# Chapter 10
# The experience-driven perspective

Noor Shaker, Julian Togelius, and Georgios N. Yannakakis

**Abstract** Ultimately, content is generated for the player. But so far, our algorithms have not taken specific players into account. Creating computational models of a player's behaviour, preferences, or skills is called player modelling. With a model of the player, we can create algorithms that create content specifically tailored to that player. The experience-driven perspective on procedural content generation provides a framework for content generation based on player modelling; one of the most important ways of doing this is to use a player model in the evaluation function for search-based PCG. This chapter discusses different ways of collecting and encoding data about the player, primarily player experience, and ways of modelling this data. It also gives examples of different ways in which such models can be used.

## 10.1 Nice to get to know you

As you play a game, you get to know it better and better. You understand how to use its core mechanics and how to combine them; you get to know the levels of the game, or, if the levels are procedurally generated, the components of the levels and typical ways in which they can be combined. You learn to predict the behaviour of other creatures, characters and systems in the game. All this you learn from your interaction from the game. While playing, you also adapt to the game: you change your behaviour so as to achieve more success in the game, or so as to entertain yourself better.

However, both you and your game take part in this interaction, and all of your interaction data is available to the game as well. In principle, the game should be able to get to know you as much as you get to know it. After all, it has seen you succeed at overtaking that other car, fail that sequence of long jumps, give up and shut down the game after crashing your plane for the seventh time or finally resort to buying extra moves after almost clearing a particular puzzle. A truly intelligent game should know how you play better than you know it yourself. And then, it should be able to adapt itself so as to entertain you better, or let you achieve more or less success in the game, or perhaps to give you some other kind of experience you would not otherwise have had.

The idea of *game adaptation*, the game adapting itself in response to how you play (or some other information it might have about you), is an old one. In its simplest form it is called "dynamic difficulty adjustment" (DDA), and simply means that the difficulty of the game is increased if the player does well and decreased if the player plays poorly. This can be seen in many car racing games, where the opponent cars always seem to be just ahead of you or just behind you, regardless of how well you play (also known as "rubber banding"). The game design rationale for rubber banding is that if the player is much in front of the opponents s/he will not perceive a challenge, and if the player is far behind the opponents s/he will lose hope of ever catching up; in either case, the player will likely lose interest in the game. This is sometimes rationalised as a way of keeping the player in the "flow channel". *Flow* is a concept which was invented by the psychologist Csikszentmihalyi to signify the "optimal experience", where someone is completely absorbed in the activity they are performing; one condition for this is constant but not unassailable challenge [5]. The flow concept has inspired several theories of challenge and engagement in games, such as GameFlow [27]; it is, however, limited to challenge, which is only one dimension of player experience [3].

DDA mechanisms in racing games are often implemented simply by letting the opponent cars drive faster or slower. There are interesting exceptions, such as the *Mario Kart* series, which gives more powerful power-ups to players who lag behind, some of which allow them to attack players who lead the pack. Other games might lower the difficulty of a particular section of the game after a player has failed numerous times; *Grand Theft Auto V* allows the player to simply skip any action sequence which the player has failed three times already. There are several proposals for how this could be done more automatically, using AI techniques [10]. A key realisation is that adaptation is about more than just difficulty: to begin with, difficulty is multi-dimensional, as a game can be difficult in many different ways, and people have unbalanced skill sets. The same game could be difficult for player A because of its requirement for quick reactions, for player B because of the spatial navigation, and for player C because of the nuances of the story that needs to be understood in order to solve its puzzles. Also, just having the right difficulty is in general not enough for a game to be perfectly tailored for a particular player. Different players might prefer different balances of game elements or atmospheres, such as scary, intense or contemplative parts of the game. Adaptation could in principle happen along many axes, which may not be formalised or even described. There are also many possible methods for adaptation, some of which involve modifying the content of the game or even generating new content.

In this chapter, we will focus on the use of PCG methods to adapt games to the experience of the player, which is called *experience-driven procedural content generation* [37]. Experience-driven PCG views game content as the building block of player experience which is, in turn, synthesised via content adaptation. In experience-driven PCG, a model of player experience is learned that can predict some aspect of the player's experience (e.g. challenge, frustration, engagement, spatial involvement) based on some aspect of game content. This model can then be used as a base for an evaluation function in search-based or mixed-initiative PCG. For example, a model might be learned that predicts how engaging some players think individual building puzzles are in a physics-based puzzle game. This model can then be used for evolving new puzzles, where the evaluation function rewards such puzzles that are predicted to be most engaging for the target player(s).

