

# Crowd-Sourcing the Aesthetics of Platform Games

Noor Shaker, Georgios N. Yannakakis, *Member, IEEE*, Julian Togelius, *Member, IEEE*

**Abstract**—What are the *aesthetics* of platform games and what makes a platform level engaging, challenging and/or frustrating? We attempt to answer such questions through mining a large-set of *crowd-sourced* gameplay data of a clone of the classic platform game *Super Mario Bros*. The data consists of 40 short game levels that differ along six key level design parameters. Collectively, these levels are played 1560 times over the Internet and the perceived experience is annotated by experiment participants via self-reported ranking (pairwise preferences). Given the wealth of this crowd-sourced data, as all details about players’ in-game behaviour are logged, the problem becomes one of extracting meaningful numerical features at the appropriate level of abstraction for the construction of generic computational models of player experience and, thereby, game aesthetics. We explore dissimilar types of features, including direct measurements of event and item frequencies, and features constructed through frequent sequence mining and go through an in-depth analysis of the interrelationship between level content, player’s behavioural patterns and reported experience. Furthermore, the fusion of the extracted features allows us to predict reported player experience with a high accuracy even from short game segments. In addition to advancing our insight on the factors that contribute to platform game aesthetics, the results are useful for the personalisation of game experience via automatic game adaptation.

## I. INTRODUCTION

An algorithm that could automatically judge how engaging or interesting a particular piece of game content is — that is, a *computational* model of the *aesthetics* of game content — would be useful for several reasons. One strong reason is that such a method would help us to automatically or semi-automatically generate good content, another is that analysis of the algorithm could help us understand what players like in games, and ultimately contribute to understanding the cognitive and affective procedures behind human entertainment and motivation in general. As players tend to vary significantly in their preferences it would further be useful to have an algorithm that, given information about a particular player, could predict the appeal of the game content for that player. Finally, having an algorithm that could observe a human playing a game and accurately judge what the human is experiencing as he/she is playing the game would also be useful, as this could allow us to adapt the game to the player, and also help us understand how human affect is expressed in behaviour.

A number of researchers have attacked this problem from a top-down perspective, that is, by creating theories of the aesthetics of game content and game play based on

introspection or qualitative research methods. For example, Malone [1] has proposed that computer games are “fun” when they have the right amount of challenge and evoke curiosity and fantasy, and Koster [2] has proposed that fun in games is connected to the player learning to play the game. Magerko et al. [3] research within learning games proposed an adaptation framework based on a predefined set of learning style.

Such theories are in general too high-level and vague about key concepts to be implemented in algorithms, though some attempts have been made to create computational models based on them [4], [5].

Other authors have tried to identify more specific and concrete elements of game design and game content that contribute to player experience, so called “patterns in game design”; Björk and Holopainen [6] are in an ambitious on-going effort cataloguing hundreds of such patterns, whereas other authors discuss patterns in content design for individual genres, such as first-person shooters. For example, Hullett and Whitehead [7] analyse some key patterns in first-person shooter games, such as sniper positions and open arenas and discuss how they contribute to player entertainment. In [8], [9] presented a system that visualizes players’ behaviours to allow analysts to easily identify patterns and design issues. Jennings-Teats et al. [10] showed how player experience can be altered by presenting sequences of level segments raked by their difficulty and presented to the player according to her behaviour.

Of particular interest for our current concern is the work of Smith et al. [11], who have analysed platform game levels and proposed a hierarchical ontology for such levels where cells contain rhythm groups which in turn consist of components such as platforms, collectibles and switches. The authors further hypothesise about how certain design choices might affect player experience, such as short and uneven rhythm groups making the level more challenging and longer rhythmic sections demanding sustained concentration. These principles were eventually incorporated into the Tanagra level generator, which can create levels with rhythmic structure but does not include methods for judging the aesthetics of completed levels [12].

If we find accurate such theories, we can then create *theory-driven* models of game aesthetics. However, even if the theories are correct and sufficiently extensive to allow prediction of player experience in a wide range of situations, they would also need to be quantitative in order to be incorporated within an algorithm, something most current theoretical efforts to understand game aesthetics are not. They would also need to be grounded in measurable quantities. For example, theories based on design patterns would

need to be accompanied by algorithmic ways of detecting and locating such patterns.

The alternative, complimentary approach is to create *data-driven* (bottom-up) models of game aesthetics based on collecting data about games, game content and player’s behaviour, and correlating this data with data annotated with player experience tags. This approach, which builds on machine learning and/or other statistical methods, can be seen as *crowd-sourcing* aesthetics modelling. A few researchers have attempted to create such experience models via the affective annotation of data streams such as sounds and videos [13], but the application of crowd-sourcing based approaches for player testing and game aesthetics has not yet been investigated. The approach is also closely linked to massive-scale game data mining [14], [15]; however, direct annotations of player experience are not generally available in those studies. For an overview of research on building aesthetic game models from data, as well as a multi-faced framework that interconnects player experience modelling and game adaptation the reader is refer to the *experience-driven procedural content generation* framework [16].

In this work we are taking a slightly narrower view of aesthetics. We judge the aesthetics of the level design from the players’ point of view based on the content generated and the gameplay experience it provides. The content can be generated automatically by an algorithm or it can handcrafted by a designer, or indeed be created in a mixed-initiative (co-creation) fashion. We are trying to devise a data-driven approach that can automatically extract game design patterns from existing games. These patterns can be used by an algorithm for adapting the game, or they can be generalized and used by game designers when constructing a new game. The focus of the proposed work is not on adding to what we know about what makes a level frustration or challenging but rather to construct a quantitative measure of game aesthetics.

#### A. Relation to our own previous work

We have in the past published several papers about the computational aesthetics of the platform game used in this paper, *Infinite Mario Bros*. Our first papers [17], [18] reported on the construction of models that predict six different aspects of player experience based on 36 features extracted from gameplay and 4 controllable features, which could be used to generate levels. We used forced choice questionnaires to collect player experience data, and neuroevolutionary preference learning combined with feature selection to induce the models, just as we do in this paper. Data was collected from 480 game sessions, played by at most 240 different players. Models were found that predicted certain aspects of player experience with between 73% and 91% accuracy.

A follow-up paper [19] used the same dataset as the previous papers, but focused on generating levels based on the models we had learnt. The levels were generated by systematically varying the four controllable features between high and low states until the parameter set was found which yielded the highest or lowest predicted value on one of the six

player experience dimensions. That parameter configuration was then used to generate personalised levels for the player that either maximise predicted challenge, frustration or fun. The generated levels were in turn tested using both human players and algorithmic agents playing the game verifying that the adaptation mechanism worked by tracking predicted player preference over several levels.

While these experiments were successful, it became apparent that the dataset had some limitations. The number of controllable features (and the number of configurations of these features that were tested) was too small to permit meaningful exploration of the search space and possibilities of finding interestingly new design parameter configurations. Also, one of the controllable features (direction switching) and three of the player experience dimensions (predictability, anxiety and boredom) turned out to be relatively uninteresting to explore in the context of the current game. The levels used in the first data set took about a minute to play each, which we judged was overly long given that we wanted our model to apply to the aesthetics of the moment, in order to enable online adaptation. Finally, and most importantly, we wanted to record more detailed information about both levels and gameplay in order to see if we could find a way to predict player experience even better — to squeeze more information out of the data, as it was. We therefore embarked on collecting a new dataset, with more levels (40) and more players (1560 games were played). In a recent paper [20] we reported on preliminary explorations of this dataset. There, we tried to predict player experience of reported engagement based only on level features (disregarding all gameplay traces), and introduced the use of frequent subsequence counts (as found by sequence mining algorithms) as features extracted from levels. We also explored predicting features from only parts of levels, in order to find the minimum level segment length which would allow us to perform meaningful adaptation. It was found that both restricting level segment lengths and disregarding player metrics significantly decreased the predictive power of derived models.

