Experience-Driven Procedural Content Generation

Georgios N. Yannakakis, *Member*, *IEEE*, and Julian Togelius, *Member*, *IEEE*

Abstract—Procedural content generation (PCG) is an increasingly important area of technology within modern human-computer interaction (HCI) design. Personalization of user experience via affective and cognitive modeling, coupled with real-time adjustment of the content according to user needs and preferences are important steps toward effective and meaningful PCG. Games, Web 2.0, interface, and software design are among the most popular applications of automated content generation. The paper provides a taxonomy of PCG algorithms and introduces a framework for PCG driven by computational models of user experience. This approach, which we call *Experience-Driven Procedural Content Generation* (EDPCG), is generic and applicable to various subareas of HCI. We employ games as an example indicative of rich HCI and complex affect elicitation, and demonstrate the approach's effectiveness via dissimilar successful studies.

Index Terms—Procedural content generation, user affect, user experience, personalization, adaptation, computer games.

1 Introduction

As information about users is becoming more readily available for all kinds of digital services and modern software development relies upon content creation, opportunity and demand for automatically generated personalized content increases in domains as diverse as e-commerce, news reading, web 2.0 services, human-computer interfaces, and computer games. Ideas and technology from computer games, including rich interactivity, three-dimensional graphical visualization, and role playing game-style incentive structures, are more and more pervasive in the aforementioned domains (a phenomenon referred to as "gamification"). By viewing games as one of the most representative examples of content creation applications, but also as elicitors of complex user emotion syntheses, we explore ongoing research on procedural content generation and propose Experience-Driven Procedural Content Generation (EDPCG) as a generic and effective approach for the optimization of user (player) experience. On that basis, we view *player experience* as the synthesis of affective patterns elicited and cognitive processes generated during gameplay.

Recent years have seen both a boost in the size of the gaming population and a demographic diversification of computer game players [1]. Twenty years ago, game players were largely young white males with an interest in technology; nowadays, gamers can be found in every part of society [2]. This means that skills, preferences, and emotion elicitation differ widely among prospective players of the same game. In order to generate the same gameplay experience, very different game content will be needed, depending on the player's skills, preferences, and emotional profile [3]. Therefore, the need for tailoring the game to individual playing experience is growing and the tasks of user (affective and/or cognitive) modeling and affective-based adaptation within games becomes increasingly difficult. Game engines [4] that are able to recognize and model the playing style and detect the affective state of the user will be necessary milestones toward the personalization of the playing experience, as will procedural mechanisms that are able to adjust elements of the game to optimize for the experience of the player.

1.1 Affective Games

Affective computing [5] research views the successful realization of the *affective loop* [6], [7], [8] as one of the ultimate goals behind the study of emotions within HCI. The phases of emotion elicitation, affective detection and modeling, and affect-driven system adaptation are critical toward a closed-loop affective-based HCI for emotion elicitation users are, in general, either asked to act a particular emotion (e.g., via guided imaginary [9]) or specific stimuli are provided via the interaction. Computer games, being generators of immersive and rich HCI experiences, are able to elicit a great variation of emotions and complex patterns of affect of the player. Games offer rich and fast-paced interaction with dynamic elements coupled with narratives which are hand-crafted to yield particular patterns of player experience. This form of interaction elicits complex emotional responses for the player, the detection of which is far from trivial. For some psychologists and game designers, the emotions elicited by games (and HCI in general) are not genuine emotions but rather *quasi* emotions [10], [11]. It is questionable, for instance, who would play a game that can cause genuine fear in its players. Thus, games as affect elicitors challenge the findings of affective computing derived from simple laboratory-based experimental designs, while their rich interaction opens up new perspectives for the study of affect detection.

The detection, modeling, and synthesis of player experience is not trivial either since emotions are conceptual

• *The authors are with the Center for Computer Games Research, IT University of Copenhagen, Room 3B02, Rued Langgaards Vej 7, DK-2300 Copenhagen S, Denmark. E-mail: {yannakakis, juto}@itu.dk.*

Manuscript received 27 Aug. 201; revised 12 Dec. 2010; accepted 22 Feb. 2011; published online 16 Mar. 2011.
Recommended for acceptance by A. Hanjalic.
For information on obtaining reprints of this article, please send e-mail to: taffc@computer.org, and reference IEEECS Log Number TAFFC-2010-08-0064.
Digital Object Identifier no. 10.1109/T-AFFC.2011.6.

constructs and emotional states are entities with unclear boundaries [12]. Nevertheless, so far a considerable amount of different game genres has been investigated, varying from simple arcade games such as pong [13], [14], tetris [15], and quiz games [16], to racing games [17], [18], physically interactive games [19], prey/predator games [20], first-person shooters [21], [22], and games for education [23]. The above studies mostly focus on the interaction of the player with nonplayer characters (NPCs) and no particular emphasis is given to the content of the game and its impact to the player's affective state. We argue that a holistic approach for affective synthesis in games requires the integration of game content to the computational model of affect.

To successfully close the affective loop [8] within games one needs to fulfill a set of system requirements: The game should be tailored to individual players' affective response patterns, the game adaptation should be fast, yet not necessarily noticeable, and the affect-based interaction should be rich in terms of game context, adjustable game elements, and player input. The EDPCG approach proposed in this paper satisfies those conditions via the efficient generation of game content which is driven by models of player experience. Those computational models can be built on multiple modalities of user input.

1.2 Procedural Content Generation

Procedural content generation (PCG) refers to the creation of content automatically through algorithmic means. Procedural content generation is tied to several research areas such as computational aesthetics and computational creativity in general [24], and recommender systems. However, in this paper we will focus on and discuss PCG for games and, at the end of the paper, we will return to how the ideas expressed here can be applied to other domains and research areas. Thus, we will start with defining content in games.

Game content refers to all aspects of a game that affect gameplay but are not nonplayer character behavior or the game engine itself. This definition includes such aspects as terrain, maps, levels, stories, dialogue, quests, characters, rulesets, camera profiles, dynamics, music, and weapons. The definition explicitly excludes the most common application of learning and search techniques in academic games research, namely, NPC artificial intelligence.

When it comes to the development of a modern computer game, the effort and time required for the creation of game content represents a large part of the development cost (and time). One would expect that with the rapid advancement of all forms of digital technology, this process would be rather streamlined and partly automated by now. But while video game technology has advanced by leaps and bounds from the pixel-based graphics and predictable interaction of *Breakout* and *Pac-Man* to the elaborate realistic 3D environments, rich dynamics, and graphical detail of titles such as *Halo* and *Call of Duty*, content creation is still largely manual. Usually a team of people from different departments of game production is responsible for hand-crafting of all the content in a game.

The cost of developing a top-tier computer game has increased by orders of magnitude in the last two decades. (For example, Rockstar's *Grand Theft Auto IV* is the work of 1,000 people over a period of three years.) This "content creation" bottleneck is a barrier to the artistic and technological progress of computer games as few developers can afford to try out new, risky ideas with this kind of stake involved.

Clearly, any technology that can alleviate the enormous burden of content creation and make it easier to tailor content to individual players or groups of players would be warmly welcomed by game developers, game critics, and the game-playing public in general—especially if this technology can also automatically adapt the game to the needs and preferences of individual players. This argument extends to games with a purpose beyond pure entertainment. Game-based technologies, and often complete games, are used more and more for simulation, training, education, and decision support in many sectors of society. And these games and simulations need content. Militaries need scenarios to train peace-keeping duties and simulate the consequences of tactical decisions [25]; rescue services need city layouts and buildings to train disaster relief workers [26]; companies in sectors from logistics to customer service to education use game-based simulations to train their employees and need scenarios for this. For training to be effective, the desired affective states of the trainees need to be reliably induced, reinforcing the need for basing the content generation on a model of the trainee's experience profile.

Attempts at generating game content procedurally have a fairly long history. Back in 1980, the game *Rogue* pioneered procedural generation through automatically generating dungeons for the player to explore. This game's endless replayability was a huge draw and it has been imitated numerous times, for example, by the fairly recent and commercially successful *Diablo* (Blizzard).

A few years later, the classic space trading and adventure game *Elite* (Acornsoft 1984) managed to keep hundreds of star systems in the few tens of kilobytes of memory available on the hardware of the day by representing each planet as just a few numbers. In expanded form, the planets had names, populations, prices of commodities, etc.

In modern days, procedural content generation is almost only used in narrowly specialized roles and almost always during development of the game. The probably most widespread technique is *SpeedTree*, which automatically generates large numbers of similar but not identical trees for populating terrains.

So why, if PCG has such a long history, is it not more widely used to generate all forms of game content? The reasons seem to be that:

1. Far from all types of game content can be satisfactorily generated with desired variability, reliability, and quality by traditional techniques.
2. Traditional PCG techniques are not *controllable* enough, meaning that not all important aspects of the generated content can easily be specified by the designer or by an algorithm. This is important as the content might need to be generated to fit into a particular section of a game, or even a particular player.

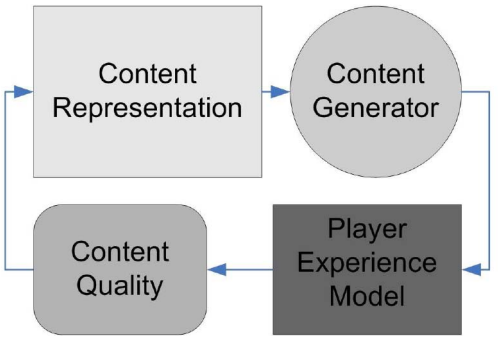


Fig. 1. The main components of the experience-driven procedural content generator.

It should also be noted that there has, until very recently, not been an academic community devoted to the study of PCG. This situation is now changing with the recent establishment of a mailing list,[1] an IEEE CIS Task Force,[2] a workshop,[3] and a wiki[4] on the topic, as well as an international PCG competition.[5] However, there is still no textbook on PCG and, to our knowledge, only a single short overview paper [27].

In the following, we will outline our approach to PCG, which is in part an attempt to overcome the problems faced by traditional PCG methods and in part a vision for new forms of personalized games and game development.

2 Experience-Driven Procedural Content Generation

Experience-Driven Procedural Content Generation (EDPCG) defines a novel approach to PCG. Even though embryos of EDPCG components can be found in the literature, the approach proposed here (and in the pilot studies referenced) is unique in linking player experience with procedural content generation.