The chapter is structured as follows. First we describe the various ways we can elicit player experience through a game and collect information about player experience. The next section discusses algorithms for creating models of player experience, such as neuroevolutionary preference learning, based on data collected during the game interaction (model's input) and annotated player experience (model's output). A short section discusses how these models can be used in content generation, followed by a prolonged example describing experience-driven level generation in *Super Mario Bros.* in detail.

## 10.2 Eliciting player experience

Games can elicit rich and complex patterns of user experience as they combine unique properties such as rich interactivity and potential for multifaceted player immersion [3]. User experience in games can be elicited primarily through long- or short-term interaction with core game elements. Arguably social interaction may have a clear impact on a player's experience; however, it offers a rather challenging problem for artificial intelligence, signal processing and experience-driven PCG techniques. While an interesting direction for further research, social interaction is not included in the set of player experience elicitors considered in this chapter.

Experience-driven PCG views game content as potential *building blocks of player experience* [37]. That is precisely the fundamental link between game content and player experience. In that regard, all potential content types can elicit player experience. Game content here refers to the game environment, and its impact on player experience can be directly linked to *spatial involvement* and *affective involvement* [3]. But it also includes, as throughout the book, fundamental game-design building blocks such as game mechanics, narrative and reward systems, as well as various other game aspects such as audiovisual settings and camera profiles and effects. In addition complex, social and emotional non-player characters can be used as triggers of desired player experience. In order for agents to elicit meaningful experience and immerse the player they need to engage players in rich and emotional interaction. Towards that purpose they may embed computational models of cognition, behaviour and emotion which are based upon theoretical models such as the OCC [20].

## 10.3 Modelling player experience

The detection and computational modelling of a user's affective state are core problems in user experience and affective computing research. Detecting and modelling affective states in games can be seen as a special case of this, though in an unusually complex domain. Given the complexity and richness of game-player interaction and the multifaceted nature of player experience, methods that manage to overcome the above challenges and model player experience successfully advance our understanding of human behaviour and emotive reaction with human computer interaction. Player experience modelling (PEM) can thus be viewed as a form of user modelling within games incorporating aspects of behaviour, cognition and affect. PEM involves all three key phases for computational model construction. These are signal processing, feature extraction and feature selection for the model's input; experience annotation for the model's output; and various machine learning and computational intelligence techniques that learn the mapping between the two. Within experience-driven PCG, game content is also represented in the underlying function that characterises player experience.

We can distinguish between *model-based* and *model-free* approaches to player experience modelling [37] as well as potential hybrids between them. The difference is whether the computational model is based on or structured by a theoretical framework. A completely model-based approach relies solely on a theoretical framework that maps game context and player responses to experience. In contrast, a completely model-free approach assumes there is an unknown function between modalities of user input, game content and experience that may be discovered by a machine-learning algorithm (or a statistical model) that does not assume anything about the structure of this function. The space between a completely model-based and a completely model-free approach can be viewed as a continuum along which any PEM approach might be placed. The rest of this section presents the key elements of both model-based and model-free approaches and discusses the core components of a learned computational model (i.e. model input, model output and common modelling methods).

### *10.3.1 Model input and feature extraction*

The PEM's input can be of three main types: a) player behavioural responses to game content as gathered from **gameplay** data (i.e. behavioural data); b) **objective** data collected as player experience manifestations to game content stimuli such as physiology and body movements; and c) the **game context** which comprises any type of game content viewed, played through, and/or created [37, 35, 36].

Given the multifaceted nature of player experience, the input of a PEM usually consists of complex spatio-temporal patterns found in user inputs, sometimes sampled from multiple modalities. These signals need to be processed and relevant data features need to be extracted to feed the model. Relevant features, however, are hard to find within such signals and the ad-hoc design of statistical features often undermines the performance of PEM. There are several available methods within feature extraction (such as principal component analysis and Fischer projection) and feature selection (such as sequential forward selection and genetic-search-based selection) that are applicable to the problem. Recently techniques such as *sequence mining* [15] for feature extraction and *deep learning* [13] for feature combination have shown potential to construct meaningful features for PEM. These methods have been able to fuse data from multiple sources across several player inputs and between player input and game content. In particular, deep learning offers powerful pattern recognition capacities which can detect the most distinct patterns across multiple signals, and provides complex spatio-temporal data attributes that complement standard ad-hoc feature extraction [13]. Sequence mining, on the other hand, identifies the most frequent sequences of events across user input modalities and game context which could be relevant as features for any PEM attempt [15].

