, we find ourselves facing a far wider and more interesting set of problems than mere generation. When the designer of a game is a piece of software, we face questions about what it means to be a designer, about Computational Creativity, and about how to assess the growth of these automated game designers and the value of their output. Answering these questions can lead to new ideas in how to generate content procedurally, and produce systems that can further the cutting edge of game design.**

This paper describes work done to take an automated game designer and advance it towards being a member of a creative community. We outline extensions made to the system to give it more autonomy and creative independence, in order to strengthen claims that the software is acting creatively. We describe and reflect upon the software's participation in the games community, including entering two game development contests, and show the opportunities and difficulties of such engagement. We consider methods for evaluating automated game designers as creative entities, and underline the need for automated game design to be a major frontier in future games research.

***Index Terms*—procedural content generation, automated game design, computational creativity**

I. Introduction

PROCEDURAL content generation is viewed by its practitioners through many different metaphors [1], but perhaps the most pervasive is that of a *tool* – a means to an end, or a solution to a problem. Procedural content generation (PCG) is employed by a human designer to complete a task, and as such is an extension of the designer. Even in [1] when the authors explore the metaphor of a PCG system as a designer, they define it in terms of a human superior:

> *[human] designers will have to decide how much design responsibility to delegate to [PCG systems]*

Automated Game Design (AGD) is a distinct offshoot of PCG research. It concerns itself with the construction of systems which take *primary* responsibility for the design of a game, i.e. systems which act as autonomous creative entities. Though such systems may not be responsible for the creation of every piece of content in the game, they are seen as the driving force behind the game's development. This is partly a technical challenge, which is why much of the work related to AGD has come from procedural content generation research. To this end, in [2] we outlined an evolutionary PCG framework for a system called ANGELINA, which designs different elements of a game simultaneously in order to produce a cohesive final game design.

However, Automated Game Design is also a creative challenge as well as a technical one. In this paper, we will describe extensions to ANGELINA that took the system from a designer of simple games to a system which could engage with a creative community and that is beginning to take steps towards being accepted as a autonomously creative. We will discuss models for assessing progress in creative software, introduce versions of ANGELINA that implement features specifically targeting computational creativity topics, and give accounts of evaluations both formal and informal, including a report on what is to our knowledge the first time a piece of software has entered a game jam, a landmark moment for automated game design and computational creativity.

The remainder of this paper is organised as follows: in section II we describe some existing approaches to categorising procedural content generators, as well as some background on the evaluation of creative software from the perspective of Computational Creativity. In section III we give an overview of ANGELINA, an automated game designer which forms the basis of our research into the area. We describe ANGELINA$_3$, which represented a step forward in the software's creativity. In section V we give details of ANGELINA's participation in the game design community, and how that community reacted to ANGELINA's recent participation in a game jam. In section VI we return to the topics covered in the background section on evaluation and apply them to ANGELINA, evaluating the software in the context of Computational Creativity, and proposing extensions to the classification of procedural content generators to better capture the properties that automated game designers like ANGELINA have. In section VII we discuss opportunities for future work on the system, and similar AGD projects. Finally, in section VIII we summarise the work undertaken and offer some conclusions.

II. Background: Analysing Generative Software

This section summarises some existing models, classifications and processes for analysing generative software, both in the general sense of software acting creatively, and more specifically for software which procedurally generates content for videogames. This section will help support the evaluation of our work later in the paper, as well as providing the basis for a discussion of the evaluation of autonomous game designers.

A. Procedural Content Generation Taxonomy

In [3] and [4] Togelius et al. propose several classifications for search-based procedural content generation systems. Most of these are not binary classifiers but instead define a spectrum along which PCG systems can be placed.

1) Online – Offline: A PCG system may operate at runtime within the game (such as the level generation in *Spelunky* [5]) or may be used prior to the game being released to generate

1943-068X (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2016.2520305, IEEE Transactions on Computational Intelligence and AI in Games

static content the game will subsequently use (such as the generation of a galaxy map in *EVE: Online* [6]).

2) Necessary – Optional: The content produced by a PCG system may be core to the gameplay and part of the critical path[1] through the game world. For example, the levels in Spelunky are procedurally generated, and it is not possible to play the game without experiencing the content produced through this system. On the other hand, optional quests in *Skyrim* [7] have a procedurally-generated component, but the content may not be experienced by some players depending on their chosen path through the game.

3) Random Seeds – Parameter Vectors: Informally, this feature of the taxonomy describes how controlled the input to the PCG system is. Some PCG systems derive their input from purely random streams, such as world generators which use Perlin noise [8]. The primary way of interacting with or varying these PCG systems is through providing a random seed to the generator that drives the system. On the other hand, some PCG systems use parameterised inputs, allowing them to offer some control to users. Map generation in the *Civilization* series [9] has many parameters that can be alterered prior to generation that affect geographic factors which influence the game world's development.

4) Stochastic – Deterministic: This feature is seen as a companion to the previous one. It distinguishes between PCG systems which produce similar or identical output on repeated runs (with the same parameters provided, in the case of a PCG system based on parameter vectors) compared with systems which always produce different outputs even when the parameters define the same potential design space. Note that this distinction explicitly does not consider the random generator's seed to be part of the calculation here, so generators such as *Minecraft*'s [10] which produce different worlds unless the same seed is provided, are still considered stochastic generators.

5) Constructive – Generate-and-Test: Some PCG systems will produce their content in discrete steps, additively contributing content to the result until it is finished. Other systems will generate a piece of content, and then evaluate it for quality before discarding it, using it to produce further content, or returning it in its entirety. The difference between these two approaches is considerable, as generate-and-test approaches are often not robust and cannot guarantee a piece of content will be generated within a time constraint. *Dwarf Fortress* [11] uses generate-and-test in its world generation phase, despite being an example of online PCG, because its playerbase is willing to wait for a considerable time for a world to be generated. In contrast, Spelunky's level generation adds tiles in stages, ensuring playability with each step, and then decorates the tiles with additional content. Here, nothing is removed and failure is not possible.

6) Generic – Adaptive: Some PCG systems adapt their content to a specific player or group of players, using modelling or heuristics, such as Left 4 Dead's adaptive Director [12]. The more standard approach is to build static content generators which create generic content in the same way for all players, such as Spelunky [5].

An alternative taxonomy is offered in [13] by Smith, who proposes a breakdown of PCG which shares some of the classifications listed above, viewed through the lens of game design with subtle differences in emphasis in some parts. Smith also makes a distinction between, for example, the use of *patterns*, large *experiential chunks*, or *subgenerators* in describing the granularity of content generated, and explicitly mentions the use of PCG as a game mechanic or driver of game design. For our purposes, working with software which uses PCG as a means to produce games which have no PCG within them, Togelius' taxonomy is better suited in our opinion. Both taxonomies are relevant to PCG discourse however and Smith's emphasis on design is very important for the field as it stands today.

B. The FACE Model

The FACE model, proposed in [14] and [15], describes a multi-part model that can be used to assess a system in terms of the creative acts it performs, whether they are performed by a person or by the system itself, and how much assistance the system is provided in performing those tasks. Its most important application is in evaluating multiple systems and showing the differences between them. This is most often used to show progress between an earlier version of a system and its latest incarnation.