In this paper, we explore the same dataset as the one used in [20] to a much greater depth. We investigate a number of ways of extracting sequence data from levels and play traces that go beyond what we used in all previous modelling attempts, and we explore the predictive power of new direct (non-sequential) measures of both levels and player metrics, both on their own and in combination with sequence data. The goal is to create models that predict player experience as well as possible from observing a game level and how well a player plays it. We believe the methods we develop along the way to be potentially useful for other games which have a linear structure.

## II. TESTBED PLATFORM GAME

The testbed platform game used for our study is a modified version of Markus Persson’s *Infinite Mario Bros* (IMB) which is a public domain clone of Nintendo’s classic platform game *Super Mario Bros* (SMB). IMB features the same art

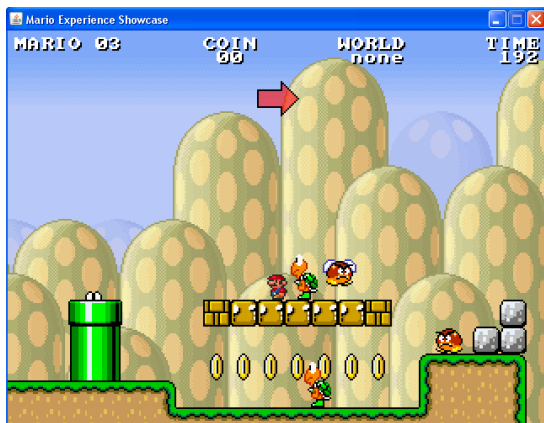


Fig. 1. Snapshot from Infinite Mario Bros, showing Mario standing on horizontally placed blocks surrounded by different types of enemies.

assets and general game mechanics as SMB but differ in level construction; while human-authored levels have been constructed for the original Super Mario Bros., Infinite Mario Bros features infinite number of procedurally generated levels. The gameplay in Infinite Mario Bros consists of moving the player-controlled character, Mario, through two-dimensional levels. Mario can walk and run, duck, jump, and shoot fireballs. The main goal of each level is to get to the end of the level. Auxiliary goals include collecting as many coins as possible, and clearing the level as fast as possible. For more details about the game the reader may refer to [17].

The game itself is very well known, and the benchmark software has been used relatively extensively as a testbed for research and as a testing environment for various AI techniques [21], [22], [23], [19], [24], [25]. The game is also being used as a benchmark for the Mario AI Championship<sup>1</sup> [26].

Infinite Mario Bros has been chosen because of the popularity of Super Mario Bros, the high similarity between the two, the availability of an open source clone of the game which makes development and data collection easier and because of the 2D design and game mechanics it provides which are similar to other games from the same genre. The game has been used for this study not primarily in order to find new design insights, but rather to validate that the methodology could be used to find new design insights if used on a less-known game genre.

While implementing most features of Super Mario Bros, the standout feature of Infinite Mario Bros is the automatic generation of levels. Every time a new game is started, levels are randomly generated by traversing a fixed width and adding features according to certain heuristics as specified by placement parameters. In our modified version, we concentrated on a number of selected parameters that affect gameplay experience.

### III. DATA COLLECTION

Data from gameplay and questionnaires have been collected from hundreds of players over the Internet via a crowd-sourcing experiment. Complete games were logged, including the levels the players played and what actions the players took at which time, enabling complete replays. The following three types of data was extracted from raw logs and replays: content, gameplay and annotated (self-reported) player experience.

#### A. Content Data

Two types of content features have been extracted: direct and sequential. The direct content features are also named *Controllable* as they are used to generate the levels and are varied to make sure several variants of the game are played and compared. The level generator of the game has been modified to create levels according to the following six controllable features:

- The number of gaps in the level,  $G$ ; gaps are the holes in the game in which Mario may fall and die.
- The average width of gaps,  $G_w$ .
- The number of enemies,  $E$ . This parameter controls the number of Goombas (mushroom-like enemies) and Koopas (turtle-like enemies) scattered around the level, affecting the level difficulty.
- Enemies placement,  $E_p$ . The way enemies is placed around the level determined by three probabilities which sum to one.
  - Around horizontal boxes,  $P_x$ : Enemies are placed on or under a set of horizontal blocks (a number of blocks placed horizontally without connection to the ground).
  - Around gaps,  $P_g$ : Enemies are placed within a close distance to the edge of a gap.
  - Random placement,  $P_r$ : Enemies are placed on a flat space on the ground.

Fig. 2 illustrates positioned enemies by giving different values for  $P_b$ ,  $P_g$  and  $P_r$ . Fig. 2.(a) of the figure shows enemies placed by setting  $P_b$  to 80%. Fig. 2.(b) illustrates the result of setting  $P_g$  to 80%, and Fig. 2.(c) is the result of  $P_r = 80\%$ .

- The number of powerups,  $N_w$ . Mario can collect powerup elements hidden in boxes to upgrade his state from little to big or from big to fire.
- The number of boxes,  $B$ . We define one variable to specify the number of the two different types of boxes that exist in Infinite Mario. These two types of boxes are here referred to as *blocks* and *rocks*. Blocks (which look like squares with question marks) contain hidden elements such as coins or powerups. Rocks (which look like squares of bricks) may hide a coin, a powerup or simply be empty. Mario can smash rocks only when he is in big mode.

The generation of levels with specified values for all parameters is guaranteed by the generator; while generating

<sup>1</sup><http://www.marioai.org/>

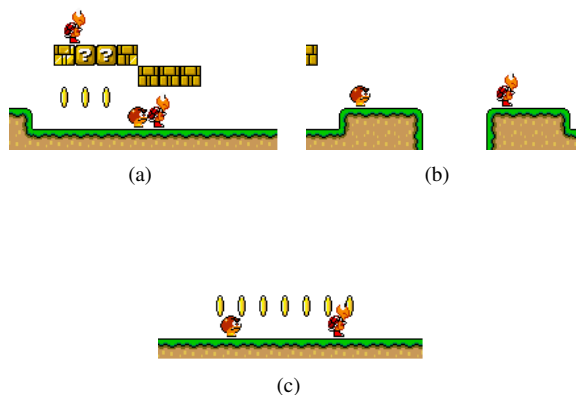


Fig. 2. Enemies placement using different probabilities: high probability is given to placement around horizontal boxes,  $P_b$  (a), around gaps,  $P_g$  (b), and to random placement,  $P_r$  (c).

the levels, and whenever an item is to be added, these parameters are checked and the item is placed accordingly.

Game designers, who are familiar with 2D platform games, and in particular Super Mario Bros, have been consulted when selecting the controllable features. The features have been chosen based on their impact on the investigated affective states and their generality to other 2D platform games. Note that consulting game designers doesn't conflict with the bottom-up approach followed which derives models of player experience based on data collected from players while the designers' knowledge is incorporated only when designing the experiment.

The first two features appeared in our previous studies [27], [19], whereas the four new features are explored for the first time here.

Two states (low and high) are set for each of the controllable parameters above except for enemies placement which has been assigned three different states allowing more control over the difficulty and diversity of the generated levels. The total number of pairwise combinations of these states is 96. This number can be reduced to 40 by analysing the dependencies between these features and eliminating the combinations that contain independent variables. All levels have been checked before starting the data collection in a way that assures their compatibilities with the intent parameters assigned. An example level generated by one possible combination of the controllable features is presented in Fig. 3.

In addition to the direct (controllable) features, sequential content features are also extracted. The topology of the levels is converted into sequences of numbers representing different types of game items and sequence mining techniques are applied to extract useful patterns from the resulted sequences (see Section V-B).