In the rest of this section, we will look in more detail at these three types of input to PEM: gameplay input, objective input, and game context input.

#### 10.3.1.1 Gameplay input

The key motivation behind the use of behavioural (gameplay-based) player input is that player actions and real-time preferences are linked to player experience as games affect the player's cognitive processing patterns, cognitive focus and emotional state. Essentially, you express the contents of your mind through gameplay. Arguably it is possible to infer a player's current experience state by analysing patterns of the interaction and associating player experience with game context variables [4, 8]. The models built on this user input type rely on detailed attributes from the player's behaviour which are extracted from player behavioural responses during the interaction with game content stimuli. Such attributes, also named *game metrics*, are statistical spatio-temporal features of game interaction [6] which are usually mapped to levels of cognitive states such as attention, challenge and engagement [23]. In general, both generic measures—such as the level of player performance and the time spent on a task—as well as game-specific measures—such as the items picked and used—are relevant for the gameplay-based PEM.

#### 10.3.1.2 Objective input

The variety of available content types within a game can act as elicitors for complex and multifaceted player experience patterns. Such patterns of experience may, in turn, cause changes in the player's physiology, be reflected in the player's facial expression, posture and speech, and alter the player's attention and focus level. Monitoring such bodily alterations can assist in recognising and synthesising predictors of player experience. The objective approach to PEM assumes access to multiple modalities of player input which manifest aspects of player experience.

Thus, the impact of game content on a number of real-time recordings of the player may be investigated. Physiology offers the primary medium for detecting a player's experience via objective measures [33]: signals obtained from electrocardiography (ECG) [34], photoplethysmography [34, 28], galvanic skin response (GSR) [9], respiration [28], electroencephalography (EEG) [18] and electromyography (among others) are commonly used for the detection of player experience given the recent advancements in sensor technology and physiology-based game interfacing [33]. In addition to physiology the player's bodily expressions may be tracked at different levels of detail and real-time cognitive or affective responses to game content may be inferred. The core assumption of such input modalities is that particular bodily expressions are linked to basic emotions and cognitive processes [2]. Motion tracking may include body posture [22], facial expression and head pose [23].

Beyond the non-verbal cues discussed above there is also room for verbal cue investigation within games. In general, social signals derived from human verbal communication can potentially be used within social games that allow player-to-player interaction (direct or indirect). Such signals challenge the principles of individual player experience modelling but are expected to open the horizon and augment the potential of the experience-driven PCG framework.

#### 10.3.1.3 Game context input

In addition to gameplay and objective data, the context of the game—e.g. the game content experienced, played, or created—is a necessary input for PEM. Game context is the real-time parameterised state of the game which could extend beyond the game content. Without the game context input, player experience models run the risk of inferring erroneous player experience states. For example, an increase in galvanic skin response (GSR) can be linked to a set of dissimilar high-arousal affective states such as frustration and excitement. Thus, the cause of GSR increase (e.g. due to a player's death in a gap between platforms, or alternatively, due to a game level completion) needs to be fused within the GSR signal and embedded in the model. Context-free modelling (while important and desired) has not been investigated to the degree that we can identify generic and context-independent content patterns, features and attributes across games and players. A few recent studies, however, such as that of Martinez et al. [14], attempt to investigate context-independent physiological features that can capture player experience across multiple game genres.

### *10.3.2 Model output: Experience annotation*

The output of a player experience model is provided through an experience annotation process which can either be based on first-person reports (self-reports) or on reports expressed indirectly by experts or external observers [37]. The model's output is, therefore, linked to a fundamental research question within player experience and affective computing: what is the ground truth of player experience and how to annotate it? To address this question a number of approaches have been proposed. The most direct way to annotate player experience is to ask the players themselves about their experience, and build a model based on these annotations. Subjective annotation can be based on either players' free response during play or on forced data retrieved through questionnaires. Alternatively, experts or external observers may annotate the playing experience in a similar fashion. Third-person player experience annotation entails the identification of particular user (cognitive, affective, behavioural) states by user experience and game design experts.