We start by redefining content within the EDPCG framework. We view game content as building blocks of games, and games as potentiators of player experience. Therefore, content can be seen as indirect building blocks of player experience which define a vital control component of the affective loop in games. Since a game is synthesized by game content building blocks that, when played by a particular player, elicit player experience, one needs to assess the quality of the content generated (linked to the experiences of the player), search through the available content, and generate content that optimizes the experience for the player (see Fig. 1). In particular, the components of EDPCG are:

- Player experience modeling. Player experience is modeled as a function of game content and player (the player is characterized by her playing style, and her cognitive and affective responses to gameplay stimuli).
- Content quality. The quality of the generated content is assessed and linked to the modeled experience of the player.
- Content representation. Content is represented accordingly to maximize efficacy, performance, and robustness of the generator.
- Content generator. The generator searches through content space for content that optimizes the experience for the player according to the acquired model.

2.1 An Example: Personalized Level Creation in Super Mario Bros

Before delving into the details of these components, we will give the reader a feel for what EDPCG entails by providing an example from a recently published paper. We take our example from Pedersen et al. [28], who modified an open-source clone of the classic platform game *Super Mario Bros* to allow for personalized level generation.

The first step was to represent the levels in a format that would yield an easily searchable space. A level was represented as a short parameter vector describing the number, size, and placement of gaps which the player can fall through, and the presence or absence of a switching mechanic. This vector was converted to a complete level in a stochastic fashion, using an algorithm that built up the level from right to left, placing gaps according to the specified parameters.

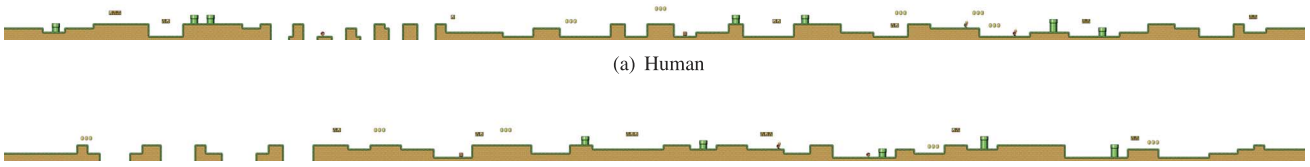
The next step was to create a model of player experience based on the level played and the player's playing style. Data was collected from hundreds of players, who played pairs of levels with different parameters and were asked to rate which of these two levels best induced each of the following affective states: fun, challenge, frustration, predictability, anxiety, and boredom. While playing, the game also recorded a number of metrics of the players' playing styles, such as the frequency of jumping, running, and shooting. This data was then used to train neural networks to predict the examined affective states using evolutionary preference learning. Automatic feature selection decided which subset of player data attributes was considered by each affective state predictor. The predictor of fun, for instance, was associated with the time spent moving left during a level and the number of enemies killed by stomping on them, whereas the predictor of frustration was linked to the time spent by the player standing still, the jump difficulty, the proportion of gameplay time within the last life, and the number of deaths due to falling into gaps [29].

Finally, these models were used to optimize game levels for particular players [29]. Two examples of such levels can be seen in Fig. 2. As seen from that figure, the level generated to maximize predicted fun for the current Super Mario AI champion (Fig. 2b)[6] contains large and challenging gaps, whereas the generated level of maximum fun value for the human (Fig. 2a) contains more gaps placed in a more unpredictable manner.

Assuming the playing style of a particular player is known, the level of each of the six affective states can be predicted for any particular level (expressed as a parameter

1. http://groups.google.com/proceduralcontent.
2. http://game.itu.dk/pcg/.
3. http://pcgames.fdg2010.org/.
4. http://pcg.wikidot.com.
5. http://www.marioai.org.

6. The Mario AI competition is about developing the best controller (agent) for Super Mario Bros—http://www.marioai.org/.



(a) Human

(b) World-Champion AI

Fig. 2. Example levels generated for two different Super Mario players. The levels generated maximize the modeled *fun* value for each player. The level on top depicts the level generated for one of the subjects who participated in our experiments, while the level below is the level generated for the world champion agent of the Mario AI competition.

vector) by simply feeding the level parameters together with the player parameters to the neural network. This means that the neural network can act as an evaluation function for black-box search or optimization, using, for example, evolutionary algorithms or exhaustive search.

While emotional response is only measured via self-reports (and not bodily reactions, for instance) in this study, our affective models rely upon the assumption that player emotions can be inferred via the association of user self-reports and game context variables [30], [31].

2.2 This Paper

Below, we survey the four main components of EDPCG and provide a taxonomy of different approaches to each and outline the main research challenges faced. We also give a nonexhaustive number of examples that fully or partly adopt the principles of EDPCG. Each component of EDPCG has its own dedicated literature and the extensive review of each would be beyond the scope of this paper. Thus, the survey attempts to highlight representative work that relates to the key components of EDPCG and discuss, in part, studies that cover central or peripheral principles of EDPCG.

Fig. 3 provides an overview of the EDPCG framework and serves as an illustration of the structure followed in the remaining of this paper. The three approaches to player experience modeling (subjective, objective, and gameplay-based), illustrated at the top of the figure, are presented in detail in Section 3. Section 4 presents the different types of content evaluation functions available (direct, simulation-based, and interactive). A discussion dedicated to content representation is provided in Section 5.1 and the generation component of EDPCG is covered in Section 5.2. The paper concludes with a summary of future visions for the EDPCG framework in Section 6.

3 Player Experience Modeling

Player experience models can be built on different types of data collected from the players, which in turn define different approaches to player experience modeling (PEM). We can identify three main classes of approaches for modeling player experience in games which rely on 1) data expressed by players (*subjective* PEM), 2) player data obtained from alternative types/modalities of player response (*objective* PEM), and 3) data obtained through the interaction between the player and the game (*gameplay-based* PEM). While the subjective and objective approaches emphasize both the affective and the cognitive aspects of playing experience, the gameplay-based approach focuses on the cognitive and behavioral components of it. The PEM approaches can be combined to more powerful hybrid methods for capturing player experience. The overview of the three approaches and their internal subclasses can be seen in Fig. 3.

3.1 Subjective PEM

The most direct way to develop a model of experience is to ask the players themselves about their playing experience and build a model based on these data. Subjective PEM considers only first person reports (self-reports) and not reports expressed indirectly by experts or external observers. Subjective player experience modeling can be based on either players' *free-response* during play or on *forced* data retrieved through questionnaires.

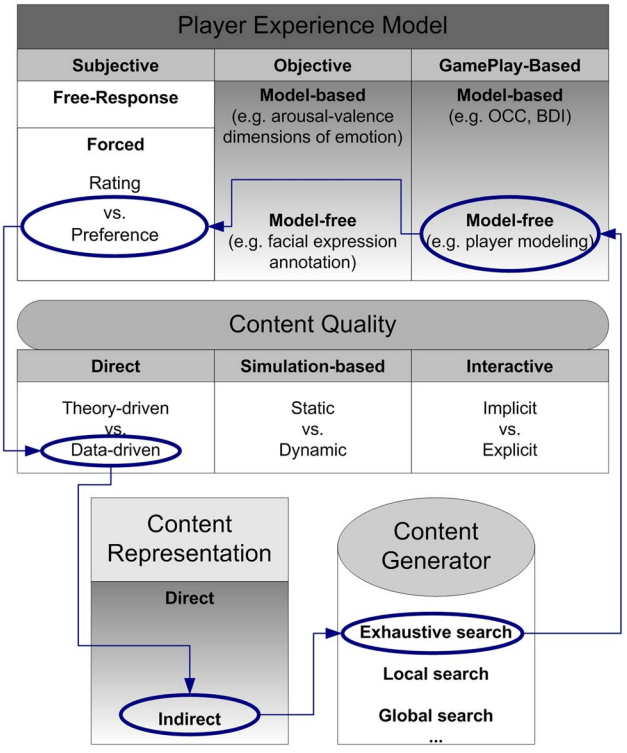


Fig. 3. The EDPCG framework in detail. The gradient grayscale-colored boxes represent a continuum of possibilities between the two ends of the box, while white boxes represent discrete, exclusive, options within the box. The blue arrows illustrate the EDPCG approach followed for the Super Mario Bros example study described in Section 2.1 [28], [29]: Content quality is assessed via a direct, data-driven evaluation function which is based on a combination of a gameplay-based (model-free) and a subjective (pairwise preference) player experience modeling approach; content is represented indirectly and exhaustive search is applied to generate better content.

Free-response naturally contains richer information about the players' affective state but it is hard to analyze appropriately. An experiment designer may decide to annotate the derived text or verbal response into specific critical words or phrases which can then be mapped to player experiences. However, doing so requires strong assumptions about the validity and the importance of the text/speech clusters identified.

Forcing players to self-report their experiences, on the other hand, constrains them into specific questionnaire items, which could vary from simple tick boxes to multiple choice items. Both the questions and the answers provided could vary from single words to sentences; even though, generally, short and clear question-and-answer items are preferred since lengthy questionnaire items may challenge the short-term memory and cognitive load of the player. Forced self-reports can be further classified as *rating*, in which the players are asked to answer questionnaire items given in a rating/scaling form [32], [21], [33], and *preferences*, in which players are asked to compare their player experience in two or more variants/sessions of the game [34], [35], [18].

Subjective player experience modeling may yield very accurate models of self-reported affective states [35]; however, there are quite a few limitations embedded in this approach. First, there is usually significant experimental noise in the responses of players; this may be caused by player learning and self-deception effects. Second, self-reports can be intrusive if questionnaire items are injected during the gameplay sessions [21], [33]; on the other hand, they are sensitive to players' memory limitations if players are asked to express their experience after a lengthy game session (postexperience effect). While efficient methods for minimizing learning effects and self-deception effects have been proposed [35], there is no universally accepted time window within which players should be asked to express their player experience. Such a time window should result in a self-reporting process that is both as unobtrusive as possible and suffering from minimal postexperience effects.

Numerous studies have shown that self-reports can guide machine learning algorithms for successfully capturing aspects of player experience in prey/predator [36], physical interactive [37], platform [28], [38], and racing [18] games.