### B. Gameplay Data

While playing the game, different player actions and interactions with game items and their corresponding timestamps have been recorded. These events are categorised

in different groups according to the type of the event and the type of interaction with the game objects. The events recorded are the following: level completion event; Mario death event and cause of death; interaction events with game items such as free coins, empty rock, coin block/rock and power-up rock/block; Mario enemy kill event associated with the type of actions performed to kill the enemy and the type of enemy; changing Mario mode (small, big or fire) event; changing Mario state (moving right, left, jump, run, duck) event; and the full trajectory of Mario as a combination of events.

Both direct and sequential gameplay features have been extracted based on the above-mentioned events. The detailed list of features is presented in Section V.

### C. Reported Player Experience Data

We rely on self-reporting annotations based on previous research in which very accurate player experience models of self-report affective states have been constructed [28], [27]. However, a number of limitations are embedded in the players self-reporting experience modeling including noise due to learning and self-deception, disruption to game play experience and sensitivity to memory limitations. In order to minimize these effects we rely on annotated player experience data collected via a 4-alternative forced choice questionnaire presented after small game sessions. The questionnaire asks the player to report the preferred game for three user states: *engagement*, *challenge* and *frustration*. The selection of these states is based on earlier game survey studies [27] and our intention to capture both affective and cognitive/behavioural components of gameplay experience [16].

The questionnaire protocol gives the players the following alternatives:

- game A [B] was/felt more  $E$  than game B [A] (cf. 2-alternative forced choice);
- both games were/felt equally  $E$  or
- neither of the two games was/felt  $E$ .

where  $E$  is the affective state under investigation.

## IV. EXPERIMENTAL PROTOCOL

The game survey study has been designed to collect subjective affective reports expressed as pairwise preferences of subjects playing the different variants (levels) of the tested game by following the experimental protocol proposed in [29].

According to the protocol, each subject plays a predefined set of two games. The games played differ in the levels of one or more of the six controllable features presented previously.

The game sessions presented to players have been constructed using a level width of 100 Infinite Mario Bros units (blocks), about one-third of the size usually employed when generating levels for Infinite Mario Bros game in our previous experiments [27], [19]. The selection of this length was due to a compromise between a window size that is



Fig. 3. An example level generated and used to collect the data.

big enough to allow sufficient interaction between the player and the game to trigger the examined affective states and a window which is small enough to set an acceptable frequency of an adaptation mechanism applied in real-time aiming at closing the affective loop of the game [30]. A previous study [20] has been conducted to test whether a smaller level width than the chosen one can be used to construct models for predicting players' engagement from game content with higher accuracy than the models constructed based on information from levels with the chosen width. The study concluded that the models perform best when trained on features extracted from levels with the selected width rather than from levels with half or one-third of the width.

A total number of 780 players participated in this crowdsourcing experiment. Participants' age covers a range between 16 and 64 years (31.5% females) while their location includes Denmark (46.11%), Greece (8.9%), Ireland (1.48%), USA (3.34%), Holland (0.74%), Finland (1.36%), France (0.37%), Syria (0.25%), Sweden (0.37%), Korea (0.12%), Spain (0.25%) or unknown (36.71%).

## V. FEATURE EXTRACTION

In the following sections we describe the types of features that we have extracted from the recorded content and gameplay data via direct and sequential feature extraction.

Most of the direct features presented appear in our previous studies [17], [19]. These features are used in this work due to their relevance for modelling player experience. In this work we also investigate sequential patterns extracted from gameplay data.

### A. Direct Gameplay Features

Several features have been directly extracted from the data recorded during gameplay (see Section III). The choice of these features is made in order to be able to represent the difference between a large variety of Infinite Mario Bros playing styles. In addition to the six controllable game features that are used to generate Infinite Mario Bros levels, the features presented in Table I are extracted from the gameplay data collected and are classified in five categories: time, interaction with items, interaction with enemies, death and miscellaneous.

### B. Sequential Features

We investigate another form of indirectly representing the gameplay interaction by means of sequences which allows

including features that are based on ordering in space or time. Gameplay features presented in the previous section provide a quantitative measure about different types of game content and playing style. Alternatively, analysing sequences of game content and players' behaviour yields patterns that might be directly linked to player experience. For example, we would like to extract features that encapsulate whether a player performed a particular action before or after encountering a specific in-game situation.

Modelling players' experience based on features extracted from sequential information provides a promising alternative for models constructed based on direct feature extraction, and by fusing these two types of representations, we anticipate to construct more accurate models of player experience than those constructed on one of these form of data representation at a time.

In the following sections, we describe different criteria of constructing sequences from game content, gameplay, and the interaction between the two. We present two sequence mining approaches and further discuss different setups that can be used for mining the extracted sequences.

Table II presents the different possible approaches that can be followed to generate different types of sequences. The columns represent the different order and frequency at which information is logged. The rows represent what type of data is logged each time an event occurs. We will be distinguishing the following orders/frequencies, while acknowledging that even more fine-grained distinctions are possible:

- $t_{small}$ : time step. Information is logged at a constant rate (e.g. once per second), regardless of what the player does. This yields a sequence with a length proportional to the time taken by the player to play the level.
- Block: Information is logged once per block in the level, independent of the time taken by the player to traverse the level. This yields a sequence with a length equal to the level.
- Gameplay event: Information is logged each time the player changes the command issued (pressing/releasing a button or changing direction) or something else happens (e.g. Mario changes the mode or stomps an enemy).

The information logged each time anything is logged can be either game content ( $C$ ), player (gameplay) behaviour ( $A$ ) or both game content and player's behaviour ( $M$ ).

We will focus the discussion for the rest of the paper on the five sequence types marked with an X in Table II.



TABLE I  
FEATURES EXTRACTED FROM DATA RECORDED DURING GAMEPLAY

Category	Feature	Description
Time	$t_{comp}$	Completion time
	$t_{play}$	Playing duration of last life over total time spent on the level
	$t_{duck}$	Time spent ducking (%)
	$t_{jump}$	Time spent jumping (%)
	$t_{left}$	Time spent moving left (%)
	$t_{right}$	Time spent moving right (%)
	$t_{run}$	Time spent running (%)
	$t_{small}$	Time spent in Small Mario mode (%)
Interaction with items	$t_{big}$	Time spent in Big Mario mode (%)
	$n_{coin}$	Free coins collected (over all coins existent in the level)
	$n_{coinBlock}$	Coin blocks pressed or coin rocks destroyed (over all blocks and rocks existent)
	$n_{powerups}$	Powerups pressed (over all powerups existent)
Interaction with enemies	$n_{blocks}$	Sum of all blocks and rocks pressed or destroyed (over all blocks and rocks existent)
	$k_{cannon}$	Times the player kills a cannonball or a flower (over all cannon and flower enemies existent)
	$k_{goomba}$	Times the player kills a goomba or a koopa (over all goombas and koopas existent)
	$k_{stomp}$	Opponents died from stomping (%)
Death	$k_{unleash}$	Opponents died from unleashing a koopa shell (%)
	$d_{total}$	Total number of deaths
Miscellaneous	$d_{cause}$	Cause of the last death
	$n_{mode}$	Number of times the player shifted the mode (Small, Big, Fire)
	$n_{jump}$	Number of times the jump button was pressed
	$n_{miscJump}$	Difference between the total number of gaps and the total number of jumps
	$n_{duck}$	Number of times the duck button was pressed
	$n_{state}$	Number of times the player changed the state between: standing still, run, jump, moving left, and moving right

TABLE II

THE DIFFERENT TYPES OF SEQUENCES THAT CAN BE GENERATED. COLUMNS PRESENT THE TYPE OF EVENT TO BE RECORDED, WHILE ROWS PRESENT WHEN TO RECORD THE EVENT. THE COMBINATIONS MARKED WITH AN X ARE THE ONES INVESTIGATED IN THIS PAPER

	$t_s$	Block	Gameplay Event
$A_h$ (Player Behaviour)	X		X
$C$ (Content)		X	X
$M$ (both)			X

Although we only investigate a few sequence types of all those available, our sample provides a variety of options that cover different aspects of playing experience.