Annotations (either forced self-reports or third-person) can be classified as *rating* (scalar), *class*, or *preference* (ranking) data. With ratings, annotators are asked to answer questionnaire items given in a rating/scaling form—such as the Game Experience Questionnaire [11] or the Geneva Emotion Wheel [1]—which labels user states with a scalar value (or a vector of values). In a class-based format, subjects are asked to pick a user state from a particular representation which is usually a simple boolean question (Was that game level frustrating or not? Is this a sad facial expression?). In the preference annotation format [29], annotators are asked to compare a playing experience in two or more variants/sessions of the game (Was that level more engaging that this level? Which facial expression looks happier?). Recent comparative studies have argued that rating approaches have disadvantages compared to ranking questionnaire schemes [32, 16], such as increased order-of-play and inconsistency effects [30] and lower inter-rater agreement [17, 31].

### 10.3.3 Modelling approaches

The approach used to construct models of player experience heavily relies on the modelling approach followed (model-based vs. model-free) and the annotation scheme adopted. With the model-based approach, components of the model and any parameters that describe them are constructed in an ad-hoc manner and, sometimes, tested for validity on a trial-and-error basis. No machine learning or sophisticated computational tools are required for these approaches. One could envisage optimising the parameter space to yield more accurate models; that, however, would require empirical studies that bring the approach closer to a model-free perspective.

Model-free approaches, on the other hand, are dependent on the annotation scheme and, in turn, the type of model output available. If data recorded includes either a scalar representation (e.g. via ratings) or classes of annotated labels of user states any of a large number of machine learning (regression and classification) algorithms can be used to build affective models. Available methods include artificial neural networks, Bayesian networks, decision trees, support vector machines and standard linear regression. Alternatively, if experience is annotated in a ranked format, standard supervised-learning techniques are inapplicable, as the problem becomes one of preference learning [7]. Neuro-evolutionary preference learning [29] and rank-based support vector machines [12], along with simpler methods such as

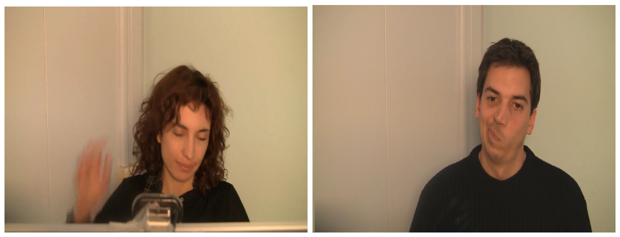

Fig. 10.1: Player responses to losing in IMB. Adapted from [23]

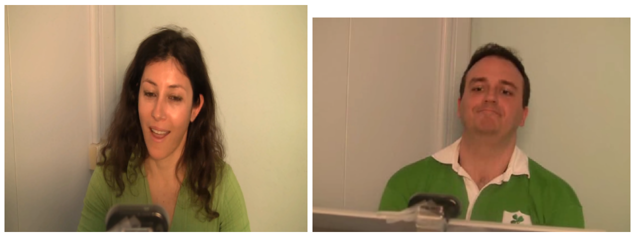

Fig. 10.2: Player responses to winning in IMB. Adapted from [23]

linear discriminant analysis [28], are some of the available approaches for learning preferences.

The ultimate goal of constructing models of player experience is to use these models as measures of content quality and, consequently, to produce affective, cognitive, and behavioural interaction in games and generate personalised or player-adapted content. Quantitative models of player experience can be used to capture player-game interaction and the impact of game content on player experience.

## 10.4 Example: *Super Mario Bros.*

The work of Shaker et al. [25, 23, 24] on modelling and personalising player experience in *Infinite Mario Bros.* (IMB) [21]—a public-domain clone of *Super Mario Bros.* [19]—gives a complete example of applying the experience-driven PCG approach. First, they build models of player experience based on information collected from the interaction between the player and the game. Different types of features capturing different aspects of player behaviour are considered: *subjective* self-reports of player experience; *objective* measures of player experience collected by extracting information about head movements from video-recorded gameplay sessions; and *gameplay* features collected by logging players' actions in the game. Figures 10.1, 10.2, and 10.3 show examples of objective video data correlated with in-game events: players' reactions when losing, winning, and encountering hard situations, respectively.