FACE is an acronym defining the four parts of the model, namely Framing, Aesthetics, Concepts and Examples. Creative systems are categorised according to which actions they perform in the process of creation, which can be expressed as a tuple of letters representing those acts. Each action in the FACE acronym can be performed in one of two ways, expressed with a superscript p or g next to the letter. The superscripts denote what kind of output the generative act produces: g acts produce artefacts, while p acts produce processes (which in turn can be utilised in g acts to generate artefacts). This results in a total of eight generative actions that can be included within a FACE tuple, which produce the following outputs:

- E^g: an expression of a concept.
- E^p: a method for generating expressions of a concept.
- C^g: a concept.
- C^p: a method for generating concepts.
- A^g: an aesthetic measure.
- A^p: a method for generating aesthetic measures.
- F^g: an item of framing information.
- F^p: a method for generating framing information.

For example, a system might be described by the tuple $< F^g, C^g >$, showing that it performs two generative acts relating to framing and concepts.

The introduction of the FACE model in [14] gives descriptions of each of the four components of the model. Concepts are executable programs, or something which can be interpreted as such, which take input and produce output. An expression of a concept is a pair of an input and an output produced by a concept. A simple example from mathematics

[1]In game design terminology, the critical path refers to the minimum path from the start to the finish of the game, i.e. content which is necessarily experienced by the player in completing the game.

1943-068X (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2016.2520305, IEEE Transactions on Computational Intelligence and AI in Games

given in [15] is that of prime numbers, which are a concept that can take a number as input and evaluate whether that number has exactly two divisors. An expression of this concept is a number (the input) and the evaluation of whether or not the number is prime (the output from the concept of prime numbers).

An aesthetic measure is a function which takes a $(concept, expression)$ pair (one of which may be null) and returns a positive real number. To continue the prime number example, we might develop an aesthetic measure which passes judgement on integer sequences according to how dense examples are on the number line. This aesthetic measure could be applied to prime numbers as a concept. Finally, framing information is used to refer to information supplied alongside a creative work, which puts the work into 'cultural or historical context', describes 'the processes underlying the generative acts' or gives 'calculations about concepts... with respect to aesthetic measures' [14]. Framing information can demonstrate that a system has an awareness of the decisions it has made, and the place its work has in a wider context. This can be important when managing the perception of a system having acted creatively.

The FACE model has been successfully applied to the analysis of poetry [16] and melody generation [17], and is still being actively developed [18] [19].

III. ANGELINA - CCE for Game Design

A. Core Structure

ANGELINA[2] is a cooperative coevolutionary (CCE) system which produces complete, playable games. In [2] we describe in more detail ANGELINA and the framework we build different versions around. We briefly give a summary here before proceeding to outline two later versions of the software. Different versions of ANGELINA are denoted by subscripts, such as ANGELINA_1, ordered chronologically.

CCE is a variant on standard computational evolution that uses *species*, smaller evolutionary systems which individually evolve part of a much larger object. Each species has its own population, methods for crossover and metrics for fitness evaluation. However, in order to carry out fitness assessments, it must combine individuals from its population with individuals from every other species in the CCE system. By combining them into the larger object being evolved (in this case, a game) it can then evaluate an individual in context, assessing how it *co-operates* with other species. For further details on how ANGELINA uses CCE, see [2].

The development of all versions of ANGELINA involved a very similar methodology. First, suitable libraries are selected to implement the system's output game in, and therefore affect the design space the system explores. A key objective for the project from the beginning was to produce games which were standalone applications, easily distributed, and easily playable ideally via the web – aside from the desire to share our results with the wider community, automated game design relies on evaluation through play, and choosing technologies that facilitate this is therefore crucial. ANGELINA has implemented games on many platforms, including HTML5, Flash, Unity and Android.

After choosing base libraries that will support the core system, abstractions must be chosen for the concepts the system will be dealing with, such as level designs or player objectives. The level of detail and structure of an abstraction for a given game feature affects the size of the generative space, impacting both the difficulty of the generation process and the potential for novelty. The choice of abstraction in commercial procedural content generators is a defining feature of the generator, and many abstractions have become very common in normal game development – such as representing levels as arrays of integers (adopted by the middleware level designer *Tiled* [21]) or grammatical systems for representing rules.

Once abstractions are well-defined, the core CCE system can be implemented, with appropriate generators and evaluation functions. The CCE system may or may not appeal to the chosen game libraries defined in the first step above in order to evaluate its fitness functions. Alternatively, the system may rely on predefined abstractions of the game in order to evaluate interactions. Some versions of ANGELINA used abstracted simulations of gameplay in order to evaluate interactions between certain species. Others have used direct execution of game code in order to assess fitness.

The core CCE system ultimately defers to some kind of finishing module that compiles the internal representation of a game into a finished executable or source bundle. More complex approaches to automated game generation may, in theory, not require this step if they directly modify code as part of their design process, since the final evolved system is already publishable. However, most systems – particularly CCE systems, which rely on abstractions for the evaluation and manipulation of genotypes – will have this finishing step as a final way to translate from abstractions to a playable result. Versions of ANGELINA to date have primarily used inline code replacement, where code is inserted into pre-existing template programs to implement key functionality, as the means of producing runnable output. This varies slightly according to platform, as described in the following sections.

B. ANGELINA 3: Creativity in Game Design

In [2] we describe ANGELINA_2, a version of ANGELINA which produced simple Metroidvania-style games in which the player explores a two-dimensional level using a series of powerups. Metroidvania games distinguish themselves as a subgenre through a notion of gated exploration – their game worlds are typically very large, but the player can only explore small parts of them, and must find new items and abilities that extend their ability to explore the world.

ANGELINA_3 was developed on the foundations laid by ANGELINA_2, and shares much of its core codebase. Retaining the domain of Metroidvania games, ANGELINA_3 was developed to investigate questions of creativity in autonomous game design. To do this, we designed ANGELINA_3 to produce games themed around current events, incorporating textual, visual and aural media to theme a game according to a

[2]A Novel Game-Evolving Lab-rat I've Named ANGELINA.

1943-068X (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2016.2520305, IEEE Transactions on Computational Intelligence and AI in Games

contemporary news story. Previous research in Computational Creativity has explored the idea of systems using dynamic web data as source material for creative works, such as in [22] or [23], as a way of increasing the perception of creativity in a system by making its output less dependent on the software's creator, more in-touch with current culture, and less predictable. We describe the approach we took in ANGELINA$_3$ to allow it to work with and be influenced by web data in this section. We also describe the extensions ANGELINA$_3$ makes to the ANGELINA$_2$ system, detailing how the core architecture is extended in terms of platform, abstractions and the CCE system itself.

C. Preliminary Design

ANGELINA$_3$ is the first version of ANGELINA to include an automated 'predesign' phase that occurs prior to the CCE system's execution. The purpose of this phase is to gather data from the web, including assets for use in the game, and to make this available to the CCE system where it is needed (as well as being compiled into the final version of the game). The predesign phase begins by selecting a newspaper article from the website of British newspaper The Guardian. It does this by downloading the current headlines at the time of execution, and ranking them according to priority based on which of several criteria they fulfil, which we defined through experimental testing. These criteria are:

1) Story novelty: ANGELINA$_3$ records the headlines of any story it has read before, and ranks new stories higher than old ones. This is to avoid repetition in the output of the system.

2) Tag novelty: Each news story has tags attached which define the story's topics or theme. ANGELINA$_3$ keeps a record of seen tags. If it encounters a new tag, it ranks the related story more highly.

3) Person novelty: ANGELINA$_3$ also records any people it detects in news stories it has read previously. We describe how it does this below. If it detects that a news story includes someone it has not encountered before, it ranks the story more highly. This is to encourage attention on emerging stories or new characters in the narrative of the news.