Once we know what to sample and when, the question remains how to turn this information into sequences using a low-cardinal alphabet. Below, we discuss how to do this for levels and for gameplay traces.

1) *Sequential Content Features*: Sequences capturing different information about level geometry have been extracted by converting the content of the levels into numbers representing different types of game items. Three different representations of game content have been investigated. The full list of events considered as well as their graphical representation is presented in Table III.







- Platform structure,  $P$ : A sequence is generated by comparing the height of each block across the level with the height of the previous block and recording the following values: 0 if no difference found (🟫); 1 if there is an increase in the platform height (🟪); 2 if there is a decrease in the platform height (🟩); and, 3 and 4 to mark the beginning (🟩) and the ending (🟫) of a gap, respectively. Fig. 4.(a) presents part of a level and the corresponding platform structure sequence

representation.

- Enemy and item placement,  $I$ : The term items refers to the coins and the different types of boxes scattered around the level. The existence and non-existence states for enemies and items have been combined together resulting in four different possible values: 0, 1, 2 and 3 corresponding, respectively, to non-existence of either enemies or items, the existence of an enemy (👹), the existence of an item (🍄), and the existence of an enemy and an item (👹🍄). Fig. 4.(b) illustrates an example level segment where the above-mentioned four states are presented.
- Content corresponding to gameplay events,  $C_g$ : We explored another method in which game content at the specific player position is recorded whenever the player performs an action or interacts with game items. In this case, different content events are used: increase in platform height, 🟪; decrease in platform height, 🟩; existence of an enemy, 👹; existence of a coin, block or rock, 🍄; existence of a coin, block or rock with an enemy, 👹🍄; and the beginning, 🟩, and the end 🟫, of a gap.

2) *Sequential Gameplay Features*: Sequences representing different players' behaviour have been generated by recording key pressed/released events (*action event*) or interaction with items events. The action event might represent a simple action performed such as pressing an arrow key to move left or right; or more complex player's behaviours that can be achieved by pressing a combination of keys at the same time (e.g. jumping over a big gap requires the player to press the run and jump keys together for a number of time steps). The following list describes the different events that

TABLE III  
THE DIFFERENT TYPES OF EVENTS CONSIDERED WHEN GENERATING THE SEQUENCES AND THEIR GRAPHICAL REPRESENTATION.

Category	Graphical Representation	Description
Platform Structure		Flat platform
		Increase/decrease in the platform height
		The beginning/ending of a gap
Enemies and Items		The existence of an enemy
		The existence of coin, block, or brick block
		The existence of enemy with a rewarding item
Gameplay	$\blacktriangleright, \blacktriangleleft, \blacktriangledown$	Moving right, left or duck
	$\uparrow$	Jumping
	$\uparrow\blacktriangleright, \uparrow\blacktriangleleft$	Jumping right or left
	$R\blacktriangleright, R\blacktriangleleft$	Running right or left
	$R\blacktriangleright\uparrow, R\blacktriangleleft\uparrow$	Running while jumping right or left
	$S$	Not pressing any key
	$E_s$	Stomping on an enemy
	$U$	Unleashing a koopa shell
	$O$	Changing Mario Mode
	$W, L$	Winning or losing the game

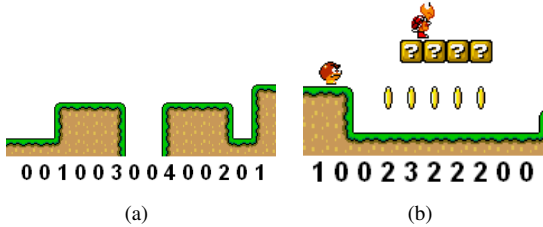


Fig. 4. Snapshot from a level and the corresponding platform structure sequence representation,  $P$  (a), and enemies and items sequence representation,  $I$  (b).

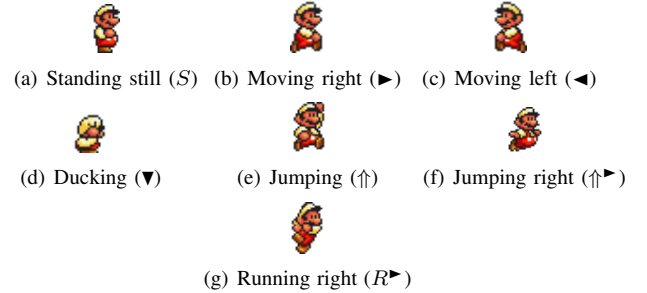


Fig. 5. A graphical representation of the different actions that can be performed by the player.

have been considered (Table III).

- Pressing an arrow key to move right, left, or duck ( $\blacktriangleright$ ,  $\blacktriangleleft$ ,  $\blacktriangledown$ ).
- Pressing the jump key,  $\uparrow$ .
- Pressing the jump key in combination with right or left key ( $\uparrow\blacktriangleright$ ,  $\uparrow\blacktriangleleft$ ).
- Pressing the run key in combination with right or left key ( $R\blacktriangleright$ ,  $R\blacktriangleleft$ ).
- Pressing the run and jump keys in combination with right or left key ( $R\blacktriangleright\uparrow$ ,  $R\blacktriangleleft\uparrow$ ).
- Not pressing any key,  $S$ .
- Winning the game,  $W$ .
- Losing the game,  $L$ .
- Killing an enemy by stomping,  $E_s$ .
- Unleashing a koopa shell,  $U$ .
- Changing Mario mode,  $O$ .

Fig. 5 presents the graphical interpretation for most of the actions that can be performed.

In this paper, we will consider two time window  $t$  values for generating sequences: 0.5 sec ( $A_{0.5}$ ) and 0.25 sec ( $A_{0.25}$ ) meaning that an event will be registered every half or quarter of a second, respectively. We also consider sequences generated whenever the player switch the action,  $A$ . (Note that Infinite Mario is a fast-paced game in which the player

could in theory perform an action every  $1/24$  sec).

The purpose of recording these events is that players' behaviour and playing style can be analysed by looking at events generated by each player and how frequent each of these event occurs. Generating a sequence combining these events in a timely manner provides a more in-depth insight about more complex behaviour patterns that might have an impact on players' experience.

The resulted sequences of players' behaviour have a wide variety both in terms of length and structure, which reflects the diversity of players' playing style and complicate any sequence mining algorithm that can be applied to extract useful information. This diversity is reflected on the normalised compression distance (NCD) measure [31] that has been applied to test for structural similarity between the sequences. The results of applying this function on each pair of the actions sequences showed a high dissimilarity between the sequences (NCD>0.6 in 71.32% of the cases).

3) *Fused Sequential Features of both Game Content and Gameplay Data*: Game content and players' behaviour events have been fused together to generate bimodal sequences ( $M$ ). Events from the two modalities have been extracted with their corresponding time stamp and then logged in temporal order. The generated event contains information

about the game content at the specific position in the game where the gameplay event occurred (which is one of the events mentioned in Section V-B1 or none if no content event from the list happens to occur at this specific position) along with the type of the gameplay event.

## VI. SEQUENCE MINING

Sequence mining techniques have been applied to extract useful information from the different types of the sequences generated. Two algorithms for frequent itemset mining have been implemented to find frequent sequence patterns within the dataset of sequences: SPADE and GSP. The SPADE [32] algorithm has been used to mine single-dimensional sequences that represent game content independently of player’s behaviour, namely, platform structure ( $P$ ) and enemies and items placement ( $I$ ). This algorithm has been used in our previous work [20] for mining content sequences, and is used in the same way in this paper. Mining sequences across multiple time series of data — content corresponding to gameplay events ( $C_g$ ), player’s behaviour ( $A$ ) and multimodal sequences ( $M$ ) — can be achieved via the Generalised Sequential Pattern (GSP) algorithm [33]; Martinez and Yannakakis [34] have used GPS to obtain frequent subsequences across multiple modalities of player input (physiology and game-based context).