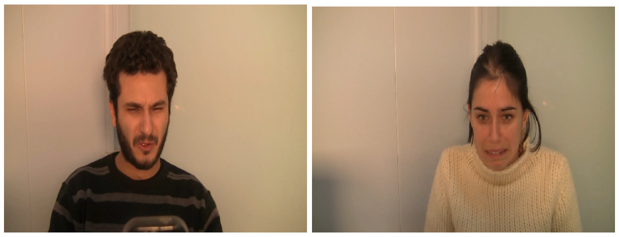

Fig. 10.3: Player responses to hard situations in IMB. Adapted from [23]

Table 10.1: The different types of representations of content and gameplay features in [25]

| Feature | Description |
|---|---|
| 🟫 | Flat platform |
| (🪙)(🪙, 🪙) | A sequence of three coins |
| $(R^{\blacktriangleright}, R^{\blacktriangleright\Uparrow})$(🐢) | Moving then jumping in the right direction when encountering an enemy |
| (🟫, 🟫)(🟫) | A gap followed by a decrease in platform height |
| $(\Uparrow^{\blacktriangleright})(S)(\blacktriangleright)$ | Jumping to the right followed by standing still then moving right |
| $t_{right}$ | Time spent moving right |
| $n_{jump}$ | Total number of jumps |
| $n_{coin}$ | Total number of coins |
| $k_{stomp}$ | Number of enemies killed by stomping |
| $N_e$ | Total number of enemies |
| $B$ | Total number of blocks |

The choice of feature representation is vitally important since it allows different dimensions of player experience to be captured. Furthermore, the choice of content representation defines the search space that can be explored and affects the efficiency of the content-creation method. To accommodate this, the different sets of features collected are represented as frequencies describing the number of occurrences of various events or the accumulated time spent doing a certain activity (such as the number of killings of a certain type of enemies or the total amount of time spent jumping). Features are also represented as sequences capturing the spatial and temporal order of events and allowing the discovery of temporal patterns [25]. Table 10.1 presents example features from each representation.

Based on the features collected, a modelling approach is followed in an attempt to approximate the unknown function between game content, players' behaviour and how players experience the game. The player experience models are developed on different types and representations of features allowing a thorough analysis of the player–content relationship.

The following sections describe the approach followed to model player experience and the methodology proposed to tailor content generation for particular players, using the constructed models as measures of content quality.

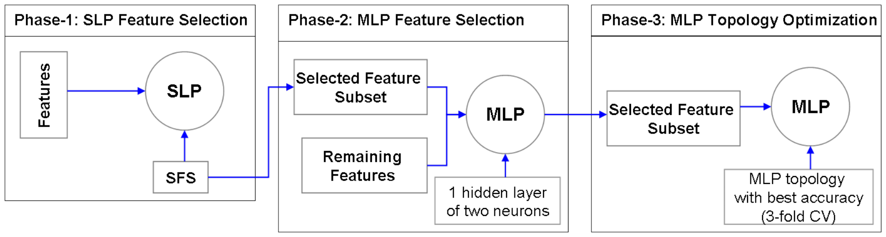
Phase-1: SLP Feature Selection
Features
SLP
SFS
Phase-2: MLP Feature Selection
Selected Feature Subset
Remaining Features
MLP
1 hidden layer of two neurons
Phase-3: MLP Topology Optimization
Selected Feature Subset
MLP
MLP topology with best accuracy (3-fold CV)

Fig. 10.4: The three-phase player experience modelling approach of [25]

### *10.4.1 Player experience modelling*

When constructing player experience models, the place to start is identifying relevant features of game content and player behaviour that affect player experience. This can be done by recording gameplay sessions and extracting features as indicators of players' affect, performance, and playing characteristics. Given the large size of the feature set that could be extracted, feature selection then becomes a critical step.

In this example, the input space consists of the features extracted from gameplay sessions. Feature selection is done by using sequential forward selection (SFS), a particular feature-selection approach (of many). Candidate features are evaluated by having neuroevolutionary preference learning train simple single-layer perceptrons (SLPs) and multi-layer perceptrons (MLPs) to predict emotional states, and choosing the features that best predict the states [25]. This yields a different subset of features for predicting each reported emotional state.