4) Opinion shift: ANGELINA$_3$ uses Twitter to gauge public opinion on people it detects in news articles. We describe this process below. If it detects a large shift in public opinion about a person featured in a news story, it ranks a story more highly. This gives ANGELINA$_3$ multiple points of influence (both the news story, and social media) and helps it react to important or controversial events.

The ordering above is from least important to most. That is, ANGELINA$_3$ will prioritise stories which include a shift in opinion about someone it has a record of. If there are none of this type, it will look for new people, then new tags, and then simply stories it hasn't read. If it finds no such stories, it will randomly select a headline. Ties are broken randomly, so that if two stories contain opinion shifts about a person, ANGELINA chooses one of them with equal probability. We mentioned two key abilities in the above list: person detection, and public opinion mining. ANGELINA$_3$ assesses if a named person in a news article is prominent or not by searching Wikipedia and checking if a page exists about a person with the same name who is currently alive. We found this approach to be effective in determining whether a name referred to someone currently in the public eye or relevant to a news story. When it detects a person it hasn't seen before, it makes an entry in a database file, along with a new value for the current public opinion of the person. ANGELINA$_3$ can similarly use Wikipedia to identify countries, using a list of sovereign states.

Public opinion is assessed by querying the social media site Twitter for completions of the phrase "$< name >$ is ...", a technique proposed in [24] which we term *web cold-reading*. The words following the phrase are looked up in the AFINN sentiment word list [25], which ranks a list of common words with a sentiment rating in $\{-5, ..., 5\}$, where -5 expresses an extremely negative sentiment, and 5 an extremely positive one. ANGELINA$_3$ averages out the sentiment rating of the tweets returning from its search query, and adds this to a running average of opinion that is updated each time the person is encountered in the news.

Once a story has been selected as the topic for a game, the headline, subheader, body text and tags are downloaded to be used as starting points for the next round of refinement. ANGELINA$_3$ then looks for visual and aural media it can use inside its game design. For each tag, person and country identified in the article, ANGELINA$_3$ can perform searches to extract images from both Google Images and Flickr. The latter is used specifically for images related to countries, which act as backdrops to the games. For people, the searches are augmented with emotionally-loaded keywords as described in [26] according to the public perception of the person recorded in ANGELINA$_3$'s database. **??**.

Other searches use the article tags unchanged as search terms. The results, along with any pictures of people, form the *image set* that is part of the output of the predesign phase. These images are later selected at random to be placed in the final game using the new species in ANGELINA$_3$'s CCE system, described below. ANGELINA$_3$ also creates a *sound set* by searching sound effect and recording libraries using tags from the article on the website FreeSound[3]. This results in a wide variety of recordings, from spoken word to singing, from ambient environmental noise to staged sound effects. Specific selections are made by sorting according to different metrics provided by the site's search engine, such as the number of times a clip has been downloaded or how long the clip is. The metric used for a particular execution is selected randomly by ANGELINA$_3$ from the site's list.

To complete the audio set, ANGELINA$_3$ downloads a piece of music from the website of Kevin Macleod[4]. Macleod organises his music according to many criteria, including mood. ANGELINA$_3$ performs a sentiment analysis on the body text of the sourced Guardian article, again using the AFINN database of word sentiments to assess individual words used, and to gain an average sentiment for the article. Using this analysis, the system can select an appropriate piece of

[3]http://www.freesound.org

[4]http://www.incompetech.org

1943-068X (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2016.2520305, IEEE Transactions on Computational Intelligence and AI in Games

I was reading the Guardian website today when I came across a story titled 'Obama to urge Afghan president Karzai to push for Taliban settlement'. It interested me because I'd read the other articles that day, and I prefer reading new things for inspiration. I looked for images of United States landscape for the background because it was mentioned in the article. I also wanted to include some of the important people from the article. For example, I looked for photographs of Barack Obama. I searched for happy photos of the person because I like them. I also focused on Afghanistan because it was mentioned in the article a lot.

Fig. 1. An excerpt from the commentary for the game *Hot NATO*.

music for the game.

Finally, ANGELINA$_3$ generates a title for the game. This is done using two sources of information: first, several corpora of pop culture references were assembled for ANGELINA$_3$ to search through: the Internet Movie Database Top 250 Films[5], the Guardian Newspaper's 1000 Best Albums Ever[6], several Top 100 games lists from major websites[7], and a list of proverbs and sayings[8]. This was combined with code written to access the online rhyming dictionaries RhymeZone[9] and WikiRhymer[10]. Tags, countries and the surnames of people detected in news articles are selected randomly and fed through the rhyming dictionaries. The resulting rhymes, if any, are then matched against results in the pop culture corpus. If any results are found, the rhyming word in the result is swapped out with the tag, surname or country originally used in the search. This creates a pun-like effect where a pop culture reference is related some way to the article. Examples of titles are given later in this section. This concludes the predesign phase, and the core CCE process now takes place.

D. Commentary Generation

In section II-B we described the FACE model for evaluating software in a Computational Creativity framework. Part of this model is concerned with *framing* information – details that place a creative work in context, or explain the creative process behind it. Following on from work on computational poetry generation in [16], ANGELINA$_3$ records key decisions made during the creation of each game, and creates a 'commentary' which frames part of the creative process. A template is used as the basis for each commentary, with key segments replaced with prepared text depending on the types of decision made, and key data from the news story (such as the names of relevant people). An excerpt from a sample commentary is shown in figure 1. The accompanying game is described in more detail below in the subsection '*Sample Games*'.

[5]http://www.imdb.com/chart/top
[6]http://music.guardian.co.uk/1000albums
[7]Such as http://www.gamesradar.com/best-games-ever/
[8]http://www.phrases.org.uk/meanings/proverbs.html
[9]http://www.rhymezone.com/
[10]http://wikirhymer.com/

E. Platform and Compilation

Like ANGELINA$_2$, ANGELINA$_3$ is built in Java as a CCE system, and when finished it modifies template ActionScript files to build into a Flash executable game. This game uses the Flixel game library.

F. Abstractions

ANGELINA$_3$ introduces a fourth species to the Metroidvania generation seen in ANGELINA$_2$, referred to as Artistic Direction or AD_3. An Artistic Direction is a set of placement co-ordinates and a mapping from each image in the image set and sound effect in the sound set to one of the placement co-ordinates. Placements for images also include a width and height which the image is scaled to. The placement specification for a sound effect also specifies a triggering distance in pixels. The first time the player character is within that distance of the sound effect's starting co-ordinate, the sound starts playing.

G. Internal CCE System

The initial generation of AD_3's population takes place randomly, with co-ordinate placement selected entirely randomly, and specialised parameters like sound effect range or image width randomly initialised within preset boundaries. Internal evaluation ensures that image and sound placements do not overlap with other placements of the same type, although image placements may overlap sound placements and vice versa, because the two do not interfere with one another. External evaluation compares placements with the level geometry and accessibility. Image placements are evolved to maximise the size of each image, while ensuring that no part of any of the images were overlapped by tiles in the level. Additionally, since both the images and the sound effects are triggered by the player's presence, the external evaluation assigns lower fitness to placements that put any kind of content outside the reachable area of the player. This uses the same reachability analysis already being calculated for ANGELINA$_2$'s level evaluation (see [2]), so the additional computational load is minimal. Figure 2 shows a screenshot from a game produced by ANGELINA$_3$, displaying a photograph of a politician in a prominent location in the game. We describe this game's design in full, along with other examples from the system, below.