3.2 Objective PEM

Player experience can be linked to a stream of emotions which may be active simultaneously, usually triggered by events occurring during gameplay. Games can elicit player emotional responses which, in turn, may affect changes in the player's physiology, reflect on the player's facial expression, posture, and speech, and alter the player's attention and focus level. Monitoring such bodily alterations may assist in recognizing and synthesizing the emotional responses of the player. The *objective* approach to player experience modeling incorporates access to multiple modalities of player input for the purpose of modeling the affective state of the player during play.

Within objective PEM, a number of real-time recordings of the player may be investigated for modeling affective aspects of player experience. There are several studies that explore the interplay between physiology and gameplay by investigating the impact of different gameplay stimuli to a number of dissimilar physiological signals. Such signals are obtained through electrocardiography (ECG) [21], [20], photoplethysmography [20], [18], galvanic skin response (GSR) [32], [17], [14], respiration [18], electroencephalography (EEG) [15], [39], electromyography (EMG), and pupillometry [40], [41] (note that the pupillometry studies do not involve games). Most of the above studies have revealed relationships between features of physiology and self-reports of players. Typical examples of these relationships include positive correlations between average heart rate [19], skin conductance [32], and player entertainment.

In addition to physiology one may track the player's bodily expressions (motion tracking) at different levels of detail and infer the real-time affective responses from the gameplay stimuli. The core assumption of such input modalities is that particular bodily expressions are linked to basic emotions and cognitive processes. Motion tracking may include body and head pose as well as gaze [42] and facial expression [43], [44].

Speech may also be used for inferring affective responses of the player [16], [45], but it is not directly applicable for the vast majority of existing game genres. Nevertheless, speech-based PEM is promising for future game implementations since it is completely unobtrusive and real-time efficient. A detailed review of speech-based affect recognition can be found in [46].

The objective PEM approach can be *model-based* or *model-free*. Model-based refers to emotional models derived from emotion theories (e.g., cognitive appraisal theory [47]) such as the popular emotional dimensions of arousal and valence [48], [49] in which bodily responses are mapped to specific emotional responses—e.g., the increased heart rate of a player corresponds to high arousal and therefore to player excitement. Model-free PEM refers to the construction of an unknown mapping (model) between modalities of player input and an emotional state representation via user annotated data. This approach is very common, for instance, for facial expression and head pose recognition since subjects are asked to annotate facial (or head pose) images of users with particular affective states (see [50] among others). Classification and regression techniques derived from machine learning or statistical approaches are commonly used for the construction of the computational model.[7]

Note that the space between a completely model-based and a completely model-free approach is a continuum, and any objective PEM approach might be placed somewhere along this axis. While a completely model-based approach relies solely on a theoretical framework that maps users' bodily responses to affect, a completely model-free approach assumes there is an unknown function between modalities of user input and affect that a machine learner or a statistical model may discover, but does not assume anything about the structure of this function. Relative to these extremes, all objective PEM approaches may be viewed as hybrids between the two ends of the spectrum,

7. One might claim that training on annotated data is a combination of subjective and objective PEM and not a purely objective PEM approach; however, we view the annotation of data as an indirect subjective PEM approach since users do not report on their *own* experience but rather on the potential experience of *other* users.

containing elements of both approaches. As a typical example of a hybrid approach, Mandryk and Atkins [51] built part of a computational model of emotion on physiological signal data while relying on a theoretical model of emotion (the *arousal-valence model*) for its structure.

Models built via the objective PEM approach may be very accurate representations of player experience since player experience is approached in a holistic manner via the use of multiple input modalities. While maximizing the amount of information available about the player through multiple modalities will most likely improve the model's accuracy, the complication of PEM increases. Therefore, a balance between the generated model's accuracy and computational and practical effort has to be kept.

The key limitations of the objective PEM approach include its high intrusiveness, low practicality (combined with high complexity), and questionable feasibility. Most modalities nowadays are still not technically plausible within commercial computer games. For instance, existing hardware for physiology requires the placement of body parts (e.g., head, chest, or fingertips) to the sensors, making physiological signals such as EEG, respiration, blood volume pulse, and skin conductance rather impractical and highly intrusive for most games. Future integrations of physiological sensors within game controllers—e.g., the upcoming Nintendo Wii[8] heart rate (vitality) sensor—and more research on wearable devices could lower the intrusiveness of biofeedback devices. Another point of concern for the use of physiology-based EDPCG is the effect of signal habitation—i.e., the level of physiological response decreases the more a specific stimuli is presented. Habitation is of particular relation to game-related research and connected to learnability in games. The design of a successful EDPCG approach should be able to provide dissimilar stimuli (via content generation) or control for habitation.

Pupillometry and gaze tracking are very sensitive to distance from screen and variations in light and screen luminance, which makes them rather impractical for use in a game application. Modalities such as facial expression and speech could be technically plausible in games even though the majority of the vision-based affect-detection systems currently available cannot operate in real-time [46]. Aside the real-time efficiency, the appropriateness of facial expression and speech for emotion recognition in games is questionable since most players tend to stay still and speechless while playing games [21]. At the positive end of the spectrum, Microsoft's XBox 360 Kinect[9] sensor device is pointing toward more natural game interaction and showcases a promising future of objective PEM.

3.2.1 Example: Affective Camera Control in 3D Prey/Predator Games

An example of model-free, objective PEM, for procedural content generation is the work of Yannakakis et al. [20] in which virtual camera profiles (game content) and physiology features are linked to expressed affective states such as challenge, frustration, and fun in 3D prey/predator games (see Fig. 4). The relationship between the player's heart rate, skin conductance, and blood volume pulse and game content are derived via both linear [52] and nonlinear [20] models.

8. http://wii.com/.
9. http://www.xbox.com/kinect/.

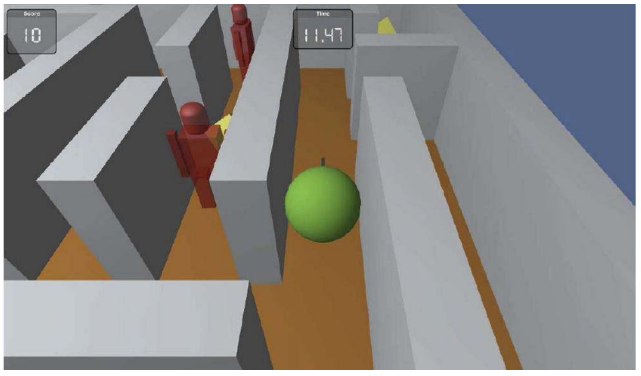


Fig. 4. The 3D prey/predator game *MazeBall* used for experiments in affective camera control.

3.3 Gameplay-Based PEM

The main assumption that drives *gameplay-based* PEM is that player actions and real-time preferences are linked to player experience since games may affect the player's cognitive processing patterns and cognitive focus. On the same basis, cognitive processes may influence emotions; one may infer the player's emotional state by analyzing patterns of the interaction and associating user emotions with context variables [30], [31]. Any element derived from the interaction between the player and the game forms the basis for gameplay-based PEM. This includes parameters from the player's behavior derived from responses to system elements (i.e., nonplayer characters, game levels, or embodied conversational agent behavior).

As in objective PEM, a gameplay-based PEM approach can be classified as *model-based*, *model-free*, or some hybrid between the two. Model-based approaches are typically inspired by a general theoretical framework of behavioral analysis and/or cognitive modeling (e.g., usability theory [53], belief-desire-intention model, the cognitive theory by Ortony, Clore, & Collins [54], Skinner's model [55], Scherer's theory [56]), but there are also theories about user affect that are specific to games, such as Malone's design components for fun games [57], Koster's *theory of fun* [58], and game-specific interpretations of Csikszentmihalyi's concept of *Flow* [59].

The inputs to a gameplay-based player experience model are statistical spatio-temporal features of game interaction. Those features are usually mapped to levels of cognitive states such as attention, challenge, and engagement [31]. General measures such as performance and time spent on a task have been used in the literature, but also game-specific measures such as the weapons selected in a shooter game [60], the times the player dies, and the unpredictability of deaths [34] in prey/predator games. Moreover, several dissimilar difficulty and challenge measures [61], [62], [63], [64], [65], [34], [66] have been proposed for different game genres. In all of these studies, difficulty adjustment is performed, based on a player experience model, that implies a direct link between challenge and fun.

Sometimes a player model [67], [68] is embedded in the process of PEM. Attempts to model and predict player actions and intentions [69], [70], [71] as well as to identify different playing patterns within a game [72], [73], [74], [75] belong to the model-free class of gameplay-based PEM.

Gameplay-based PEM is certainly the most computationally efficient and least intrusive PEM approach of all three, but it usually results in a low-resolution model of playing experience and its affective component. The models are often based on several strong assumptions that relate player experience to gameplay actions and preferences.

3.3.1 Example: Galactic Arms Race

Hastings et al. [60] developed a multiplayer game built on model-based, gameplay-based PEM for PCG. In the game, players guide a spaceship through various parts of space, engaging in firefights with enemies and collecting weapons (each weapon is optional, but having a good set of weapons is necessary for success). A key mechanism in the game is the generation of new weapons, based on which weapons are selected by players. Player preferences define the fitness value of the content. Thus, weapons players would select are directly linked to a high fitness value for the selected content and implicitly to a higher entertainment value for the player. Highly fit weapons are then recombined and the resulting generated weapons introduced directly into the game, making the content generation an online evolutionary process.

3.4 Hybrid PEM Approaches

The three PEM approaches can be combined to hybrid, and possibly more effective, solutions for capturing player experience. The combination between *subjective* and *objective* measures of player experience leads to the research areas of psychophysiology [76] in games [19], [32], [51], [77], [20] and affective gaming [4].

The combination between *subjective* and *gameplay-based* PEM results in self-report-driven cognitive modeling. Examples of this hybrid approach include the generation of predictors of reported affect grounded on in-game statistics and expressed affective state preferences of players [28], [38], [78].

Finally, the study of both *gameplay-based* and *objective* inputs for PEM has led to basic correlation analysis of the mapping between physiology and gameplay preferences ([79] among others) as well as to the investigation of the search space between affective states (derived from video and/or speech annotated data) and gameplay characteristics ([45] among others).