The underlying function between gameplay, content features, and reported player experience is complex and cannot be easily captured using the simple neuroevolution model used in the feature-selection step. Therefore, once all features that contribute to accurate simple neural network models are found, an optimisation step is run to build larger networks with more complex structures. This is carried out by gradually increasing the complexity of the networks by adding hidden nodes and layers while monitoring the models' performance. Figure 10.4 presents an overview of the process.

Following this approach, models with high accuracies were constructed for predicting players' reports of engagement, frustration and challenge from different subsets of features from different modalities. The models constructed were also of varying topologies and prediction accuracies.

### *10.4.2 Grammar-based personalised level generator*

In Chapter 5, we described how grammatical evolution (GE) can be used to evolve content for IMB. GE employs a design grammar to specify the structure of possible level designs. The grammar is used by GE to transform the phenotype into a level structure by specifying the types and properties of the different game elements that will be presented in the final level design. The fitness function used in that chapter scored designs based on the number of elements presented and their placement properties.

It is possible to use player experience measurements as a component of the fitness function for grammatical evolution as well. This allows us to evolve personalised content. The content is ranked according to the experience it evokes for a specific player and the content generator searches the resulting space for content that maximises particular aspects of player experience. The fitness value assigned for each individual in the population (a level design) in the evolutionary process is the output of the player experience model, which is the predicted value of an emotional state. The PEM's output is calculated by computing the values of the model's inputs; this includes the values of the content features which are directly calculated for each level design generated by GE and the values of the gameplay features estimated from the player's behavioural style while playing a test level.

The search for the best content features that optimise a particular state is guided by the model's prediction of the player experience states, with higher fitness given to individuals that are predicted to be more engaging, frustrating, or challenging for a particular player.

#### 10.4.2.1 Online personalised content generation

Personalisation can be done online. While the level is being played, the playing style is recorded and then used by GE to evaluate each individual design generated. Each individual is given a fitness according to the recorded player behaviour and the values of its content features. The best individual found by GE is then visualised for the player to play.

It is assumed that the player's playing style is largely maintained during consecutive game sessions and thus his playing characteristics in a previous level provide a reliable estimator of his gameplay behaviour in the next level. To compensate for the effect of learning while playing a series of levels, the adaptation mechanism only considers the recent playing style, i.e. the one which the player exhibited in the most recent level. Thus, in order to effectively study the behaviour of the adaptation mechanism, it is important to monitor this behaviour over time. For this purpose, AI agents with varying playing characteristics have been employed to test the adaptation mechanism since this requires the player to playtest a large number of levels. Figure 10.5 presents the best levels evolved to optimise player experience of challenge for two AI agents with different playing styles. The levels clearly exhibit different structures; a slightly more challenging level was evolved for the second agent, with more gaps and enemies than the one generated for the first agent.

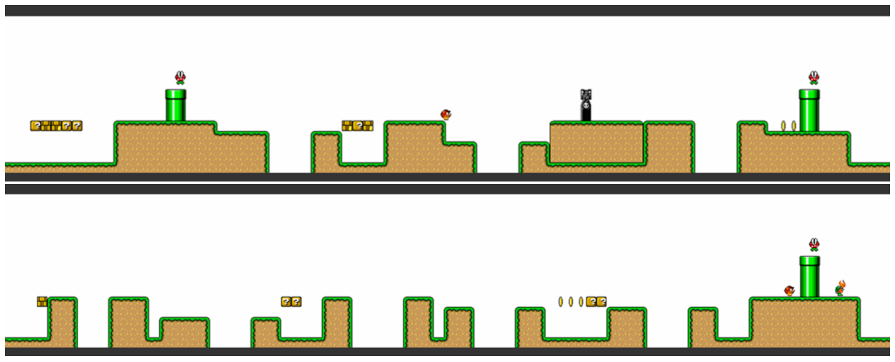

Fig. 10.5: The best levels evolved to maximise predicted challenge for two AI agents. Adapted from [26]

## 10.5 Lab exercise: Generate personalised levels for *Super Mario Bros.*

In this lab session, you will generate levels personalised for a specific player using the *InfiniTux* software. This is the same software interface used in Chapter 3, but this time the focus is on customising content to a specific playing style.