Crossover of AD_3's population uses one-point crossover, at the granularity of individual placements, considering both sound and image assignments as a single array of placements. Mutation alters individual co-ordinates as well as special data such as the range of activation for sound effects, randomly varying them within sensible bounds which we set after some preliminary experimentation.

H. Sample Games

Below are descriptions of two games produced by ANGELINA$_3$, showing the kinds of output achievable with the system currently. We give details of the preliminary design phase and the theme selected for the game, as well as the mechanical design of the game itself.

1943-068X (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2016.2520305, IEEE Transactions on Computational Intelligence and AI in Games

Fig. 2. Screenshot from *The Conservation of Emily*, a newsgame about deforestation and political scandal. Playable online at www.gamesbyangelina.org/games

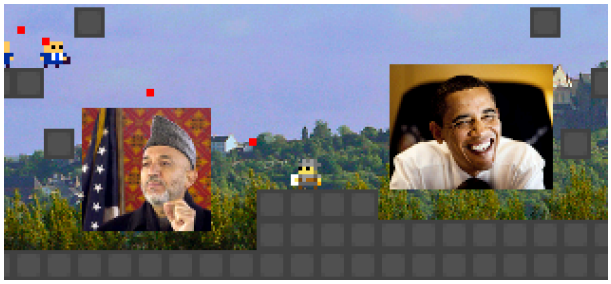

Fig. 3. Screenshot from *Hot NATO*, a newsgame about the war in Afghanistan. Playable online at www.gamesbyangelina.org/games

1) The Conservation of Emily: Created on May 10th 2012, in response to an article titled *Lord Mandelson confirms he is advising company accused of illegal logging* on the front page of The Guardian website. The title is a play on the title of the 1964 film *The Americanization of Emily*, which ANGELINA selected from a corpus of famous pop culture artefacts including music and film, and has combined with the word *Conservation*, which is one of the themes of the news article. The game features imagery of small animals, photographs of Lord Mandelson looking bad-tempered, as well as sound effects of animals in the rainforest, and a man screaming in pain. The sound and images were downloaded from the Internet based on search terms including *Activism*, *Endangered Species* and *Conservation*. Figure 2 shows a screenshot from the game, while Figure 4 shows the level layout.

The only specific parameters passed to ANGELINA$_3$ for this game's creation were the dimensions of the map in chunks – 4 chunks tall by 3 wide. The height of the level combined with the natural lack of jump power in the starting player, leads ANGELINA$_3$ to introduce a jump powerup early on in the level. This allows the player to reach the top of the level, where they can find a key which unlocks the extreme left corner of the map in which the exit is placed. Some imagery downloaded for use in the game is placed in an area which is technically accessible but unlikely to be seen by the player on their initial playthrough, as it requires them to backtrack and explore the level. Many players enjoyed this feature of the game, although it is a serendipitous result of the system's assessment of reachable areas. When calculating areas which become reachable upon collection of a powerup (like the key picked up at the end of the game), ANGELINA$_3$ does not take into account whether the newly reachable areas are on the critical path to the exit. In many cases they are areas which the player is yet to pass through on their route to the exit, but in some cases – like this – the player may interpret them as 'secret' areas, which they would have to explore to find.

2) Hot NATO: Created on May 20th 2012, in response to an article titled *Obama to urge Afghan president Karzai to push for Taliban settlement* on the website of The Guardian newspaper. The title is a play on the colloquialism 'hot potato', with NATO (a military group involved in the Afghanistan war) substituted in. Unlike The Conservation of Emily, this play on words is doubly appropriate, since the meaning of the original phrase (a controversial topic) fits the topic of the game. This higher level of creative meaning is not intentional in ANGELINA$_3$, but something we aspire to with future versions of the software. The game features powerful military symbolism, including photographs of the Afghan mountains with soldiers walking across them, pictures of chinook helicopters, and sound effects including the sound of large machine guns firing. Photographs of the leaders of both the US and Afghan governments also appear – the Afghan leader photos include accusatory and angry photographs, while the US leader is depicted as smiling and friendly, influenced by ANGELINA$_3$'s

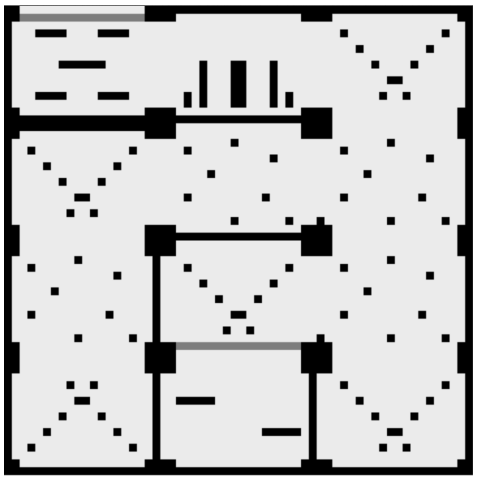

Fig. 4. The map from *The Conservation of Emily*. The player begins in the bottom right, and must reach the exit in the top left. Grey bars are locked until the player retrieves a key. In this level, the key is located in the far top left of the map near the exit.

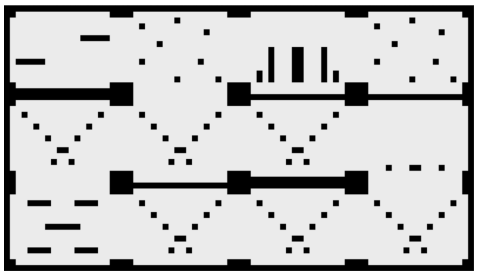

Fig. 5. The map from *Hot NATO*. The player begins in the bottom right, and must reach the exit in the top left.

1943-068X (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2016.2520305, IEEE Transactions on Computational Intelligence and AI in Games

analysis of these leaders on Twitter and the Internet. Figure 3 shows a screenshot from the game, while Figure 5 shows the level layout.

The game is created on a 4 chunk wide, 3 chunk high map. The player crosses right to left, collecting jump powerups which let them ascend to the next level, where they progress right, before ascending and moving left again. The fitness functions are such that they do not look for perfect usage of the map space, to avoid the system designing maps where the player always passes through every part of the game world, as this produces a sense of repetition in the map design. In this case, the game also includes drops whereby failed jumps bring the player back down to a lower level again. Simulation of failure in the game simulation, such as the player mis-timing a jump or getting stuck in a part of the game, is not present in ANGELINA$_3$ which means the system can't intentionally plan failure into the design of its levels. Our current work includes expanding ANGELINA's analysis of the levels it designs, so that it can understand both critical level paths and optional or failed paths.

IV. ANGELINA 4: Creative Communities

The most recent version of ANGELINA attempts to take on more creative responsibility in the process of game design, in particular by broadening its space of possible inputs to allow it to design games based on free text input. This means that instead of being given a Guardian news article as in ANGELINA$_3$, or no theme at all as in the case of previous versions, it can now be given a word or a short phrase and try to make a game that somehow expresses this theme. This is an important step in the development of the larger ANGELINA project, as it allows the system to enter game design contests for the first time, since usually the only requirement is that the entrants design their games with a simple theme in mind.

A. Preliminary Design

As with the prior section, ANGELINA$_4$ has a predesign phase in which it gathers media for use in the CCE phase. We describe this process in more detail in [27]. Using frequency analysis on a corpus of English text, ANGELINA$_4$ extracts a word from the input phrase that is rare, but has a minimum number of mentions in the corpus. It then proceeds to download sound, images and music from the same corpora as ANGELINA$_3$, with the exception of 3D models which it obtains from TF3DM[11], something which is only relevant to ANGELINA$_4$ since its games are three-dimensional.