In the following sections we provide a short list of frequent subsequence mining definitions and give a brief description of the two algorithms and the way they have been used in this paper.

### A. Definitions

A *data-sequence* is a sample of a sequential dataset where each sequence consists of a number of events, each one associated with a time stamp. The events are ordered by increasing time.

A *sequence pattern*  $l_i$  is a non-empty set of simultaneous events denoted by  $\langle e_0e_1e_2\dots e_n \rangle$  where  $e_i$  is an event. A data-sequence supports a sequence pattern if and only if it contains all the events present in the pattern in the same order but not necessarily consecutive.

A minimum support  $min_{sup}$  is the minimum number of times a pattern  $l_i$  has to occur in the data-sequences to be considered frequent. If the number of occurrences of  $l_i$  in the data-sequences exceeds the  $min_{sup}$ , we call  $l_i$  a *frequent pattern* and in this case, the fraction of data-sequences that support  $l_i$  is referred to as *support count*,  $sup_{count}$ .

### B. SPADE

A modified version of the SPADE algorithm [32] has been implemented to extract frequent subsequences of different game events. Game content for the 40 levels has been converted into numbers representing different types of content events as described in Section V-B1. Different subsequence lengths and minimum support thresholds values have been explored. A minimum support threshold of 20 has been used, meaning that each subsequence should occur at least in half of the levels to be considered frequent.

### C. GSP

The GSP algorithm [33] solves the sequence mining problem based on an apriori algorithm with a number of generalisations. Using GSP, we can discover patterns with a predefined minimum support, define time constraints within which adjacent events can be considered elements of the same pattern, and specify a time window for events from different modalities to be considered as synchronous events.

GSP generalises the basic definition of frequent sequential pattern by introducing two relaxation schemes:

- **Sliding window:** This generalisation allows the items of a pattern to be contained in the union of the items belonging to different time-series. According to this relaxation, a sequence  $s = \langle s_i s_j \rangle$  — where  $s_i$  and  $s_j$  can be contained in different time-series — is allowed to be counted as a support for a subsequence  $c$  as long as the time difference between  $s_i$  and  $s_j$  is less than the user specified window-size,  $max_{win}$ .
- **Time constraints:** This relaxation specifies the time gap between consecutive events from one or two different time-series. Given a user-defined gap,  $max_{gap}$ , a data-sequence supports the a pattern of two consecutive events  $\langle s_i s_{i+1} \rangle$  if and only if  $s_i$  and  $s_{i+1}$  occur in the sequence of the specified order and with a time difference lower than the specified  $max_{gap}$ .

The GSP algorithm is used for mining sequences that rely on players’ behaviour ( $C_g$ ,  $A_t$ ,  $A$ ) since it allows more generalised frequent patterns to be found by exploring different  $max_{gap}$ , and it is also used of mining multimodal sequences ( $M$ ) as, by using  $max_{win}$ , we can discover simultaneous events from two different modalities.

Different  $min_{sup}$  values have also been explored to obtain a reasonable trade off between considering patterns that are generalised over all players and more specific patterns. For the experiments presented in this paper, we use a  $min_{sup}$  of 500 which forces a sequence pattern to occur in at least 31.8% of the samples to be considered frequent.

The  $max_{win}$  defines the threshold under which events from two different modalities can be considered as simultaneous events. In this paper, we use  $max_{win} = 1$  second. The value for this parameter has been chosen as a trade off between a small window size that does not consider simultaneous events, and a window size that process clearly asynchronous events from two modalities as events happening in a very small interval. For the rest of this paper we will use parentheses to group simultaneous events.

The  $max_{gap}$  parameter is used to set up the time gap between two events to be considered as belonging to the same pattern. This parameter has a great impact to the number of frequent patterns that can be extracted. By assigning a large value to this parameter, we allow more generalised patterns to be taken into account and as a consequence, a large number of sequences will be counted as  $sup_{count}$ . Another drawback for using large  $max_{gap}$  values is that it allows considering less informative patterns. Correctly tuning this parameter has a large impact on the informativeness of the resulted patterns,



TABLE IV

NUMBER OF FREQUENT SEQUENTIAL PATTERNS FOUND FOR DIFFERENT SEQUENCE LENGTH VALUES ACROSS DIFFERENT TYPES OF SEQUENCES ( $min_{sup}$  IS 500 AND  $max_{gap}$  IS 1 SEC). THE COLUMNS STAND FOR: CONTENT CORRESPONDING TO GAMEPLAY EVENTS ( $C_g$ ), GAMEPLAY BEHAVIOUR ( $A$ ), GAMEPLAY BEHAVIOUR REGISTERED EVERY 0.5 SEC ( $A_{0.5}$ ) AND EVERY 0.25 SEC ( $A_{0.25}$ ) AND MULTIMODAL SEQUENCES OF GAME CONTENT AND PLAYER'S BEHAVIOUR ( $M$ ).

Length	$C_g$	$A$	$A_{0.5}$	$A_{0.25}$	$M$
1	7	8	2	8	18
2	27	64	2	57	205
3	25	310	0	69	939
4	23	939	0	741	1982
5	29	2065	0	1810	2957
6	0	2636	0	2547	2806
7	0	1403	0	1400	1402
8	0	115	0	112	112

specially when mining multimodal sequences. For instance, if we use  $max_{gap} = 3sec$ , the pattern ( $\uparrow, \rightarrow, \rightarrow$ ) can be supported by any sequence in which the player jumps, moves right and encounters an enemy within a 3 seconds interval (note that within this interval, the player might encounter more than one enemy or a gap between the jumping and moving right events which makes this pattern somehow misleading). The experiments conducted for tuning the value of this parameter showed that a  $max_{gap}$  of 1 sec provides a good trade off between the number of patterns extracted and their expressiveness value.

#### D. Length of Sequences

Table IV presents the different types of sequences and the number of frequent subsequence found for a number of different sequence generation methods. As can be seen from the table, the number of extracted subsequences is quite large for sequences containing information about players' behaviour ( $A$ ), and the search space for automatic feature selection increases substantially when fusing content and gameplay events for generating sequences ( $M$ ); more than 2000 subsequences of length six have been extracted from the players' behaviour and multimodal sequences.

In order to lower the feature space dimensionality and the computational cost of searching for relevant features we chose to use only frequent sequences of length three.

#### VII. PREFERENCE LEARNING FOR MODELLING PLAYING EXPERIENCE

In order to construct models that approximate the function between gameplay features, controllable features and reported affective preferences, we use neuroevolutionary preference learning. In other words, we use artificial evolution for shaping artificial neural networks (ANNs) whose output matches the reported (pairwise) preferences of the players.

We proceed in a three-phase procedure in order to find networks that predict preferences with high accuracy.

- 1) Feature selection: In the first step, we use single-layer perceptrons (SLPs) to approximate the preferences of the players; Sequential Forward Selection (SFS) [35],

[36] is applied to generate the input vector for the SLPs by finding the subset of features that yields the highest performance. The quality of a feature subset is determined by 3-fold cross-validation on unseen data.

- 2) Feature space expansion: The subset of features derived from SFS using SLP is then used as the input of small multi-layer perceptron (MLP) models (containing one layer of two hidden neurons) and SFS is used again to extract additional features from the set of remaining features allowing features with more complicated non-linear relationships to be selected.
- 3) Setting ANN topology: Once all features that contribute to accurate simple MLP models are found we optimise the topology of models using neuroevolutionary preference learning. We start with a simple MLP topology of one hidden layer of two neurons, we then increase the number of neurons up to ten by adding two neurons at each step. Further, we investigate MLPs with two hidden layers, with up to ten neurons in the first and second layer. Again, the number of hidden neurons starts at two and increases by adding two neurons at each step; this sums to 30 different MLPs topologies which are tested for each input vector.