In order to facilitate meaningful detection of player experience and to allow you to develop player experience models, you will be given a dataset of 597 instances containing several statistical gameplay and content features collected from hundreds of players playing the game. The data contains information about several aspects of players' behaviour captured through features representing the frequencies of performing specific actions such as killing an enemy or jumping and the time spent doing certain behaviour such as moving right or jumping. Your task is to use this data to build a player-experience model using a machine learning or a data-mining technique of your choice. The models you build can then be used to recognise the gameplaying style of a new player.

After you build the models and successfully detect player experience, you should implement a method to adjust game content to changes of player experience in the game. You can adopt well-known concepts of player experience such as *fun*, *challenge*, *difficulty* or *frustration* and adjust the game content according to the aspect you would like your player to experience.

## 10.6 Summary

This chapter covered the experience-driven perspective for generating personalised game content. The rich and diverse content of games is viewed as a building block to be put together in a way that elicits unique player experiences. The experience-driven PCG framework [37] defines a generic and effective approach for optimising player experience via the adaptation of the experienced content.

To successfully adapt game content one needs to fulfill a set of requirements: the game should be tailored to individual players' experience-response patterns; the game adaptation should be fast, yet not necessarily noticeable; and the experience-based interaction should be rich in terms of game context, adjustable game elements and player input. The experience-driven PCG framework satisfies these conditions via the efficient generation of game content that is driven by models of player experience. The experience-driven PCG framework offers a holistic realization of affective interaction as it elicits emotion through variant game content types, integrates game content into computational models of user affect, and uses game content to adapt the experience.

## References

1. Bänziger, T., Tran, V., Scherer, K.R.: The Geneva Emotion Wheel: A tool for the verbal report of emotional reactions. In: Proceedings of the 2005 Conference of the International Society for Research on Emotion (2005)
2. Bianchi-Berthouze, N., Isbister, K.: Emotion and body-based games: Overview and opportunities. In: K. Karpouzis, G.N. Yannakakis (eds.) Emotion in Games: Theory and Praxis. Springer (2016)
3. Calleja, G.: In-Game: From Immersion to Incorporation. MIT Press (2011)
4. Conati, C.: Probabilistic assessment of user's emotions in educational games. Applied Artificial Intelligence **16**(7-8), 555–575 (2002)
5. Csikszentmihalyi, M.: Flow: The Psychology of Optimal Experience. Harper & Row (1990)
6. Drachen, A., Thurau, C., Togelius, J., Yannakakis, G.N., Bauckhage, C.: Game data mining. In: M. Seif El-Nasr, A. Drachen, A. Canossa (eds.) Game Analytics, pp. 205–253. Springer (2013)
7. Fürnkranz, J., Hüllermeier, E. (eds.): Preference Learning. Springer (2011)
8. Gratch, J., Marsella, S.: A domain-independent framework for modeling emotion. Cognitive Systems Research **5**(4), 269–306 (2004)
9. Holmgård, C., Yannakakis, G.N., Karstoft, K.I., Andersen, H.S.: Stress detection for PTSD via the StartleMart game. In: Proceedings of the 5th International Conference on Affective Computing and Intelligent Interaction, pp. 523–528 (2013)
10. Hunicke, R., Chapman, V.: AI for dynamic difficulty adjustment in games. In: Proceedings of the AAAI Workshop on Challenges in Game Artificial Intelligence, pp. 91–96 (2004)
11. IJsselsteijn, W., de Kort, Y., Poels, K., Jurgelionis, A., Bellotti, F.: Characterising and measuring user experiences in digital games. In: Proceedings of the 2007 Conference on Advances in Computer Entertainment Technology (2007)
12. Joachims, T.: Optimizing search engines using clickthrough data. In: Proceedings of the 8th International Conference on Knowledge Discovery and Data Mining, pp. 133–142 (2002)
13. Martínez, H.P., Bengio, Y., Yannakakis, G.N.: Learning deep physiological models of affect. IEEE Computational Intelligence Magazine **8**(2), 20–33 (2013)
14. Martínez, H.P., Garbarino, M., Yannakakis, G.N.: Generic physiological features as predictors of player experience. In: Proceedings of the 4th International Conference on Affective Computing and Intelligent Interaction, pp. 267–276 (2011)
15. Martínez, H.P., Yannakakis, G.N.: Mining multimodal sequential patterns: A case study on affect detection. In: Proceedings of the 13th International Conference on Multimodal Interfaces, pp. 3–10 (2011)