B. Main CCE System

ANGELINA$_4$ is composed of three species: a *level design* species which evolves arrangements of hand-designed tiles, in much the same way that ANGELINA$_3$ does; a *layout design* species which places the player, the exit and any other entities in the game world, similar to previous version of ANGELINA; and a *rule design* species which selects and applies rules from a premade database of rules to the game world, shaping the mechanics of the game and the objectives of the player. A detailed exploration of ANGELINA$_4$'s species and the evolutionary parameters can be found in [27].

[11]http://www.tf3dm.com

This is a game about a disgruntled child. A founder. The game only has one level, and the objective is to reach the exit. Along the way, you must avoid the Tomb as they kill you, and collect the Ship.
I use some sound effects from FreeSound, like the sound of Ship. Using Google and a tool called Metaphor Magnet, I discovered that people feel charmed by Founder sometimes. So I chose a unnerving piece of music to complement the game's mood.

Fig. 6. Title screen and excerpted commentary of *To That Sect*.

V. Evaluation

In [2] we described a mixed evaluation approach for automated game design systems, looking at properties of the design space, fitness growth, and other features. Such approaches are useful for providing hard objective metrics for computational systems, but evaluating the degree to which a system is creative, or seeking objective measures for assessing media like art or videogames, is a much harder task.

In this section we describe some approaches we have taken to evaluating ANGELINA as a piece of creative software – by allowing stakeholders in the creative process to pass judgement on the software; through formal models of the creative process; and through extensions to existing taxonomies for procedural content generation.

A. Game Jam Participation

One way in which we could evaluate automated game designers would be to ask people to evaluate its output. There are many different stakeholders in the games industry who we might ask, such as other researchers, journalists, developers, gamers, and so on. The difficulty with this approach, however, is that we are dealing with subjective opinions of highly contentious concepts like creativity. In [18] we assert that creativity is an 'essentially contested' concept, meaning that part of its function in society is to be argued over and disagreed upon, in order to continue to ferment change and progress in creative disciplines. Furthermore, there is evidence that both positive bias [28] and negative bias [29] can affect evaluations of software that acts in creative domains such as videogame design.

In order to investigate some of these issues, we have begun a regular effort to enter ANGELINA into game jams. A *game jam* is a time-limited contest in which entrants make a videogame from scratch, including artwork, music and

1943-068X (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2016.2520305, IEEE Transactions on Computational Intelligence and AI in Games

	To That Sect	Stretch Bouquet Point
Overall	500	551
Fun	515	543
Audio	211	444
Graphics	441	520
Mood	180	479
Innovation	282	525
Theme	533	545
Humour	403	318

Fig. 7. Rankings for ANGELINA$_4$'s two games entered into Ludum Dare 28. There were 780 total submissions to this track. Lower rankings are better.

programming. Jams are often organised around themes, which entrants try to include in their game designs. The theme is usually employed to stimulate creativity by forcing people to come up with a new idea at the start of the contest. Themes can be simple words or phrases like 'fishing' or more complex concepts. For instance, the Public Domain Jam tasks entrants with making games based on properties with expired copyrights, such as the works of Lovecraft or Shakespeare, for example, while the Global Game Jam[12] has featured both images and audio files as their themes, allowing looser and more unusual interpretations.

Game jams are an important part of modern videogame culture, both for the consumers of games – for whom the jams are a source of new and undiscovered games, some of which are eventually developed into large commercial releases – and the developers – for whom the events represent an important exchange of ideas and criticism, and where all entrants are theoretically equal, with major commercial programmers entering their games alongside first-time student developers. The Global Game Jam registered 23,198 entrants in January 2014, while Ludum Dare[13] saw 2,497 games submitted in 72 hours during their April 2014 jam.

ANGELINA has entered three game jams at the time of writing, both organised by Ludum Dare in December 2013, and April and August 2014 respectively. Ludum Dare follows a very traditional game jam format. The theme is a short phrase, and entrants are given either 48 or 72 hours to develop a game, depending on which track of the jam they wish to enter – the 72 hour track relaxes the rules to allow external assets to be used, which means ANGELINA can enter this track despite not creating its own artwork. After games are submitted to Ludum Dare, a three-week rating period takes place where those who developed games can rate other games according to eight categories: Fun, Audio, Graphics, Mood, Innovation, Theme, Humour and Overall. These rankings are each out of five, and reviewers do not need to enter a rating for each category (if they feel it is not applicable, for instance a game with no audio).

To investigate the response to ANGELINA's participation in such an event for the first time, we entered two of ANGELINA$_4$'s games to Ludum Dare 28, which was held in December 2013. The first game, *To That Sect*, was submitted with a commentary by ANGELINA$_4$ (similar to the commentary in Figure 1) along with a descriptive text written by us which identified ANGELINA$_4$ as a piece of software and asked the reviewers to rate the game as objectively as possible, without regarding the fact that it was created by software. The second game, *Stretch Bouquet Point*, was submitted with a modified commentary by ANGELINA$_4$ which fixed the grammar and removed text which implied the creator was a piece of software. We did not include any other descriptive text with the submission that identified ANGELINA$_4$ as the author, intending to compare the reactions to both games. This was not intended, in any way, to be a Turing test-style evaluation - instead, we were interested in assessing whether *positive* bias existed in the reviews given, to assess how reliable this might be as a form of evaluation.

The event organisers were unable to provide us with specific rating data for the games, even after many requests. However, the final rankings are provided in Table 7. The game which openly identified ANGELINA$_4$ as a piece of software ranked consistently higher than the anonymised game in all but one category, including noticeable gulfs in Mood, Audio and Innovation. The latter category is particularly notable since neither game is particularly innovative. Comments left by the reviewers reveal that people ranked the game as innovative because they believed the *software* to be innovative, rather than the games.

We interpret this as evidence that a positive bias can exist when evaluating automated game designers, either because of the novelty perceived by people in a piece of software creating games (as one reviewer of *To That Sect* put it: "it's certainly not something you see every day") or a positive cultural reaction to technology, given their background in videogames and computing. Procedural content generation is a large part of modern videogames culture, and as a result people are already accepting of related concepts, which might not be found in other media.

The evidence of positive bias suggests that evaluating automated game designers through direct engagement with people is currently unreliable. While all evaluation is subjective, even of human-created artefacts, we nevertheless should be aware when additional bias is present in evaluations. We believe that such positive bias will ultimately disappear in time, particularly as more automated game designers emerge and their participation in community events such as game jams become more common. Through saturation, the novelty associated with the concept will disappear, and more realistic evaluations will hopefully take their place.

B. Showing Progress In Creativity

Another possible way to evaluate autonomous game designers, regardless of an ultimate aim of creating a system which is perceived as creative, may be to consider what *kinds* of creative activity the software engages in. By comparing the creative actions the software makes, and showing how the number and type of these actions change between subsequent versions of the software, we can argue that the system is gradually making progress towards being independently creative. In section II-B we described the FACE model, a Computational Creativity formalism for describing progress between different versions of

[12]http://globalgamejam.org/

[13]http://www.ludumdare.com

1943-068X (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2016.2520305, IEEE Transactions on Computational Intelligence and AI in Games

creative software, or comparing two creative systems. In [30] we perform a preliminary FACE evaluation of ANGELINA$_3$ and ANGELINA$_2$, and use the model to show that progress has been made between the two versions of the software.