The performance of each MLP is obtained through the average classification accuracy in three independent runs using 3-fold cross validation. Parameter tuning tests have been conducted to set up the parameters' values for neuroevolutionary user preference learning that yield the highest accuracy and minimise computational effort. A population of 100 individuals is used, and evolution runs for 20 generations. A probabilistic rank-based selection scheme is used, with higher ranked individuals having higher probability of being chosen as parents. Finally, reproduction was performed via uniform crossover, followed by Gaussian mutation of 1% probability.

#### VIII. ANALYSIS

This section provides a thorough analysis conducted for testing simple and more complex relationships between the features extracted and the three reported states of player experience. We further investigate the generality of the proposed approach by comparing the models constructed on the presented dataset and the ones constructed in our previous work in terms of the models' performance and the features selected.

##### A. Linear Relationships

We performed an analysis for exploring statistically significant correlations (p-value < 5%) between player's expressed preferences and extracted features. Correlation coefficients are obtained by following the method proposed in [17]. According to this method, correlation coefficients are calculated through  $c(z) = \sum_{i=0}^{N_s} \{z_i / N_s\}$  where  $N_s$  is the total number of game pairs where players expressed a clear preference ( $game_A > game_B$  or  $game_B > game_A$ ) for one of the

two games and  $z_i = 1$ , if the player preferred the game with the larger value of the examined feature and  $z_i = -1$ , if the player preferred the other game in the game pair  $i$ . The top five significantly correlated features for each emotional state are presented in Table V.

Nineteen direct features are significantly correlated with engagement with some of them also strongly correlated with frustration and challenge while 21 features are significantly correlated with frustration, and 17 features with challenge. The features that are strongly correlated with engagement and not with challenge are mostly related to the interaction between the player and blocks (mainly powerups); These features point to the task of searching for powerups, in which the player has to destroy blocks looking for powerups which as a result changes Mario’s mode, as being particularly engaging.

The avatar death feature (signifying that Mario loses a life) is the most significantly correlated with both frustration and challenge indicating a strong relationship between death and these two player experience states.

Regarding changes in platform height patterns,  $P$ , only the features presented in Table V for engagement and frustration are significantly correlated with engagement and frustration while 15 features are strongly correlated with challenge. Seven and 15 out of the features that correlated best with frustration and challenge, respectively, relate to the presence of a gap while engagement is significantly correlated with only four features that indicate a gap. It is interesting to note that despite the small patterns’ length (three) almost all features presented for the three emotional states require two or three gameplay actions to be performed.


Ten out of the 12 features from  $I$  (item and enemy placement) are significantly correlated with engagement while only three and two features correlate significantly with frustration and challenge, respectively. A first observation is that it is obviously much easier to predict engagement from  $I$  than to predict challenge and frustration due to many more features significantly correlated to engagement, and the correlations are stronger. Most features that correlate with engagement point to the placement of items and enemies. This is not the same for frustration which demonstrates less significant effects and the majority of those that do focus on the existence of an enemy; features that correlate with challenge highlight the importance of the relative placement of items and enemies in the challenged perceived.

Large subsets of features of players’ actions ( $A$ ) are significantly correlated with engagement, frustration and challenge (99, 72 and 74, respectively). All features that are highly correlated to frustration are also correlated to challenge. It is worth mentioning that the features that correlate the most with engagement are also significantly correlated with frustration and challenge but at different significance levels. It appears that the number of jumps the player performs plays an important role in predicting engagement as it appears in all top-5 action patterns combined in most of the cases with moving right and pressing the speed button. This

can also explain the significance correlation found between engagement and sequences of  $I$  that contains items which mostly require jumping to be collected and enemies which require a jump to be killed or overcome.

While jumping and moving right are the most important actions for predicting engagement, standing still,  $S$  (supposedly thinking about how to overcome the next obstacle) is the most frequent action in the subset of features correlated with frustration and challenge

Nine features out of the 25 features of  $C_g$  are significantly correlated with engagement, while only three features are strongly correlated with challenge and frustration. It’s worth noticing that all the features that correlate with challenge contain the same items and differ only in the placement of parentheses (the same applies for frustration). This indicates that the existence or non-existence of a sequence of certain content items is more important for the experienced frustration and challenge than its relative placement.

The three correlated  $C_g$  features with frustration linked to the existence of gaps and the placement of parentheses within each pattern reflect the width of a gap (a gap beginning and ending within the same item indicates a small gap width since the parentheses enclose events happening within a very short time). The fact that the significant patterns contain  in combination with a gap points out to stairs surrounding the gap or changes in platform height within a very close distance to the gap which add to the difficulty of jumping and as a result, on the reported frustration. Somewhat surprisingly, patterns including the presence of gaps are not correlated with challenge. This suggests a possible nonlinear relationship.

Large subsets of multimodal features are strongly correlated with engagement, frustration and challenge (232, 95 and 119, respectively). While most features correlated with frustration are also strongly correlated with challenge, the most significantly correlated features with engagement, are not strongly correlated with either frustration or challenge. Patterns correlated with engagement draw a picture of most players enjoying running in a non-flat platform that requires jumping. From the patterns correlated with frustration, it seems that frustrated players spend more time standing still, less time running through the level (this can also be seen in  $A$  and  $A_{0.25}$  patterns where the standing still and moving right — without the speed button pressed — are the most dominant actions).

The correlations calculated and analysed above provide basic analysis with linear relationships between the extracted features and reported emotions. However, these relationships are most likely more complex than those that can be captured by linear models. The aim of the following section is to analyse the nonlinear relationships found using the players’ experience models.

## B. Nonlinear Relationships

In this section, we base our analysis on which features were selected by the SFS algorithm for constructing neural

TABLE V

TOP FIVE STATISTICALLY SIGNIFICANT CORRELATION COEFFICIENTS BETWEEN REPORTED ENGAGEMENT, FRUSTRATION AND CHALLENGE AND EXTRACTED FEATURES. THE SIGN BEFORE THE FEATURES INDICATES POSITIVE (+) OR NEGATIVE (-) CORRELATION. FOR EXAMPLE, THE TIME REQUIRED TO COMPLETE THE LEVEL WAS FOUND TO BE POSITIVELY CORRELATED WITH ENGAGEMENT, WHILE A SEGMENT OF CONTENT WITH TWO ADJACENT DECREASES IN THE PLATFORM HEIGHT WAS FOUND TO BE NEGATIVELY CORRELATED WITH ENGAGEMENT AS CAN BE SEEN FROM THE FIRST ROW.