3.5 General Modeling Principles

A model of player experience predicts some aspect of the experience of a player in general, a type of player, or a particular player would have in some game situation. As already mentioned, there are many ways this can be done, with approaches to player experience modeling varying both regarding the inputs (from what the experience is predicted, e.g., physiology, level design parameters, playing style, or game speed), outputs (what sort of experience is predicted, e.g., fun, frustration, attention, or immersion), and the modeling methodology.

If data recorded includes a scalar representation of affect, or classes and annotated labels of affective states, using the PEM methods discussed above, any of a large number of machine learning (regression and classification) algorithms can be used to build affective models. Available methods include neural networks, Bayesian networks, decision trees, support vector machines, and standard linear regression.

On the other hand, if affect is given in a pairwise preference format (e.g., game version X is more frustrating than game version Y—this is often appropriate in subjective PEM), standard supervised learning techniques are inapplicable as the problem becomes one of *preference learning* [80], [81], [35]. In particular, neuro-evolutionary preference learning has proven suitable for this task; in this method, the weights of neural networks are evolved to minimize the error between reported and predicted preferences [82], [35]. Simpler methods such as linear discriminant analysis [18] have also proven to yield efficient affective predictors based on preferences.

4 Evaluating Game Content

In EDPCG, the main use of the acquired player models is to judge the quality (usefulness, fitness) of game content items. As mentioned above and discussed in more detail in the next section, assessing the quality of the content is necessary in the content generation phase, when candidate content items are evaluated and used to generate new content. However, just having a good model of some aspect of player experience does not necessarily allow us to directly judge the quality of particular items of game content, and the evaluation function might utilize the model in unexpected ways.

The task of the evaluation function is to evaluate an item of game content and assign it a scalar (or a vector of real numbers[10]) that accurately reflects its suitability for use in the game, and its capacity for instilling the desired affective state. Designing the evaluation function is ill-posed; the designer first needs to decide what, exactly, should be optimized and then how to formalize it. For example, one might intend to design an optimization algorithm that creates fun, immersive, frustrating, or exciting game content, and thus an evaluation function that reflects how much the particular piece of content contributes to the player's respective affective states while playing (recognized via PEM). Or, alternatively, one might want to consider immersion, frustration, anxiety, or other emotional response representations when designing such an evaluation function. The type and nature of playing experience is hand-crafted by the designer and dependent on the game under investigation and the optimization goals set. For instance, a designer might need to draw a mapping between player experience and acceleration of the rehabilitation process via a Wiihabilitation game [84] and design an evaluation function that encapsulates that. Or, alternatively, the designer might want to design an evaluation function that reflects to the successful training of social skills within a serious game [85].

Three key classes of evaluation functions can be distinguished for assessing the quality of generated content: *direct*, *simulation-based*, and *interactive* functions. The overview of the relationship between the three different PEM approaches and the dissimilar classes of evaluation functions can be found in Table 1. Each cell of Table 1 contains representative studies surveyed in this paper that correspond to the

10. In case of multidimensional evaluation functions, multi-objective or multicriteria optimization methodologies are employed [83].

TABLE 1
Overview Table for the Relationship between PEM Approaches and Types of Evaluation Functions

		S	O	G	SO	SG	OG	SOG
Direct	Theory-driven			[84], [66] ([23], [86])	([21], [77])			
	Data-driven			([73], [79], [74])	[20] ([18], [51], [14])	[38], [28]	([22])	[87], [45]
Simulation-based	Static	—	—	[88], [83], [89]([62])	—	[90]		
	Dynamic			[91]				
Interactive	Explicit	[92]	—	—			—	[85]
	Implicit	—		[60]	—	—		—

PEM includes the Subjective (S), Objective (O), and GamePlay-Based (G) approaches and their hybrids: subjective and objective (SO), objective and gameplay-based (OG), subjective and gameplay-based (SG), and all three combined (SOG). Representative studies surveyed in this paper that follow each approach appear in the corresponding table cell. References in parentheses denote that the game content is not considered explicitly as part of the evaluation; the interaction with nonplayer characters is considered instead. A dash (—) symbolizes an infeasible combination between a PEM approach and an evaluation function type.

respective combination of the PEM approach and the evaluation function type.

4.1 Direct Evaluation Functions

In a direct evaluation function, some features are extracted from the generated content, and these features are mapped directly to a content quality value. Hypothetically, such features might include the number of paths to the exit in a maze, the firing rate of a weapon, the spatial concentration of resources on a strategy map, and the material balance in randomly selected legal positions for board game rule set. The mapping between features and content quality might be linear or nonlinear, but typically does not involve large amounts of computation, and is typically specifically tailored to the particular game and content type. This mapping can be contingent on a model of the playing style, preferences, or affective state of the player yielding an element of *personalization* for content generation. An important distinction within direct evaluation functions is between *theory-driven* and *data-driven* functions. In theory-driven functions, the designer is guided by intuition and/or some qualitative theory of emotion or player experience to derive a mapping between an experience model and the quality of content. Examples of theory-driven direct evaluation functions can be found in the following studies: [23], [84], [86], [21], [77], [66]. On the other hand, data-driven functions are based on collecting data on the effect of various examples of content via, e.g., questionnaires and/or physiological measurements [20], and then using automated means to tune the mapping from content to player experience and finally to evaluation functions. More examples of data-driven direct evaluation functions can be found, among others, in [73], [79], [74], [20], [18], [51], [38], [28], [22], [87], [45].

As seen from Table 1, direct evaluation functions have not yet been utilized within the context of solely subjective or solely objective PEM. Gameplay-based PEM and the combination of subjective self-reports with other modalities of user input are the most popular PEM approaches for the design of direct evaluation functions.

4.1.1 Example

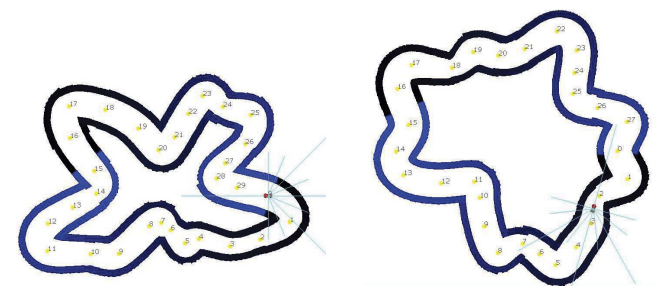
The automatic level generation for Super Mario Bros [28] that was discussed in Section 2.1 is a good example of a data-driven direct evaluation function which is based on a combination of subjective and gameplay-based PEM. As both design level and playing style are represented as vectors of real numbers, ordinary neural networks could be trained to map from the concatenation of a level design vector and a playing style vector to a predicted level of affect in each of the six affective dimensions included in the preference questionnaire.

4.2 Simulation-Based Evaluation Functions

It is not always apparent how to design a meaningful direct evaluation function for some game content—in some cases, it seems that the content must be sufficiently experienced and operated for particular emotional responses to be elicited and evaluated. A simulation-based evaluation function is based on an artificial agent playing through some part of the game that involves the content being evaluated. Such playthrough might include finding the way out of a maze while not being killed or playing the board game that results from the newly generated rule set against another artificial agent. Features that map to player experience models are then extracted from the observed gameplay (e.g., did the agent win? how fast? how was the variation in playing styles employed?) and used to calculate the quality value of the content. The artificial agent might be completely hand-coded or might be based on a learned behavioral model of human players.

A key distinction is between *static* and *dynamic* simulation-based functions. In a static evaluation function, it is not assumed that the agent changes while playing the game; in a dynamic evaluation function the agent changes during the game and the quality value somehow incorporates this change. For example, the implementation of the agent can be based on a learning algorithm and the evaluation function be dependent on *learnability*, i.e., how well and/or fast the agent learns to play the content that is being evaluated. Learning-based dynamic evaluation functions are especially appropriate when little can be assumed about the content and how to play it. Other uses for dynamic evaluation functions are to capture, e.g., order, effects and user fatigue.

It should be noted that while simulations of the game environment can typically be executed faster than real-time, simulation-based evaluation functions are, in general, more computationally expensive than direct evaluation functions; dynamic simulation-based evaluation functions can be time-consuming, all but ruling out online content generation. Moreover, the design of simulation-based evaluation



(a) (b)

Fig. 5. Two procedurally generated racing tracks, one evolved for a proficient player (a) and one for a not proficient one (b). Although this method is biased toward "flower-like tracks," it is clear that the first track is more difficult to drive, given its very narrow section and its sharp turns. A less proficient controller instead produced an easy track with gentle turns and no narrow sections.

functions is based on the assumption that an artificial agent plays the game similarly to how a human player would, functionally emulating the experiences of the human. This is a clear limitation of the approach that could be resolved, in part, by constructing agents that imitate human playing styles and backing up the evaluation function with user studies—e.g., as in [34].

It is obvious that simulation-based functions can only be coupled with gameplay-based player experience models; this is clearly reflected in Table 1. Studies in the literature have already been concerned with the design of both static [88], [83], [89] and dynamic [91], [90] simulation-based evaluation functions for content creation.

4.2.1 Example: Racing Game Tracks

Togelius et al. [88] designed a system for generation of tracks for a simple racing game (see Fig. 5). Tracks were represented directly as fixed-length parameter vectors, interpreted as b-splines (sequences of Bezier curves), which defined the course of the track. The player experience was modeled based on gameplay statistics (gameplay-based PEM); data had been collected as human players drove test tracks, and neural networks had been trained to drive the car similarly to how the human players drove. For the static simulation-based evaluation function, each candidate track was assessed by letting one of the neural network-based controllers drive on the track. The actual assessment of the quality of content was inspired by Malone's principles for engaging game design [57] and depended on the driving performance of the human-like neural network controller on the particular track: amount of progress, variation in progress, and difference between maximum and average speed.

4.2.2 Example: Predator/Prey Games

Togelius and Schmidhuber [91] conducted an experiment in which rulesets were evolved for grid-based games in which the player moves an agent around, in a manner similar to a discrete version of Pac-Man. Apart from the agent, the grid was populated by walls and "things" of different colors, which could be interpreted as items, allies, or enemies, depending on the rules. Rulesets were represented as fixed-length parameter vectors, interpreted as the effects on various things when they collided with each other or the agent, and their behavior. A relatively wide range of games could be represented using this vocabulary, and genotype generation was deterministic except for the starting position of things.