Under the latest version of the model, which has developed further since we published our initial assessment of ANGELINA$_3$, we now describe ANGELINA$_2$ with the tuple $< C^g, E^g >$, demonstrating the CCE system's basic ability to work with concepts and expressions of those concepts (referring to the genotype and phenotype representations of game content, which the system is capable of generating, manipulating and converting into concrete game code). ANGELINA$_3$ is described with the tuple $< F^g, C^g, E^g >$, which shows the addition of new types of creative acts, in this case the addition of framing activities such as the use of live web data and the generation of commentaries. This description is possible because the system is capable of giving justifications for its creative decisions, and placing its work within a wider cultural context. We argue that the addition of elements to the tuple shows progress between versions of ANGELINA.

The FACE model is still under development, meaning it is difficult to apply to complex acts of creation, particularly those combining different media domains (which are common in the design of videogames, an inherently multimedia domain). The FACE model makes no guarantees to the quality of the output of a system, nor is there any suggestion that systems which are more creative under the FACE model are perceived to be more creative by human evaluators, except within the field itself. As the model develops, we expect many of these features will be added to the model, making it a serious tool for the analysis of software generating artefacts such as videogames. However, future development of the model may render earlier analyses of systems no longer applicable, as the understanding of different elements of the model may shift, and new definitions may be introduced.

C. Curation and Quality

In [20], Ritchie describes several criteria which can be used to assess a system which generates content of some kind. The criteria are defined mathematically, and so can be applied precisely to a system, assuming certain definitions can be made, such as a notion of 'quality' for the domain the system works within. Many of the criteria refer to the system's output directly, which Ritchie calls the result set. This distinguishes the approach from other ways of evaluating systems, such as the FACE model described in section II-A, by focusing purely on the system's input and output, rather than the system itself. Inspired by Ritchie's criteria, Colton and Wiggins put forward the idea of a *curation coefficient* in [28]. A curation coefficient is the percentage of a system's output that the system's designer would be happy to show to other people. Llano et al later describe the curation coefficient as 'an informal reading of the typicality, novelty and quality measures put forward by [Ritchie]' [31]. The three measures mentioned are defined as follows: *typicality* is the degree to which an artefact is identifiable as belonging to the class of artefacts the system is interested in generating (in this case, videogames); *novelty* is the degree to which artefacts similar to this have been seen before; *quality* is a measure of how good the artefact is, a measure of its value. Curation is, as the name suggests, entirely subjective - it is simply based on the opinions of the system's designers, or whoever else is performing the curation task.

Showing a rise in the curation coefficient can indicate an improvement in the performance of creative software, since it indicates that the system is moving towards a point where it can produce and distribute output without intervention or filtering from a person such as the system's designer. Considering a sample of 30 games each from ANGELINA$_1$, ANGELINA$_3$ and ANGELINA$_4$, we calculated a curation coefficient of 33% for ANGELINA$_1$'s games, 60% for ANGELINA$_3$'s games and 80% for ANGELINA$_4$'s games. While this may only be considered a baseline of quality – the games are certainly not culturally influential or innovative – it demonstrates an increasingly confidence in the system as we step towards a more stable and reliable framework for automatically designing simple games.

VI. Classifying Game Designers

A. Extensions to the PCG Taxonomy

In section II-A we described a taxonomy proposed in [3] for categorising search-based procedural content generation systems according to the manner in which they generate content. The classifiers help to define a wide space of PCG systems and highlight many important distinctions that can be made about such systems. The taxonomy speaks to the role of the PCG system within the wider game (*Online/Offline* and *Necessary/Optional*), the variability and repeatability of the system's execution (*Seeds/Vectors* and *Stochastic/Deterministic*) and the generative approach taken to content creation (*Constructive/Generate-and-Test*). The taxonomy does not offer a way to talk about PCG systems which are composed of multiple generators, or the ways in which multiple generators may interact, however. This is understandable as PCG systems tend to be perceived as black boxes whose internals are not typically studied or analysed. However, this need not be the case, and in order to analyse systems which autonomously design games, including versions of ANGELINA, it will be helpful to be able to talk about the degree to which the content generators inside an autonomous game designer interact and execute in conjunction with one another.

We propose two further classifiers for search-based procedural content generation systems here. These new classifiers are based on the premise that content generation often occurs in distinct stages, in which multiple types of content may be generated. For example, Ed Key's *Proteus* [32] takes place on a procedurally generated island. While this can be thought of as a single generator under the taxonomy in [3] that is *online*, *seeded*, producing *necessary* content, and so on, it is also informative to consider it as a system composed of many generative steps. In particular, the heightmap of the island is generated separately from the placements of the landmarks and creatures, or the assignment of vegetation and varieties of flora.

1943-068X (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2016.2520305, IEEE Transactions on Computational Intelligence and AI in Games

By considering such a system as performing many generative steps, we can classify it in greater detail and build up a more expressive taxonomy of procedural systems, particularly as we expand our knowledge in procedural content generation to automated game design systems that may tackle several generative tasks simultaneously. For a series of procedural content generation tasks, our new classifiers distinguish the degree to which the tasks are interrelated (*dependence-independence*) and whether the generative acts are interleaved (*sequential-parallel*). Both of these features are useful in analysing our approach with ANGELINA as well as distinguishing related work in automated game design from other existing work in procedural content generation.

1) Dependent versus Independent: This distinction is between generators which produce output with no regard for the content that has already been generated, and those which generate dependent on what is already in the game. Given a system with a set of static content and one or more generative tasks yet to be completed, independent generative systems will produce their output only considering the static pre-existing game design, or may have no consideration at all for what content exists. For example, early versions of the village placement algorithm in the world generator of Minecraft [10] put villages in the world with no consideration for the surrounding geography. Dependent systems will use the content generated already as a basis for making decisions about what content should be generated next. Many games in the *Roguelike* genre, such as the eponymous *Rogue* [33], place items and monsters in levels after the level geometry has been generated, ensuring that content is evenly spread throughout the world.

The Dependent/Independent classifier is a linear rather than a binary classifier. Some games exhibit high levels of dependence in their generation such as *Dwarf Fortress* [11], whose world generation is performed in discrete stages which use geographical and geological models to place rivers, settlements and natural features according to the features of the generated world's topology. Other games exhibit some dependence in their generators but not as comprehensively, only having small amounts of information shared between generators. For instance, *Spelunky* [5] places monsters and items with little regard for level flow, but always ensures level exits are accessible based on the level layout.

2) Sequential versus Parallel: This distinction separates generators whose tasks are performed one after another, with no interleaving between generative steps, and generators which perform multiple generative tasks simultaneously. While this is largely a binary classification, we can imagine generators which perform some computation in parallel but whose generative steps are mostly performed sequentially – e.g., a system which generated a dungeon one room at a time, first generating the topology of the room, then filling it with objects. While the generative steps are performed sequentially, the dungeon itself is being produced in parallel at the level of rooms.

Most procedural content generation in commercial or hobbyist videogame development is performed sequentially. We believe that the reasons for this include a more direct analogy with human content generation, and a conceptually simpler system design. For example, ambient exploration game Proteus [32] designs its islands in distinct stages that complete fully before the next generative step begins. Parallel examples are rare, but work in [34] using multiple agents to lay content in a level simultaneously can be understood as a parallel content generation system. In this case, each agent is placing content at the same time, allowing interactions between the separate content-generating agents.