16. Martínez, H.P., Yannakakis, G.N., Hallam, J.: Don't classify ratings of affect; rank them! IEEE Transactions on Affective Computing **5**(3), 314–326 (2014)
17. Metallinou, A., Narayanan, S.: Annotation and processing of continuous emotional attributes: Challenges and opportunities. In: Proceedings of the IEEE Conference on Automatic Face and Gesture Recognition (2013)
18. Nijholt, A.: BCI for games: A 'state of the art' survey. In: Proceedings of the International Conference on Entertainment Computing, pp. 225–228 (2008)
19. Nintendo: (1985). Super Mario Bros., Nintendo
20. Ortony, A., Clore, G., Collins, A.: The Cognitive Structure of Emotions. Cambridge University Press (1990)
21. Persson, M.: Infinite Mario Bros. URL http://www.mojang.com/notch/mario/
22. Savva, N., Scarinzi, A., Berthouze, N.: Continuous recognition of player's affective body expression as dynamic quality of aesthetic experience. IEEE Transactions on Computational Intelligence and AI in Games **4**(3), 199–212 (2012)
23. Shaker, N., Asteridadis, S., Karpouzis, K., Yannakakis, G.N.: Fusing visual and behavioral cues for modeling user experience in games. IEEE Transactions on Cybernetics **43**(6), 1519–1531 (2013)
24. Shaker, N., Togelius, J., Yannakakis, G.N.: Towards automatic personalized content generation for platform games. In: Proceedings of the Artificial Intelligence and Interactive Digital Entertainment Conference, pp. 63–68 (2010)
25. Shaker, N., Yannakakis, G., Togelius, J.: Crowdsourcing the aesthetics of platform games. IEEE Transactions on Computational Intelligence and AI in Games **5**(3), 276–290 (2013)
26. Shaker, N., Yannakakis, G.N., Togelius, J., Nicolau, M., O'Neill, M.: Evolving personalized content for Super Mario Bros using grammatical evolution. In: Proceedings of the Artificial Intelligence and Interactive Digital Entertainment Conference, pp. 75–80 (2012)
27. Sweetser, P., Wyeth, P.: Gameflow: A model for evaluating player enjoyment in games. ACM Computers in Entertainment **3**(3) (2005)
28. Tognetti, S., Garbarino, M., Bonarini, A., Matteucci, M.: Modeling enjoyment preference from physiological responses in a car racing game. In: Proceedings of the IEEE Symposium on Computational Intelligence and Games, pp. 321–328 (2010)
29. Yannakakis, G.N.: Preference learning for affective modeling. In: Proceedings of the 3rd International Conference on Affective Computing and Intelligent Interaction (2009)
30. Yannakakis, G.N., Hallam, J.: Ranking vs. preference: A comparative study of self-reporting. In: Proceedings of the International Conference on Affective Computing and Intelligent Interaction, pp. 437–446 (2011)
31. Yannakakis, G.N., Martínez, H.P.: Grounding truth via ordinal annotation. In: Proceedings of the 6th International Conference on Affective Computing and Intelligent Interaction, pp. 574–580 (2015)
32. Yannakakis, G.N., Martínez, H.P.: Ratings are overrated! Frontiers in ICT **2**, 13 (2015)
33. Yannakakis, G.N., Martínez, H.P., Garbarino, M.: Psychophysiology in games. In: K. Karpouzis, G.N. Yannakakis (eds.) Emotion in Games: Theory and Praxis. Springer (2016)
34. Yannakakis, G.N., Martínez, H.P., Jhala, A.: Towards affective camera control in games. User Modeling and User-Adapted Interaction **20**(4), 313–340 (2010)
35. Yannakakis, G.N., Paiva, A.: Emotion in games. In: R.A. Calvo, S. D'Mello, J. Gratch, A. Kappas (eds.) Handbook of Affective Computing. Oxford University Press (2013)
36. Yannakakis, G.N., Spronck, P., Loiacono, D., Andre, E.: Player modeling. In: Dagstuhl Seminar on Artificial and Computational Intelligence in Games, pp. 45–59 (2013)
37. Yannakakis, G.N., Togelius, J.: Experience-driven procedural content generation. IEEE Transactions on Affective Computing **2**(3), 147–161 (2011)