The dynamic simulation-based evaluation function that assessed the rule sets was based on gameplay-based PEM and inspired by Koster's *theory of fun* [58] in games and implemented as follows: An evolutionary reinforcement learning algorithm was used to learn each ruleset and the ruleset was scored dependent on how well it was learned. Games that were impossible or trivial were given low quality value, whereas those that could be learned after some time scored well.

4.3 Interactive Evaluation Functions

Interactive evaluation functions score content based on interaction with a player in the game, which means that fitness is evaluated during the actual gameplay. Data can be collected from the player either *explicitly*, using questionnaires or verbal input data, or *implicitly* by measuring, e.g., how often or long a player chooses to interact with a particular piece of content [60], when the player quits the game, or expressions of affect such as intensity of button-presses, shaking the controller, physiological response, gaze fixation, speech quality, facial expressions, and postures. Data are used to tailor the player experience models to the specific player, which in turn affects the evaluation function of the content presented to the player. If an interactive evaluation function is coupled with a subjective PEM component (e.g., self-reports shape the quality of content interactively), the function is classified as *explicit*; otherwise, the function is classified as *implicit* (see Table 1).

As mentioned earlier, the problem with explicit data collection is that it can interrupt the game play, whereas the problem with implicit data collection is that data may often be noisy, inaccurate, delayed, and of low-resolution.

4.3.1 Examples

Interactive evaluation functions have not been explored as much as the other two types of evaluation functions. The Galactic Arms Race game [60] discussed in Section 3.3.1 is the most prominent example of an implicit interactive evaluation function we are aware of; the utility of any particular weapon is directly proportional to how much it is used by the various players of the game. This example demonstrates that successful use of interactive evaluation is as much a question of game design as of computational intelligence.

A good example of an explicit interactive evaluation function can be found in Martin et al.'s system for interactively evolving building for the game *Subversion* (Introversion, In development) [92]. The work of Yannakakis et al. [85] contains elements of both explicit and implicit interactive evaluation functions for personalized quest generation in serious games. It should be pointed out that while there are, so far, rather few examples of interactive evaluation functions in EDPCG for games, there is much research on this topic in the neighboring field of evolutionary art [93].

5 Optimizing Game Content for Experience

Once a player experience model has been created based on acquired player data, and a content evaluation function has

been created based on the model, content can be optimized to maximize this evaluation function.

It is common to use some form of evolutionary algorithm (EA) as the main search mechanism. In an EA, a population of candidate content instances are held in memory. Each generation, these candidates are evaluated by the evaluation (fitness) function and ranked. The worst candidates are discarded and replaced with copies of the good candidates, except that the copies have been randomly modified (i.e., *mutated*) and/or recombined. However, EDPCG does not need to be married to evolutionary computation (EC); other search mechanisms are viable as well. The same considerations about representation and the search space largely apply, regardless of the approach to search.

5.1 Representing the Game Content

A central question in EDPCG concerns how to represent whatever is generated. Content may be represented symbolically within a tree or a graph data structure. That is usually the practice within interactive storytelling studies (see [94], [95] among others). While symbolic representation allows for content generation in a designer controlled-fashion, subsymbolic representations such as artificial genotypes allow for greater content variation and innovative content creation. Hybrid symbolic and subsymbolic approaches can also be very powerful alternatives. EDPCG primarily focuses on bottom-up, search-based [27] approaches for generating content which are driven by computational heuristics of player experience, but also allow for human (e.g., game designer) top-down intervention.

Viewing the generation of content as an artificial evolution process, an important question is how genotypes (i.e., the data structures that are internally represented by the content generator) are mapped to phenotypes (i.e., the data structure or process that is assessed by the evaluation function). An important distinction among representations is between *direct encodings*, wherein the size of the genotype is linearly proportional to the size of phenotype and each part of the genome maps to a specific part of the phenotype, and *indirect encodings*, wherein the genotype maps nonlinearly to the genotype and the former need not be proportional to the latter ([96], [97], [98]; see [99] for a review).

The study of representations in evolutionary computation is a broad field in its own right, where several concepts have originated that bear on PCG [100]. A particularly well-studied case is where candidates are represented as vectors of real numbers. These can more easily be analyzed, and standard algorithms can more easily be brought to work on such representations compared to more unusual representations. The problem representation should have the right dimensionality to allow for precise searching while avoiding the "curse of dimensionality" associated with representation vectors that are too large (or the algorithm should find the right dimensionality for the vector). Another principle is that the representation should have a high *locality*, meaning that a small change to the genotype should on average result in a small change to the phenotype and a small change to the utility value.

Apart from these concerns, of course, it is important that the chosen representation is capable of representing all the interesting solutions; this ideal can be a problem in practice for indirect encodings, for which there might be areas of phenotype space to which no genotypes map.

These considerations are important for EDPCG as the representation and search space must be well-matched to the domain if it is to perform optimally. There is a continuum between EDPCG that works with direct and indirect representation.

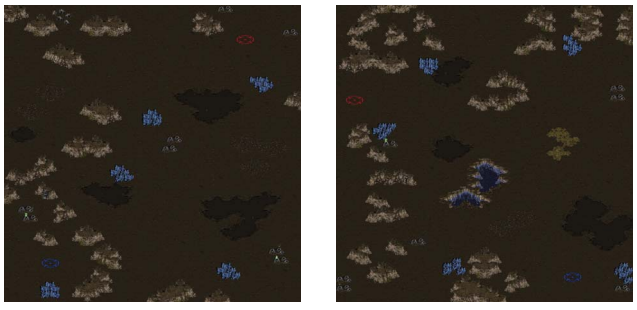
As a concrete example, a level for a 2D platform game (such as *Super Mario Bros* or *Sonic the Hedgehog*) might be represented:

1. *directly* as a 2D grid where the contents of each cell (e.g., ground, coin, wall, enemy, and free space) is specified separately, and mutation works by changing directly on the cells,
2. *more indirectly* as a list of positions and shapes of walls and pieces of ground that each occupy more than a single cell in the underlying grid, and another list of positions of enemies and items,
3. *even more indirectly* as a repository of different reusable patterns of walls and free space (e.g., a long jump followed by a particular type of enemy), and a list of how they are distributed across the level,
4. *very indirectly* as a list of desirable properties (e.g., number of gaps, distribution of gaps, number of enemies, average height of coins over ground), or
5. *most indirectly* as a random number seed.

These representations yield very different search spaces. In the first case, all parts of phenotype space are reachable, as the one-to-one mapping ensures that there is always a genotype for each phenotype. Locality is likely to be high because each mutation can only affect a single cell (e.g., turn it from wall into free space), which in most cases changes fitness only slightly. However, because the length of the genotype would be the number of cells in the grid, levels of any interesting size quickly encounter the curse of dimensionality. For example, a level based on a 100×100 grid (corresponding to a few screens in Super Mario Bros) would need to be encoded as a vector of length 10,000, which is more than many search algorithms can effectively approach.

At the other end of the spectrum, option number 5 does not suffer from search space dimensionality because it searches a 1D space. The question of whether all interesting points of phenotype space can be reached depends on the genotype-to-phenotype mapping, but it is possible to envision one where they can (e.g., iterating through all cells and deciding their content based on the next random number). However, the reason this representation is unsuitable for EDPCG is that there is no locality; one of the main features of a good random number generator is that there is no correlation between the numbers generated by different seed values. All search performs as badly (or as well) as random search.

Options 2 to 4 might all be suitable representations for searching for good platform levels. In options 2 and 3, the genotype length would grow with the desired phenotype (level) size, but sublinearly, so that reasonably large levels could be represented with tractably short genotypes. In option 4, genotype size is independent of phenotype size and can be made relatively small. On the other hand, the locality of these intermediate representations depends on



(a) Map 1 (b) Map 2

Fig. 6. Example maps evolved for the StarCraft game. (a) Map 1. (b) Map 2.

the care and domain knowledge with which each genotype-to-phenotype mapping is designed; both high- and low-locality mechanisms are conceivable.

5.1.1 Example: Strategy Game Maps

In [83] and [101], methods for generating playable and enjoyable strategy game maps using multi-objective evolution are introduced. Representations for two types of game terrains are investigated: a heightmap-based terrain for games for strategy games taking place in smooth landscapes where each location has an associated elevation, and a binary map representation where each map cell is either passable or impassable. The latter representation was designed so that maps could automatically be translated to the *StarCraft* map format, for use with the popular real-time strategy game (see Fig. 6).

The representation of terrain features was semidirect, akin to alternative 2 in the enumeration of approaches to representing levels above. The position and shape of terrain features such as mountains and rock formations were compactly encoded in the genotype, together with the locations of bases and resources. Candidate maps are evaluated through a set of direct and simple simulation-based evaluation functions; the multi-objective evolutionary algorithm then automatically identifies the trade-offs between these evaluation functions, allowing a human designer to choose among generated maps that are each Pareto-optimal in utility space.

The PCG described above can be linked to affective models of playing behavior via any of the three PEM approaches so that the multi-objective evolution of maps is guided by predicted affective states of individual players. While a particular map generates frustration for most players, it may very well elicit excitement for a few particular players. A successful affective model should be able to identify those across-subject differences and an efficient content generator should be able to accommodate the emotional response of all players.

5.1.2 Example: Game Rules

Game rules can also be seen as a form of game content, and represented in many different ways depending on the type of game and the associated generation mechanism. In Section 4.2.2, we discussed the evolution of rules in a quite restricted domain with a relatively direct representation, allowing fine-grained search in rule space.

As an example of a less restricted domain and somewhat less direct representation, Browne [90] developed a system for offline design of rules or two-player perfect-information board games using a form of genetic programming. Game rules were represented as expression trees, formulated in a custom-designed game description language. The fitness function was mostly simulation-based and derived from a combination of subjective and gameplay-based PEM: Standard game-tree search algorithms were used to play the generated game, and features such as material balance fluctuation and average playthrough length were extracted and fed to a model constructed from preference questionnaires.

5.2 Generating the Game Content

Once player experience is captured, content is appropriately represented, and content evaluation functions are designed, the content generator needs to search within the resulting search space for content that maximizes particular aspects of player experience.