	ANGELINA	GoM	Variations Forever
Dependence	High	High	High
Parallelism	High	Low	High

Fig. 8. A table classifying several systems (ANGELINA, the Game-o-Matic, and Variations Forever) in the domain of videogame design, according to our extensions to the PCG taxonomy.

B. Classifying ANGELINA and Similar Systems

Under the original PCG taxonomy given in [3], all versions of ANGELINA are offline PCG system, which produces both necessary and optional content. While the system contains many parameter vectors, these are typically not varied very often, and the system more often relies on its random seed to affect how its output varies. Many of its fitness functions can be altered to change the kinds of game content that it ranks highly, however. ANGELINA's generative processes are stochastic, particularly where its interactions with web data are concerned, as this varies the output greatly regardless of parameterisations, and produces content using generate-and-test, due to the nature of evolutionary systems.

In our extension to the taxonomy, ANGELINA is classified as a highly parallel, highly dependent system. All aspects of game generation, with the exception of ANGELINA's preliminary design phase, happen simultaneously, and all of these generative processes influence one another by affecting the playouts which contribute to the evaluation of game content.

Figure 8 shows a tabular summary of two other design systems, alongside ANGELINA, classified under the extensions to the PCG taxonomy. The first of these systems is the *Game-o-Matic*, a tool for automatically generating games that portray relationships between concepts defined by a human designer. By defining concepts (such as *police* and *protester*) and the relationships between them (such as *arrests* or *escapes*) the Game-o-Matic can produce playable games in which the relationships between objects are expressed as mechanics that interact between agents in a game world. For instance, given a relationship graph expressing *police arrests protester*, a resulting game may have the player taking the role of a police officer attempting to arrest protester objects by colliding with them, or controlling a protester who must evade police objects to avoid being arrested. Under the extensions to the PCG taxonomy proposed earlier in this section, the Game-o-Matic is a sequential but highly dependent system. The design proceeds in distinct phases which build upon the design generated in the previous phase.

The second system is *Variations Forever*, a system for designing simple game configurations using answer set programming (ASP). ASP is proposed as a new method for procedural

1943-068X (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2016.2520305, IEEE Transactions on Computational Intelligence and AI in Games

content generation and a key contribution in [35] is arguing for its strengths, which include simple constraint addition such as requiring that the games produced have a particular feature (e.g. the player moves using a particular control scheme). Classification of the work under our extensions to the PCG taxonomy is not straightforward as the structure of an answer set solver is dissimilar to the most common approaches to procedural content generation.

The output of an answer set solver, a *stable model* showing possible solutions to the expressed set of constraints in the input answer set program, is computed through a combination of translation and constraint solving often using satisfiability solvers. In the case of a system such as the one in Variations Forever, the entire output is generated in a single pass by considering the entire answer set program, so we may consider it to be a parallel generator. Additionally, since constraints express dependencies between different aspects of the generated content, answer set-driven content generation should be considered dependent as well.

VII. Future Work

The challenges of expanding procedural content generators into automated game designers, and the complexities of dealing with concepts like creativity within that work, offers a broad and varied selection of future work challenges. This section describes some major areas we hope to investigate in the near future.

A. More Detailed Framing

In section II-B we described the notion of framing from Computational Creativity, which we introduced into our project with the development of ANGELINA$_3$, a system which was capable of producing short commentaries describing the decisions it had made in designing a particular game. Framing in ANGELINA is currently restricted to post-hoc template-based textual descriptions of decisions made during the design process. While this is adequate for many creative systems, and similar to how people frame their work in domains such as art, it misses out on one of the key strengths of videogames as a domain: the interactivity of the medium. Many game developers use *developer commentary* as a means of communicating framing information to the player of a game, where interactive nodes appear inside the game which play audio clips to the user when interacted with. These audio clips describe aspects of the game that they are situated nearby – in some advanced cases, activating a node can cause specific code to execute, demonstrating to the user a cut piece of content, or showing them alternative visual effects or 3D models that would not be seen in normal play.

Taking this approach to framing allows ANGELINA to have more autonomy in deciding what to comment on, and how to comment upon it. Most importantly, the notion of 'cut content' offers up a possibility for the system to explicitly discuss alternatives it considered but turned down, which may lead to a greater perception of creativity in the system as it is seen to be explicitly considering options. Discussion of choices not made is not something that has been investigated from a Computational Creativity perspective, making this an interesting avenue to explore.

B. Creating Meaning

Game designer and critic Anna Anthropy defines a game as 'an experience created by rules' [36]. Conveying meaning through rules can be as simple as falling objects in a 2D game conveying the concept of gravity, or they can be much more sophisticated and complex. In the game *By Your Side* by Alan Hazelden, the player controls a groom and bride who initially mimic each other's input. The puzzles are solved by navigating the characters together to an exit. Later in the game, the bride's behaviour switches to make the opposite move to that of the groom. The rules of the game change to convey something about the relationship between the two characters.

Incorporating such meaning into games is difficult for any designer, and especially so for an automated system. This is a complex task that requires an understanding of the real world on a factual and cultural level, as well as an understanding of the shared vocabulary that videogame consumers use when describing depictions of real-world concepts in videogames. In [37] we propose a solution to this latter problem in particular, namely to give automated game designers the ability to create a vocabulary of their own. If ANGELINA can relate game concepts to real world concepts in a consistent way, then we believe people may be willing to accept the fact that ANGELINA's vocabulary differs from all other game designers, because we can demonstrate the system's intelligence and understanding through framing and repeated use of the vocabulary in a reliable manner. Our hope is that such consistency will help increase the perception of creativity in the system, and expand the things that ANGELINA can express through the games it creates.

VIII. Conclusions

In this paper, we outlined versions of ANGELINA which transitioned the system from a merely generative automated game designer into a system which we believe is taking steps towards being accepted as a creative individual. By giving the system the capability to interact with real world contexts, allowing it to evaluate its own work, thus providing it more responsibility in the creative process, we have progressed the system towards being a member of the creative community that makes up modern games development.

We presented some existing methods of evaluating creative software, and of categorising procedural content generators, and applied them both to ANGELINA. In doing so, we were able to propose extensions to the PCG taxonomy, and also offered an evaluation under the FACE model that will enable researchers in automated game design to compare their progress against our own work. We also raised questions that go beyond formal evaluation methods, by showing the responses to ANGELINA from different stakeholders in the creative community – journalists, games designers, and the players themselves. Their relationship with automated game designers will be increasingly important as we seek to get our software to engage with the community directly, and

1943-068X (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2016.2520305, IEEE Transactions on Computational Intelligence and AI in Games

their perception of our software will be the ultimate test of acceptance for creative software aspiring to design games.

Computational Creativity is a blossoming and vast new field of artificial intelligence, and videogames represent a contemporary, engaging and incredibly complex creative domain to work within. Automated game design is the area of study that can bring these two fields together and allow us to tackle a dizzying array of exciting problems [38]. We hope that the work we have put forward in this paper will offer a useful inspiration for many new projects in automated game design.

ACKNOWLEDGMENTS

We would like to thank Phillipe Pasquier, Alison Pease, Azalea Raad, Adam Smith, Julian Togelius, Joe Martin, Mike Treanor and Tony Veale for many fruitful conversations over the course of this work's development. We also thank Michael Brough, Darren Grey, Ed Key and Jeff Lait for discussions that helped shape this paper. This project has been supported by EPSRC grants EP/L00206X and EP/J004049.