Direct	$P$	$I$	$A$	Sequential $A_{0.25sec}$	$C_g$	$M$
Engagement						
$+t_{comp}$		-	$+(S)(\blacktriangleright)(\uparrow)$	$-(\blacktriangleright)(\blacktriangleright)(S)$	$+(0, 0, \uparrow)$	$+(\uparrow)(\blacktriangleright)(\uparrow)$
$+t_{play}$		+	$-(R^{\uparrow})(\blacktriangleright)(\blacktriangleright)$	$-(\blacktriangleright)(\uparrow)(S)$	$+(0, \uparrow)(0)$	$+(\uparrow)(\blacktriangleright)(\uparrow)$
$+n_{powerups}$		+000	$-(\uparrow)(S)(S)$	$-(\blacktriangleright, \uparrow)(\uparrow)$	$+(0, 0)(\uparrow)$	$+(\uparrow)(\uparrow, \uparrow)$
$+n_{state}$		+	$-(R^{\uparrow})(\blacktriangleright)(R^{\uparrow})$	$-(\blacktriangleright, \uparrow)(S)$	$+(\uparrow, \uparrow)(\uparrow)$	$+(\uparrow)(\uparrow)(\uparrow)$
$+N_w$	-	+	$-(\blacktriangleright)(\uparrow)(\blacktriangleright)$	$+(\blacktriangleright, \uparrow, S)$	$+(\uparrow, \uparrow)(\uparrow)$	$+(\uparrow)(\uparrow)(\uparrow)$
Frustration						
$+d_{cause}$	-	+	$+(S)(\blacktriangleright)(\uparrow)$	$+(\blacktriangleright, \blacktriangleright)(S)$	$-(\uparrow, \uparrow)(\uparrow)$	$+(0)(\blacktriangleright)(S)$
$-n_{coin}$	+	+	$-(R^{\uparrow})(\blacktriangleright)(\blacktriangleright)$	$-(\blacktriangleright)(\uparrow)(S)$	$-(\uparrow, \uparrow)(\uparrow)$	$+(\blacktriangleright)(\blacktriangleright)(0)$
$+d_{total}$	+	+	$-(\uparrow)(S)(S)$	$-(\blacktriangleright, S)(\uparrow)$	$-(\uparrow)(\uparrow)(\uparrow)$	$+(0)(0)(\blacktriangleright)$
$-n_{jump}$	+	+	$-(R^{\uparrow})(\blacktriangleright)(R^{\uparrow})$	$+(\blacktriangleright)(\blacktriangleright)(S)$		$+(S)(0)(S)$
$-t_{play}$	+		$-(\blacktriangleright)(\uparrow)(\blacktriangleright)$	$+(\blacktriangleright)(\uparrow)(S)$		$+(0)(\blacktriangleright)(\uparrow)$
Challenge						
$+d_{total}$	-	+	$-(S)(\blacktriangleright)(\uparrow)$	$-(\blacktriangleright)(\blacktriangleright)(\uparrow)$	$+(0, \uparrow)(\uparrow)$	$-(S)(\blacktriangleright)(0)$
$+d_{cause}$	+	+	$-(R^{\uparrow})(\blacktriangleright)(\blacktriangleright)$	$+(S)(\uparrow, S)$	$+(\uparrow)(0, \uparrow)$	$-(0, \blacktriangleright, R^{\uparrow})$
$-k_{cannon}$	-		$-(\uparrow)(S)(S)$	$-(S, S)(S)$	$+(0, \uparrow)(\uparrow)$	$-(\blacktriangleright)(0)(S)$
$-t_{right}$	+		$-(R^{\uparrow})(\blacktriangleright)(R^{\uparrow})$	$-(\blacktriangleright)(\blacktriangleright, \blacktriangleright)$		$-(\blacktriangleright)(0)(\blacktriangleright)$
$-n_{state}$	+		$-(\blacktriangleright)(\uparrow)(\blacktriangleright)$	$-(\blacktriangleright, \uparrow)(\uparrow)$		$-(\blacktriangleright)(0)(\uparrow)$

network-based player experience models. As these models take nonlinear relations into account, the features selected for these models might reveal more complicated and in a sense deeper relationships, but the analysis is also less straightforward.

All direct features and the number of occurrences of all sequential features extracted are uniformly normalised to [0,1] using standard max-min normalisation. After normalisation, these values are used as inputs for feature selection and ANN model optimisation. Table VI presents the features selected for reported engagement, frustration and challenge, respectively.

Note that to design Infinite Mario level generation mechanisms that are driven by the player experience models we construct here, all remaining controllable features that are not selected in the feature selection process are forced into the input of the MLPs. The MLP performance and topologies of the best MLPs (for both direct and various types of sequential features) are presented in Table VII.

1) *Engagement*: Using MLPs with the selected direct features and the remaining controllable features, we were able to predict engagement, frustration and challenge with relatively high accuracy (see Table VII). Out of the three emotional states, engagement appears to be the hardest to predict both in terms of network topology and model's performance.

Using different patterns of content and/or gameplay to construct player experience models resulted in models that vary in topology and performance. The best-performing model for predicting engagement has been constructed using selected patterns of players' gameplay taken every 0.25 seconds and

achieved a performance that is significantly better than all other models (83.8%) followed by the model constructed on patterns extracted from items and enemies placement,  $I$  (71.02%) with no significant difference from the model constructed on direct features. It is interesting to note that this model outperforms other models after including the direct controllable features in the inputs. Without including the controllable features, the model constructed on direct features outperforms the ones constructed on sequential features with no significant difference from the model constructed on patterns extracted from players' gameplay,  $A$ .

The subset of direct features for predicting engagement (see Table VI) consists of the total time spent playing the game, the time spent doing different activities (running, jumping, in big mode and in little mode), the number of coins collected, the number of blocks destroyed (which in part relates to the number of collected coins since player smashes blocks to collect hidden coins and it also relates to the time spent in big/small mode), the number of times the jump button is pressed (which relates to the time spent jumping), the cause of death, and the controllable feature that defines the number of goombas and koopas scattered around the level.

Two of the directed features selected appear to be dominant in the selected sequential features of players' actions,  $A_{0.25}$ , more specially, running right and jumping. The two selected patterns  $(\blacktriangleright, \uparrow, S)$  and  $(\blacktriangleright)(\uparrow, S)$  point out to the existence of a content event that causes jumping and standing still behaviours. This can be better explained by looking at the selected content patterns that relates to gameplay events,  $C_g$ . By investigating the subset of selected features from

TABLE VI  
THE FEATURES SELECTED FROM THE SET OF DIRECT AND SEQUENTIAL FEATURES FOR PREDICTING ENGAGEMENT, FRUSTRATION AND CHALLENGE USING SEQUENTIAL FEATURE SELECTION WITH SLP AND SIMPLE MLP MODELS