If content is represented via a small number of dimensions (indirectly), exhaustive search should be able to provide robust solutions for online PCG [29]. In general, the more direct the representation becomes, the larger the content search space becomes. Where exhaustive search is infeasible, other techniques could be used, varying from simple heuristic and gradient-search (if gradient is computable) [37] to stochastic global optimization techniques such as evolutionary algorithms and particle swarm optimization.

Ideally, the content generator should be able to identify *if*, *how much*, and *how often* content should be generated for a particular player. There are players that dislike adaptation and emergence, and others that embrace it and instead loathe the idea of having to repeat any section of a game, raising questions about the significance and appropriateness of the affective loop within games. We believe that a successful EDPCG mechanism should be able to recognize if a player dislikes the notion of adaptation. This adds to and further emphasizes the importance of suitable methods for modeling the experience of the player during play.

Optimizing content creation during play could be viewed as a closed-loop control problem in which player experience defines the feedback for the controller. Closed-loop control is traditionally tied to certain limitations. The main concern for EDPCG is the lack of a priori knowledge of the effect of content generation. In other words, how would a mechanism be able to accurately assess the effect of a particular piece of content to player experience before it is generated in the game? Imitating and predicting player behavior could eliminate part of the problem.

It should be mentioned that there are many PCG techniques that are not search-based as the term is defined in [27]; these are variously classified as *constructive* or *generate-and-test*. Common techniques include L-systems [102], [103], which are used to generate trees and other vegetation in many games, and the diamond-square algorithm [104], which is commonly used to generate fractal landscapes. Other examples include the various dungeon generation algorithms used in rogue-like games (discussed in Section 1.2), which are rarely if ever published in academic venues.

A common feature for most constructive PCG techniques is the emphasis on randomness or, conversely, the lack of controllability. For example, it is hard to know anything at all about how a particular landscape generated by the diamond-square algorithm will look before generating it, and there is no way to tell the algorithm to, e.g., include two plateaus connected by a ridge. In contrast, the search-based approach allows the designer and/or player experience model to explicitly specify desirable properties of the content in gameplay terms; this is why we couple EDPCG with search-based algorithms. Note that many constructive PCG algorithms can be used as components in search-based algorithms, for example, L-systems can be used as genotype-to-phenotype mappings for landscape evolution [105]. Also note that there are attempts to make constructive PCG more controllable through declarative modeling [106].

6 Visions

As we have discussed in this paper, a number of successful experiments are already beginning to show the promise of experience-driven procedural content generation. By classifying these experiments according to the taxonomies presented in this paper, it can be seen both that 1) though all are examples of EDPCG, they differ from each other in several important dimensions, and 2) there is room for approaches other than those that have already been tried. There are several hard and interesting research challenges. These include the appropriate representation of game content and the design of relevant, reliable, and computationally efficient evaluation functions based on reliable computational models of player experience. The potential gains from providing good solutions to these challenges are significant: The invention of new game genres built on PCG, streamlining of the game development process, and further understanding of the mechanisms of human entertainment and player emotion are all possible.

The quantification of player experience and the assessment of content quality based on a computational model of player experience constitute one of the main challenges of EDPCG. While there are numerous different approaches to capturing user affect, there is no universally accepted approach for games and player experience. Games, being highly interactive and immersive environments, are capable of eliciting complex patterns of player affective states which have only been explored via small-scale experiments. Most possible combinations of evaluation function types and different PEM (subjective, objective, and gameplay-based) approaches for PCG have not been explored yet and there is much room for exploration of new combinations. Direct functions built solely on subjective or objective player experience models as well as interactive evaluation functions across all PEM approaches define some of the future research challenges of EDPCG. A combination of all three PEM approaches in a multimodal and unobtrusive fashion is most likely to provide the most reliable and accurate measures of player affective and cognitive responses.

The selection of a suitable emotional response representation within games and the EDPCG framework is also yet to be explored. While there is no globally accepted representation of emotional responses, discrete emotional states appear to be more relevant and appropriate for game design purposes than continuous multidimensional approaches (e.g., arousal, valance, and dominance). Nevertheless, the EDPCG framework is able to generate content for any emotional representation chosen as long as the representation is linked appropriately to the quality of the content.

An open research question concerns the optimization part of the EDPCG algorithm. As previously mentioned, generation of content online can be viewed as a closed-loop control problem that incorporates a noisy approximation of a feedback signal. Player-game interaction, which in this case forms the basis of the feedback signal, is stochastic by nature. Also, emotions as constructs have stochastic boundaries by nature which further augment the noise of the signal. Therefore, it could be interesting to apply other techniques from adaptive control (apart from global optimization) to this problem. In particular, stochastic control is suitable for problems with substantial noise and disturbances within the system. One could therefore use such techniques not only to redesign the content generation policy (controller) but also to tailor the player experience model (system model) per se during play.

As discussed in Section 5.1, the representation of content could be anything from bit strings and real-valued representations to trees and graphs. The same type of content can always be represented in different ways, having impact on the granularity, dimensionality, and locality the search space, and human-readability of the produced content items. Finding the most appropriate representation for different types of content and adaptation needs is a key research challenge. While, for instance, aspects of narrative have been represented as trees (see [95], [94], [86] among others) and real-value parameters have been used for platform game levels [28], other representations might be more suitable. The appropriate representation for game mechanics and game rules [91], [90], [89], [107], for instance, is still largely an open research question.

The need for automatic personalized content generation expands beyond games. The EDPCG approach is inspired by and built for games; its applicability to other HCI domains, however, is rather obvious. Recommender systems, web 2.0 applications, interface design, and computational creativity and art are some of the diverse HCI subdomains EDPCG is suitable for. We can imagine such "content" as personalized exercise plans, furniture assembly instructions, decorative elements (for use as Windows backgrounds or printed on 3D printers and placed on the window porch), schedules, menu systems, and shopping lists to be generated via nongame EDPCG.

In EDPCG, the user drives the generation of new (personalized) content; the designer's role becomes that of making high-level decisions about the type of content to be generated and the type of experience to be optimized, arguing moving the designer role up the value chain while saving labor extending the limits of what technology can do. Thus, EDPCG constitutes an innovative mixture of both user-driven (through PEM) and design-driven (through parameter design) content creation.

Acknowledgments

The authors thank all the participants in the discussions in the Procedural Content Generation Google Group, and the anonymous reviewers of this paper. The research was

supported, in part, by the FP7 ICT project SIREN (project no. 258453) and by the Danish Research Agency, Ministry of Science, Technology and Innovation project *AGameComIn*; project number: 274-09-0083.

REFERENCES

[1] J. Juul, *A Casual Revolution: Reinventing Video Games and Their Players.* MIT Press, 2009.

[2] T. Taylor, *Play Between Worlds: Exploring Online Game Culture.* MIT Press, Mar. 2006.

[3] C. Bateman and R. Boon, *21st Century Game Design*. Charles River Media, 2005.

[4] E. Hudlicka, "Affective Game Engines: Motivation and Requirements," *Proc. Fourth Int'l Conf. Foundations of Digital Games,* pp. 299-306, 2009.

[5] R.W. Picard, *Affective Computing.* MIT Press, 1997.

[6] I. Leite, A. Pereira, S. Mascarenhas, G. Castellano, C. Martinho, R. Prada, and A. Paiva, "Closing the Loop: from Affect Recognition to Empathic Interaction," *Proc. Third Int'l Workshop Affect Interaction in Natural Environments, ACM Multimedia '10.* 2010.

[7] P. Sundström, A. Ståhl, and K. Höök, "In Situ Informants Exploring an Emotional Mobile Messaging System in Their Everyday Practice," *Int'l J. Human-Computer Studies,* vol. 65, pp. 388-403, Apr. 2007.

[8] P. Sundström, "Exploring the Affective Loop," technical report, Stockholm Univ., 2005.

[9] R.W. Picard, E. Vyzas, and J. Healey, "Toward Machine Emotional Intelligence: Analysis of Affective Physiological State," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 23, no. 10, pp. 1175-1191, Oct. 2001.

[10] K.L. Walton, *Mimesis as Make-Believe.* Harvard Univ. Press, 1990.

[11] C. Bateman and L.E. Nacke, "The Neurobiology of Play," *Proc. Future Play '10,* pp. 1-8, 2010.

[12] R.A. Calvo and S.D. Mello, "Affect Detection: An Interdisciplinary Reveiw of Models, Methods and Their Applications," *IEEE Trans. Affective Computing,* vol. 1, no. 1, pp. 18-37, Jan.-June 2010.

[13] P. Rani, C. Liu, N. Sarkar, and E. Vanman, "An Empirical Study of Machine Learning Techniques for Affect Recognition in Human-robot Interaction," *Pattern Analysis and Applications,* vol. 9, no. 1, pp. 58-69, 2006.

[14] P. Rani, N. Sarkar, and C. Liu, "Maintaining Optimal Challenge in Computer Games through Real-Time Physiological Feedback," *Proc. 11th Int'l Conf. Human Computer Interaction,* 2005.

[15] G. Chanel, C. Rebetez, M. Betrancourt, and T. Pun, "Boredom, Engagement and Anxiety as Indicators for Adaptation to Difficulty in Games," *Proc. 12th Int'l Conf. Entertainment and Media in the Ubiquitous Era,* pp. 13-17, 2008.

[16] J. Kim and E. Andre, "Emotion Recognition Using Physiological and Speech Signal in Short-Term Observation," *Proc. Int'l Tutorial and Research Workshop Perception and Interactive Technologies,* pp. 53-64, 2006.

[17] R. Mandryk and K. Inkpen, "Physiological Indicators for the Evaluation of Co-Located Collaborative Play," *Proc. ACM Conf. Computer Supported Cooperative Work,* pp. 102-111, 2004.

[18] S. Tognetti, M. Garbarino, A. Bonarini, and M. Matteucci, "Modeling Enjoyment Preference from Physiological Responses in a Car Racing Game," *Proc. IEEE Conf. Computational Intelligence and Games,* pp. 321-328, Aug. 2010.

[19] G.N. Yannakakis, J. Hallam, and H.H. Lund, "Entertainment Capture through Heart Rate Activity in Physical Interactive Playgrounds," *User Modeling and User-Adapted Interaction,* special issue: affective mModeling and adaptation, vol. 18, nos. 1/2, pp. 207-243, Feb. 2008.