REFERENCES

[1] R. Khaled, M. J. Nelson, and P. Barr, "Design metaphors for procedural content generation in games," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2013, pp. 1509–1518.
[2] M. Cook, S. Colton, and J. Gow, "The ANGELINA videogame design system, part I," *Transactions on Computational Intelligence and AI in Games (Under Review)*, 2014.
[3] J. Togelius, G. N. Yannakakis, K. O. Stanley, and C. Browne, "Search-based procedural content generation: A taxonomy and survey," *IEEE Trans. Comput. Intellig. and AI in Games*, 2011.
[4] N. Shaker, J. Togelius, and M. J. Nelson, *Procedural Content Generation in Games: A Textbook and an Overview of Current Research*. Springer, 2015.
[5] D. Yu, "Spelunky," 2008, http://www.spelunkyworld.com.
[6] CCP Games, "Eve: Online," 2003, http://www.eveonline.com.
[7] Bethesda Game Studios, "The Elder Scrolls V: Skyrim," 2011.
[8] K. Perlin, "An image synthesizer," in *Proceedings of the 12th annual conference on computer graphics and interactive techniques*, 1985.
[9] Firaxis Games, "Civilization IV," 2005, http://www.civilization5.com/.
[10] M. Persson, "Minecraft," 2011, http://www.minecraft.net.
[11] T. Adams, "Dwarf Fortress," http://www.bay12games.com/dwarves/, 2006.
[12] V. Software, "Left 4 dead," 2009.
[13] G. Smith, "Understanding procedural content generation: A design-centric analysis of the role of pcg in games," in *Proceedings of the ACM CHI Conference on Human Factors in Computing Systems*, 2014.
[14] S. Colton, J. Charnley, and A. Pease, "Computational Creativity Theory: The FACE and IDEA models," in *Proceedings of the Second International Conference on Computational Creativity*, 2011.
[15] A. Pease and S. Colton, "Computational Creativity Theory: Inspirations behind the FACE and the IDEA models," in *Proceedings of the Second International Conference on Computational Creativity*, 2011.
[16] S. Colton, J. Goodwin, and T. Veale, "Full-FACE poetry generation," in *Proceedings of the Third International Conference on Computational Creativity*, 2012.
[17] K. Monteith, T. Martinez, and D. Ventura, "Automatic generation of melodic accompaniments for lyrics," in *Proceedings of the Third International Conference on Computational Creativity*, 2012.
[18] S. Colton, M. Cook, R. Hepworth, and A. Pease, "On acid drops and teardrops: Observer issues in computational creativity," in *Proceedings of the 7th AISB Symposium on Computing and Philosophy*, 2014.
[19] S. Colton, A. Pease, J. Corneli, M. Cook, and T. Llano, "Assessing progress in building autonomously creative systems," in *Proceedings of the Fifth International Conference on Computational Creativity*, 2014.
[20] G. Ritchie, "Some empirical criteria for attributing creativity to a computer program," *Minds and Machines*, vol. 17, no. 1, 2007.
[21] T. Lindeijer, "Tiled," http://www.mapeditor.org/.
[22] A. Krzeczkowska, J. El-Hage, S. Colton, and S. Clark, "Automated collage generation with intent," in *Proceedings of the First International Conference on Computational Creativity*, 2010.
[23] T. Veale, "Less rhyme, more reason: Knowledge-based poetry generation with feeling, insight and wit," in *Proceedings of the Fourth International Conference on Computational Creativity*, 2013.
[24] ——, "From conceptual "mash-ups" to "bad-ass" blends: A robust computational model of conceptual blending," in *Proceedings of the Third International Conference on Computational Creativity*, 2012.
[25] F. Å. Nielsen, "A new anew: Evaluation of a word list for sentiment analysis in microblogs," *Computing Research Repository*, 2011.
[26] M. Cook and S. Colton, "Automated collage generation – with more intent," in *Proceedings of the Second International Conference on Computational Creativity*, 2011.
[27] "Ludus ex machina: Building a 3D game designer that competes alongside humans," in *Proceedings of the Third International Conference on Computational Creativity*, 2014.
[28] S. Colton and G. A. Wiggins, "Computational creativity: The final frontier?" in *Proceedings of the European Conference on AI*, 2012.
[29] D. Moffat and M. Kelly, "An investigation into people's bias against computational creativity in music composition," in *Proceedings of Third Joint Workshop on Computational Creativity*, 2006.
[30] M. Cook, S. Colton, and A. Pease, "Aesthetic considerations for automated platformer design," in *Proceedings of the Artificial Intelligence and Interactive Digital Entertainment Conference*, 2012.
[31] M. T. Llano, R. Hepworth, S. Colton, J. Gow, J. Charnley, N. Lavrac, M. Znidaršic, M. Perovšek, M. Granroth-Wilding, and S. Clark, "Baseline methods for automated fictional ideation," *Proceedings of the 5th International Conference on Computational Creativity*, 2014.
[32] E. Key, "Proteus," 2012, http://www.visitproteus.com.
[33] M. Toy and G. Wichman, "Rogue," http://www.hexatron.com/rogue/oldrogue.html, 1980.
[34] M. Kerssemakers, J. Tuxen, J. Togelius, and G. N. Yannakakis, "A procedural procedural level generator generator," in *Proceedings of the IEEE Conference on Computational Intelligence and Games*, 2012.
[35] A. Smith and M. Mateas, "Variations forever: Flexibly generating rulesets from a sculptable design space of mini-games," in *Proceedings of the IEEE Conference on Computational Intelligence and Games*, 2010.
[36] A. Anthropy, *Rise Of The Videogame Zinesters: How Freaks, Normals, Amateurs, Artists, Dreamers, Drop-outs, Queers, Housewives, and People Like You Are Taking Back an Art Form*. Seven Stories Press, 2012.
[37] M. Cook and S. Colton, "From mechanics to meaning and back again: Exploring techniques for the contextualisation of code," in *Proceedings of the AI & Game Aesthetics Workshop at AIIDE*, 2013.
[38] A. Liapis, G. Yannakakis, and J. Togelius, "Computational game creativity," in *Proceedings of the Fifth International Conference on Computational Creativity*, 2014.

Michael Cook gained a PhD in Artificial Intelligence from Imperial College, London in 2015. His research interests include the automation of game design, applications of computational creativity to videogames, and computational models of meaning in videogames. He is currently working at the University of Falmouth as a Senior Research Fellow in the Computational Creativity Group, and is the organiser of the annual Procedural Generation Jam. He maintains a research log at www.gamesbyangelina.org.

1943-068X (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2016.2520305, IEEE Transactions on Computational Intelligence and AI in Games

Simon Colton is Professor of Digital Games at the University of Falmouth. He was awarded a PhD. in Artificial Intelligence from the University of Edinburgh in 2001. He has published more than 150 papers on a range of AI topics, as well as application papers ranging from poetry generation to the automated discovery of new mathematical concepts, and philosophy of Computational Creativity. He is well known for his software The Painting Fool, the aim of which is to be taken seriously as a creative artist in its own right one day.

Jeremy Gow received a BSc. in Artificial Intelligence & Mathematics from the University of Edinburgh, UK in 1997, and a PhD. in Informatics from the same institution in 2004.

He is a Lecturer in Games Programming at the Computational Creativity Group, Department of Computing, Goldsmiths, UK. His research interests are the modelling of behaviour and experience in interactive systems, and the use of learning agents in design and creative activities, with a particular interest in play and video game design. He has published over 50 research papers on topics in AI and HCI.

1943-068X (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.