[20] G.N. Yannakakis, H.P. Martínez, and A. Jhala, "Towards Affective Camera Control in Games," *User Modeling and User-Adapted Interaction,* vol. 20, no. 4, pp. 313-340, 2010.

[21] A. Drachen, L. Nacke, G.N. Yannakakis, and A.L. Pedersen, "Correlation between Heart Rate, Electrodermal Activity and Player Experience in First-Person Shooter Games," *Proc. ACM SIGGRAPH '10,* 2010.

[22] S. McQuiggan, S. Lee, and J. Lester, "Predicting User Physiological Response for Interactive Environments: An Inductive Approach," *Proc. Second Artificial Intelligence for Interactive Digital Entertainment Conf.,* pp. 60-65, 2006.

[23] R. Aylett, J. Dias, and A. Paiva, "An Affectively Driven Planner for Synthetic Characters," *Proc. Int'l Conf. Automated Planning and Scheduling,* 2006.

[24] P.F. Camara, *Creativity and Artificial Intelligence: A Conceptual Blending Approach; Applications of Cognitive Linguistics.* Mouton de Gruyter, 2006.

[25] K. Stanley, B. Bryant, and R. Miikkulainen, "Real-Time Evolution in the NERO Video Game," *Proc. IEEE Symp. Computational Intelligence and Games,* pp. 182-189, Apr. 2005.

[26] D. Djordjevich, P. Xavier, M. Bernard, J. Whetzel, M. Glickman, and S. Verzi, "Preparing for the Aftermath: Using Emotional Agents in Game-Based Training for Disaster Response," *Proc. IEEE Symp. Computational Intelligence and Games,* pp. 266-275, Apr. 2008.

[27] J. Togelius, G.N. Yannakakis, K.O. Stanley, and C. Browne, "Search-Based Procedural Content Generation," *Proc. European Conf. Applications of Evolutionary Computation,* 2010.

[28] C. Pedersen, J. Togelius, and G.N. Yannakakis, "Modeling Player Experience for Content Creation," *IEEE Trans. Computational Intelligence and AI in Games,* vol. 2, no. 1, pp. 54-67, Sept. 2010.

[29] N. Shaker, G.N. Yannakakis, and J. Togelius, "Towards Automatic Personalized Content Generation for Platform Games," *Proc. Artificial Intelligence and Interactive Digital Entertainment,* pp. 63-68, Oct. 2010.

[30] J. Gratch and S. Marsella, "Evaluating a Computational Model of Emotion," *Autonomous Agents and Multi-Agent Systems,* vol. 11, no. 1, pp. 23-43, 2005.

[31] C. Conati, "Probabilistic Assessment of User's Emotions in Educational Games," *J. Applied Artificial Intelligence,* special issue on merging vognition and affect in HCI, vol. 16, pp. 555-575, 2002.

[32] R.L. Mandryk, K.M. Inkpen, and T.W. Calvert, "Using Psychophysiological Techniques to Measure User Experience with Entertainment Technologies," *Behaviour and Information Technology,* special issue on user experience, vol. 25, no. 2, pp. 141-158, 2006.

[33] R.J. Pagulayan, K. Keeker, D. Wixon, R.L. Romero, and T. Fuller, *User-Centered Design in Games: The HCI Handbook.* Lawrence Erlbaum Assoc., 2003.

[34] G.N. Yannakakis and J. Hallam, "Towards Optimizing Entertainment in Computer Games," *Applied Artificial Intelligence,* vol. 21, pp. 933-971, 2007.

[35] G.N. Yannakakis, "Preference Learning for Affective Modeling," *Proc. Int'l Conf. Affective Computing and Intelligent Interaction,* pp. 126-131, Sept. 2009.

[36] G.N. Yannakakis and J. Hallam, "Towards Capturing and Enhancing Entertainment in Computer Games," *Proc. Fourth Hellenic Conf. Artificial Intelligence,* pp. 432-442, May 2006.

[37] G.N. Yannakakis and J. Hallam, "Real-Time Game Adaptation for Optimizing Player Satisfaction," *IEEE Trans. Computational Intelligence and AI in Games,* vol. 1, no. 2, pp. 121-133, June 2009.

[38] C. Pedersen, J. Togelius, and G.N. Yannakakis, "Modeling Player Experience in Super Mario Bros," *Proc. IEEE Symp. Computational Intelligence and Games.* pp. 132-139, Sept. 2009.

[39] A. Nijholt, "BCI for Games: A State of the Art Survey," *Proc. Entertainment Computing,* pp. 225-228, 2009.

[40] T. Partala and V. Surakka, "Pupil Size Variation as an Indication of Affective Processing," *Int'l J. Human-Computer Studies,* vol. 59, nos. 1/2, pp. 185-198, 2003.

[41] A. Barreto, J. Zhai, and M. Adjouadi, "Non-Intrusive Physiological Monitoring for Automated Stress Detection in Human-Computer Interaction," *Proc. Human Computer Interaction,* pp. 29-39, 2007.

[42] S. Asteriadis, K. Karpouzis, and S.D. Kollias, "A Neuro-Fuzzy Approach to User Attention Recognition," *Proc. Int'l Conf. Artificial Neural Networks,* pp. 927-936, 2008.

[43] M. Pantic and G. Caridakis, "Image and Video Processing for Affective Applications" *Emotion-Oriented Systems: The Humaine Handbook,* pp. 101-117, Springer-Verlag, 2011.

[44] L. Kessous, G. Castellano, and G. Caridakis, "Multimodal Emotion Recognition in Speech-Based Interaction Using Facial Expression, Body Gesture and Acoustic Analysis," *J. Multimodal User Interfaces,* vol. 3, pp. 33-48, 2010.

[45] T. Kannetis, A. Potamianos, and G.N. Yannakakis, "Fantasy, Curiosity and Challenge as Adaptation Indicators in Multimodal Dialogue Systems for Preschoolers," *Proc. Workshop Child, Computer and Interaction, ICMI '09,* Nov. 2009.

[46] Z. Zeng, M. Pantic, G. Roisman, and T. Huang, "A Survey of Affect Recognition Methods: Audio, Visual, and Spontaneous Expressions," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 31, no. 1, pp. 39-58, Jan. 2009.
[47] N. Frijda, *The Emotions.* Cambridge Univ. Press, 1986.
[48] L. Feldman, "Valence Focus and Arousal Focus: Individual Differences in the Structure of Affective Experience," *J. Personality and Social Psychology,* vol. 69, pp. 53-166, 1995.
[49] J.A. Russell, "Core Affect and the Psychological Construction of Emotion," *Psychological Rev.,* vol. 110, pp. 145-172, 2003.
[50] S. Asteriadis, D. Soufleros, and K. Karpouzis, "A Natural Head Pose and Eye Gaze Dataset," *Proc. Int'l Conf. Multimodal Interfaces,* 2009.
[51] R.L. Mandryk and M.S. Atkins, "A Fuzzy Physiological Approach for Continuously Modeling Emotion during Interaction with Play Environments," *Int'l J. Human-Computer Studies,* vol. 65, pp. 329-347, 2007.
[52] H.P. Martinez, A. Jhala, and G.N. Yannakakis, "Analyzing the Impact of Camera Viewpoint on Player Psychophysiology," *Proc. Int'l Conf. Affective Computing and Intelligent Interaction.* pp. 394-399, Sept. 2009.
[53] K. Isbister and N. Schaffer, *Game Usability: Advancing the Player Experience.* Morgan Kaufman, 2008.
[54] A. Ortony, G.L. Clore, and A. Collins, *The Cognitive Structure of Emotions.* Cambridge Univ. Press, 1988.
[55] B.F. Skinner, *The Behavior of Organisms: An Experimental Analysis.* B.F. Skinner Foundation, 1938.
[56] K.R. Scherer, "Studying the Emotion-Antecedent Appraisal Process: An Expert System Approach," *Cognition and Emotion,* vol. 7, pp. 325-355, 1993.
[57] T.W. Malone, "What Makes Things Fun to Learn? Heuristics for Designing Instructional Computer Games," *Proc. Third ACM SIGSMALL Symp. and the First SIGPC Symp. Small Systems,* pp. 162-169, 1980.
[58] R. Koster, *A Theory of Fun for Game Design.* Paraglyph Press, 2005.
[59] M. Csikszentmihalyi, *Flow: The Psychology of Optimal Experience.* Harper Collins, 1990.
[60] E. Hastings, R. Guha, and K.O. Stanley, "Evolving Content in the Galactic Arms Race Video Game," *Proc. IEEE Symp. Computational Intelligence and Games,* pp. 241-248, 2009.
[61] H. Iida, N. Takeshita, and J. Yoshimura, "A Metric for Entertainment of Boardgames: Its Implication for Evolution of Chess Variants," *Proc. Int'l Wireless Comm. Expo,* pp. 65-72, 2003.
[62] J.K. Olesen, G.N. Yannakakis, and J. Hallam, "Real-Time Challenge Balance in an RTS Game Using rtNEAT," *Proc. IEEE Symp. Computational Intelligence and Games,* pp. 87-94, Dec. 2008.
[63] G. van Lankveld, P. Spronck, and M. Rauterberg, "Difficulty Scaling through Incongruity," *Proc. Fourth Int'l Artificial Intelligence and Interactive Digital Entertainment Conf.,* pp. 228-229, 2008.
[64] P. Spronck, I. Sprinkhuizen-Kuyper, and E. Postma, "Difficulty Scaling of Game AI," *Proc. Fifth Int'l Conf. Intelligent Games and Simulation,* pp. 33-37, 2004.
[65] G. Andrade, G. Ramalho, H. Santana, and V. Corruble, "Extending Reinforcement Learning to Provide Dynamic Game Balancing," *Proc. Workshop Reasoning, Representation, and Learning in Computer Games, 19th Int'l Joint Conf. Artificial Intelligence,* pp. 7-12, Aug. 2005.
[66] N. Sorenson and P. Pasquier, "Towards a Generic Framework for Automated Video Game Level Creation," *Proc. European Conf. Applications of Evolutionary Computation,* pp. 130-139, 2010.
[67] R. Houlette, "Player Modeling for Adaptive Games," *AI Game Programming Wisdom II,* pp. 557-566. Charles River Media Inc, 2004.
[68] D. Charles and M. Black, "Dynamic Player Modelling: A Framework for Player-Centric Digital Games," *Proc. Int'l Conf. Computer Games: Artificial Intelligence, Design, and Education,* pp. 29-35, 2004.
[69] G.N. Yannakakis and M. Maragoudakis, "Player Modeling Impact on Player's Entertainment in Computer Games," *Proc. 10th Int'l Conf. User Modeling,* pp. 74-78, July 2005.
[70] C. Thurau, C. Bauckhage, and G. Sagerer, "Learning Human-Like Movement Behavior for Computer Games," *From Animals to Animats 8: Proc. 8th Int'l Conf. Simulation of Adaptive Behavior,* pp. 315-323, July 2004.
[71] D. Thue, V. Bulitko, M. Spetch, and E. Wasylishen, "Interactive Storytelling: A Player Modelling Approach," *Proc. Third Conf. Artificial Intelligence and Interactive Digital Entertainment,* pp. 43-48, 2007.
[72] R. Thawonmas, M. Kurashige, K. Iizuka, and M. Kantardzic, "Clustering of Online Game Users Based on Their Trails Using Self-Organizing Map," *Proc. Entertainment Computing,* pp. 366-369, 2006.
[73] A. Drachen, A. Canossa, and G.N. Yannakakis, "Player Modeling Using Self-Organization in Tomb Raider: Underworld," *Proc. IEEE Symp. Computational Intelligence and Games,* pp. 1-8, Sept. 2009.
[74] O. Missura and T. Gärtner, "Player Modeling for Intelligent Difficulty Adjustment," *Proc. ECML-09 Workshop from Local Patterns to Global Models,* Sept. 2009.
[75] B. Weber and M. Mateas, "A Data Mining Approach to Strategy Prediction," *Proc. IEEE Symp. Computational Intelligence in Games,* pp. 140-147, Sept. 2009.
[76] S.H. Fairclough, "Fundamentals of Physiological Computing," *Interacting with Computers,* vol. 21, nos. 1/2, pp. 133-145, 2009.
[77] N. Ravaja, T. Saari, M. Turpeinen, J. Laarni, M. Salminen, and M. Kivikangas, "Spatial Presence and Emotions during Video Game Playing: Does It Matter with Whom You Play?" *Presence Teleoperators and Virtual Environments,* vol. 15, no. 4, pp. 381-392, 2006.
[78] M. Schwartz, H.P. Martinez, G.N. Yannakakis, and A. Jhala, "Investigating the Interplay between Camera Viewpoints, Game Information, and Challenge," *Proc. Artificial Intelligence and Interactive Digital Entertainment,* Oct. 2009.
[79] A. Drachen and A. Canossa, "Towards Gameplay Analysis via Gameplay Metrics," *Proc. 13th MindTrek,* Sept. 2009.
[80] J. Fürnkranz and E. Hüllermeier, "Preference Learning," *Künstliche Intelligenz,* vol. 19, no. 1, pp. 60-61, 2005.
[81] J. Doyle, "Prospects for Preferences," *Computational Intelligence,* vol. 20, no. 2, pp. 111-136, May 2004.
[82] G.N. Yannakakis, M. Maragoudakis, and J. Hallam, "Preference Learning for Cognitive Modeling: A Case Study on Entertainment Preferences," *IEEE Systems, Man, and Cybernetics; Part A: Systems and Humans,* vol. 39, no. 6, pp. 1165-1175, Nov. 2009.
[83] J. Togelius, M. Preuss, and G.N. Yannakakis, "Towards Multiobjective Procedural Map Generation," *Proc. Workshop Procedural Content Generation, Foundations of Digital Games,* June 2010.
[84] D. Dimovska, P. Jarnfelt, S. Selvig, and G.N. Yannakakis, "Towards Procedural Level Generation for Rehabilitation," *Proc. Workshop Procedural Content Generation, Foundations of Digital Games,* June 2010.
[85] G.N. Yannakakis, J. Togelius, R. Khaled, A. Jhala, K. Karpouzis, A. Paiva, and A. Vasalou, "Siren: Towards Adaptive Serious Games for Teaching Conflict Resolution," *Proc. Fourth European Conf. Games Based Learning,* 2010.
[86] Y. Cheong and M. Young, "A Computational Model of Narrative Generation for Suspense," *Proc. AAAI '06 Computational Aesthetic Workshop,* 2006.
[87] H.P. Martínez and G.N. Yannakakis, "Genetic Search Feature Selection for Affective Modeling: A Case Study on Reported Preferences," *Proc. Third Int'l Workshop Affective Interaction in Natural Environments,* pp. 15-20, 2010.
[88] J. Togelius, R. De Nardi, and S.M. Lucas, "Towards Automatic Personalised Content Creation in Racing Games," *Proc. IEEE Symp. Computational Intelligence and Games,* 2007.
[89] J. Marks and V. Hom, "Automatic Design of Balanced Board Games," *Proc. Artificial Intelligence and Interactive Digital Entertainment Int'l Conf.,* pp. 25-30, 2007.
[90] C. Browne, "Automatic Generation and Evaluation of Recombination Games," PhD dissertation, Queensland Univ. of Technology, 2008.
[91] J. Togelius and J. Schmidhuber, "An Experiment in Automatic Game Design," *Proc. IEEE Symp. Computational Intelligence and Games,* pp. 252-259, Dec. 2008.
[92] A. Martin, A. Lim, S. Colton, and C. Browne, "Evolving 3D Buildings for the Prototype Video Game Subversion," *Proc. EvoApplications,* 2010.
[93] H. Takagi, "Interactive Evolutionary Computation: Fusion of the Capacities of EC Optimization and Human Evaluation," *Proc. IEEE,* vol. 89, no. 9, pp. 1275-1296, 2001.
[94] M.O. Riedl and N. Sugandh, "Story Planning with Vignettes: Toward Overcoming the Content Production Bottleneck," *Proc. First Joint Int'l Conf. Interactive Digital Storytelling,* pp. 168-179, 2008.

[95] M.J. Nelson, C. Ashmore, and M. Mateas, "Authoring an Interactive Narrative with Declarative Optimization-Based Drama Management," *Proc. Artificial Intelligence and Interactive Digital Entertainment Int'l Conf.*, 2006.
[96] P.J. Bentley and S. Kumar, "The Ways to Grow Designs: A Comparison of Embryogenies for an Evolutionary Design Problem," *Proc. Genetic and Evolutionary Computation Conf.*, pp. 35-43, 1999.
[97] G.S. Hornby and J.B. Pollack, "The Advantages of Generative Grammatical Encodings for Physical Design," *Proc. IEEE Congress on Evolutionary Computation*, 2001.
[98] K.O. Stanley, "Compositional Pattern Producing Networks: A Novel Abstraction of Development," *Genetic Programming and Evolvable Machines*, special issue on developmental systems, vol. 8, no. 2, pp. 131-162, 2007.
[99] K.O. Stanley and R. Miikkulainen, "A Taxonomy for Artificial Embryogeny," *Artificial Life*, vol. 9, no. 2, pp. 93-130, 2003.
[100] F. Rothlauf, *Representations for Genetic and Evolutionary Algorithms.* Springer, 2006.
[101] J. Togelius, M. Preuss, N. Beume, S. Wessing, J. Hagelbäck, and G.N. Yannakakis, "Multiobjective Exploration of the Starcraft Map Space," *Proc. IEEE Conf. Computational Intelligence and Games*, pp. 265-272, Aug. 2010.
[102] A. Lindenmayer, "Mathematical Models for Cellular Interaction in Development Parts I and II," *J. Theoretical Biology*, vol. 18, pp. 280-299 and 300-315, 1968.
[103] P. Prusinkiewicz, "Graphical Applications of L-Systems," *Proc. Graphics Interface/Vision Interface*, pp. 247-253, 1986.
[104] G.S.P. Miller, "The Definition and Rendering of Terrain Maps," *Proc. ACM SIGGRAPH*, vol. 20, 1986.
[105] D. Ashlock, "Automatic Generation of Game Elements Via Evolution," *Proc. IEEE Conf. Computational Intelligence and Games*, 2010.
[106] R.M. Smelik, T. Tutenel, K.J. de Kraker, and R. Bidarra, "Integrating Procedural Generation and Manual Editing of Virtual Worlds," *Proc. ACM Foundations of Digital Games*, June 2010.
[107] A.M. Smith and M. Mateas, "Variations Forever: Flexibly Generating Rulesets from a Sculptable Design Space of Mini-Games," *Proc. IEEE Conf. Computational Intelligence and Games*, pp. 273-280, Aug. 2010.

Georgios N. Yannakakis received both the five year Diploma (1999) in production engineering and management and the MSc (2001) degree in financial engineering from the Technical University of Crete and the PhD degree in informatics from the University of Edinburgh in 2005. He is currently working as a associate professor at the IT University of Copenhagen (ITU). Prior to joining the Center for Computer Games Research, ITU, in 2007, he was a postdoctoral researcher at the Mærsk McKinney Møller Institute, University of Southern Denmark. His research interests include user modeling, neuro-evolution, computational intelligence in computer games, cognitive modeling and affective computing, emergent cooperation, and artificial life. He has published around 60 journal and international conference papers in the aforementioned fields. He is an associate editor of the *IEEE Transactions on Affective Computing* and the *IEEE Transactions on Computational Intelligence and AI in Games*, and the chair of the IEEE Computational Intelligence Society Task Force on Player Satisfaction Modeling. He is a member of the IEEE.

Julian Togelius received the BA degree in philosophy from Lund University in 2002, the MSc degree in evolutionary and adaptive systems from the University of Sussex in 2003, and the PhD degree in computer science from the University of Essex in 2007. He is currently working as a assistant professor at the IT University of Copenhagen (ITU). Before joining ITU in 2009, he was a postdoctoral researcher at IDSIA in Lugano, Switzerland. His research interests include applications of computational intelligence in games, procedural content generation, automatic game design, evolutionary computation, and reinforcement learning; he has published around 50 papers in journals and conferences about these and related topics. He is an associate editor of *IEEE Transactions on Computations Intelligence and AI in Games* and the current chair of the IEEE Computational Intelligence Society's Technical Committee on Games. He is a member of the IEEE.