		Direct	$P$	$I$	$A$	Sequential $A_{0.25sec}$	$C_g$	$M$
Engagement								
$SFS_{slp}$	$t_{comp}$		000		(▶)(▶)(▶)	(▶)(▶)(▶)	(○)(○, ○)	(▶)(○)(○)
	$n_{coin}$				$(R^{\blacktriangleright})(R^{\blacktriangleright\uparrow})(R^{\blacktriangleright})$	$(\blacktriangleright, \uparrow^{\blacktriangleright}, S)$		$(R^{\blacktriangleright})(R^{\blacktriangleright\uparrow})(\blacksquare)$
	$d_{cause}$				$(\uparrow^{\blacktriangleright})(\uparrow^{\blacktriangleright})(S)$	$(R^{\blacktriangleright}, R^{\blacktriangleright}, R^{\blacktriangleright\uparrow})$		$(S)(\blacktriangleright)(\circ)$
	$t_{small}$				$(R^{\blacktriangleright\uparrow}, \uparrow^{\blacktriangleright})(S)$	$(\blacktriangleright)(\uparrow^{\blacktriangleright}, S)$		
	$E$				$(\blacktriangleright)(\uparrow^{\blacktriangleright})(S)$	$(S)(\blacktriangleright)(S)$		$(R^{\blacktriangleright})(\circ)(R^{\blacktriangleright})$
	$t_{jump}$				$(R^{\blacktriangleright})(S)(S)$	$(\blacktriangleright, R^{\blacktriangleright}, R^{\blacktriangleright\uparrow})$		
	$n_{coinBlock}$				$(\uparrow^{\blacktriangleright})(\uparrow^{\blacktriangleright})(\blacktriangleright)$	$(\blacktriangleright)(S)(\uparrow^{\blacktriangleright})$		$(\circ)(R^{\blacktriangleright})(R^{\blacktriangleright\uparrow})$
$SFS_{mlp}$	$t_{big}$				$(\blacktriangleright, \uparrow^{\blacktriangleright})(\blacktriangleleft)$			
	$t_{run}$							
	$n_{jump}$							
Frustration								
$SFS_{slp}$	$t_{right}$		000		$(S)(\blacktriangleright)(S)$	$(\blacktriangleright)(\blacktriangleright)(\blacktriangleright)$	$(\circ)(\circ, \blacksquare)$	$(\blacktriangleright)(\blacksquare)(\circ)$
	$d_{total}$				$(R^{\blacktriangleright\uparrow})(R^{\blacktriangleright\uparrow})(R^{\blacktriangleright})$	$(R^{\blacktriangleright}, R^{\blacktriangleright\uparrow})(R^{\blacktriangleright})$	$(\circ, \circ, \circ)$	
	$d_{cause}$				$(\uparrow^{\blacktriangleright})(S)(\blacktriangleright)$	$(S)(\blacktriangleright, S)$		$(\blacktriangleright)(S)(\blacksquare)$
	$k_{goomba}$				$(\uparrow^{\blacktriangleright})(\blacktriangleright)(\blacktriangleright)$	$(S)(\blacktriangleright)(\blacktriangleright)$		$(\circ, R^{\blacktriangleright})(\blacktriangleright)$
	$t_{play}$				$(\blacktriangleright)(\uparrow^{\blacktriangleright})(\blacktriangleleft)$	$(\blacktriangleright)(\uparrow^{\blacktriangleright})(\blacktriangleleft)$		
	$\bar{G}_w$				$(\blacktriangleright)(S)(\blacktriangleright)$	$(\blacktriangleright)(S)(\blacktriangleright)$	$(\circ, \circ)(\blacksquare)$	$(R^{\blacktriangleright})(\blacktriangleright)(\blacktriangleright)$
$SFS_{mlp}$	$G$				$(\blacktriangleleft, \uparrow^{\blacktriangleright})(\blacktriangleright)$	$(\uparrow^{\blacktriangleright}, S)(\blacktriangleright)$		
	$n_{jump}$					$(\blacktriangleright)(S)(S)$		
Challenge								
$SFS_{slp}$	$t_{play}$		000		$(\blacktriangleright)(\blacktriangleright)(S)$	$(\blacktriangleright, \blacktriangleright)(\blacktriangleright)$	$(\circ)(\circ, \circ)$	$(\blacktriangleright)(\circ)(\blacksquare)$
	$n_{jump}$				$(\blacktriangleright)(R^{\blacktriangleright})(R^{\blacktriangleright})$	$(\blacktriangleright, R^{\blacktriangleright})(R^{\blacktriangleright})$		
	$d_{total}$				$(\uparrow^{\blacktriangleright})(\blacktriangleright)(S)$	$(R^{\blacktriangleright})(R^{\blacktriangleright\uparrow})(R^{\blacktriangleright})$		$(\blacktriangleright)(\uparrow^{\blacktriangleright})(S)$
	$n_{coin}$				$(\uparrow^{\blacktriangleright})(\uparrow^{\blacktriangleright})(S)$	$(S)(\blacktriangleright)(\uparrow^{\blacktriangleright})$	$(\circ, \circ, \circ)$	$(S)(\blacktriangleright)(\blacksquare)$
	$t_{right}$				$(\blacktriangleright)(S)(S)$	$(\blacktriangleright)(S)(S)$		$(\circ)(\circ)(R^{\blacktriangleright})$
	$\bar{G}_w$							$(\blacktriangleright, \blacktriangleright)(\blacksquare)$
	$E_p$							
$SFS_{mlp}$	$t_{left}$					$(\blacktriangleright)(\blacktriangleright)(S)$		
	$k_{stomp}$							

TABLE VII  
BEST MLP TOPOLOGIES AND CORRESPONDING PERFORMANCE ON DIRECT AND SEQUENTIAL FEATURES. THE PERFORMANCE OF MLP MODELS BUILT ON THE SUBSET OF SELECTED FEATURES,  $MLP_s$  IS COMPARED AGAINST THE MODELS BUILT ON SELECTED AND FORCED CONTROLLABLE FEATURES,  $MLP_c$ . THE TOPOLOGIES ARE PRESENTED IN THE FORM: NUMBER OF INPUTS-NUMBER OF NEURONS IN THE FIRST HIDDEN LAYER-NUMBER OF NEURONS IN THE SECOND HIDDEN LAYER.

		Direct	Sequential					
			$P$	$I$	$A$	$A_{0.25sec}$	$C_g$	$M$
Engagement	$MLP_{topology}$	15-6-8	14-8-4	13-2-6	14-2-2	14-10-0	11-2-2	14-2-2
	$MLP_s$	73.50%	68.84%	72.19%	73.19%	66.85%	67.16%	63.81%
	$MLP_c$	69.80%	65.80%	71.02%	68.00%	83.80%	68.00%	66.49%
Frustration	$MLP_{topology}$	12-6-0	12-8-2	10-10-10	13-2-0	12-2-0	13-6-0	14-4-0
	$MLP_s$	83.00%	77.21%	66.66%	68.92%	68.45%	66.29%	72.88%
	$MLP_c$	80.70%	71.93%	69.30%	72.50%	71.37%	68.36%	72.32%
Challenge	$MLP_{topology}$	13-2-2	10-4-0	10-2-8	10-8-6	12-4-4	13-8-6	12-6-2
	$MLP_s$	79.10%	73.04%	69.22%	63.84%	62.83%	66.50%	68.88%
	$MLP_c$	77.50%	69.60%	69.05%	70.62%	67.62%	71.29%	71.45%

these two types together, simple jumping actions,  $\uparrow$ , can be explained by changes in platform height and placement of items; moving right followed by jumping and standing still patterns ( $(\blacktriangleright, \uparrow, S)$  and  $(\blacktriangleright)(\uparrow^{\blacktriangleright}, S)$ ) mostly relates to the behaviour of overcoming enemies ( , , ); the more complex navigation patterns that has been selected, such as  $(R^{\blacktriangleright}, R^{\blacktriangleright}, R^{\blacktriangleright\uparrow})$  that defines the behaviour of pressing a

combination of buttons at the same time within a very small window time, suggest the existence of a gap that requires speeding up followed by jumping while the moving right and the speed button are still pressed ( , , ). Note that the two patterns ( $(\blacktriangleright, \uparrow, S)$  and  $(\blacktriangleright)(\uparrow^{\blacktriangleright}, S)$ ) can also be the result of overcoming a gap, which in that case reflect a beginner player playing style. On the contrary the pattern



$(R^{\blacktriangleright}, R^{\blacktriangleright}, R^{\blacktriangleright\uparrow})$  captures a more advanced playing behaviour. Since the methodology proposed constructs average models in the sense that the models are trained on a composite of subjective preferences of several subjects, patterns that capture the playing style of beginner and expert players can be selected and presented as inputs to the models.

2) *Frustration*: The best models for predicting frustration have been constructed using the subset of direct features and the remaining controllable features and they significantly outperform all other models. The models trained on patterns of players' actions achieve the highest performance among other sequential-based models (no significant difference however). Using the subset of selected features without enforcing the controllable features, the best performance obtained from the models constructed on frequent patterns of changing platform height; again, the model's performance is significantly lower than the performance of the models constructed on direct features.

From the features selected for predicting frustration (see Table VI), it appears that for a game to be frustrating, it should contain at least a certain number of gaps with certain width (both positively correlated). The number of kills of goombas and koopas points out to the importance of the number of enemies presented in the game. The selection of the features that relate to avatar death (the number of deaths, the cause of death and the time spent playing in last life) also reveals the importance of gaps and enemies since these two elements constitute major causes of death.

Selected sequential features highlight specific patterns that have an impact on reported frustration. As selected direct features already demonstrated, the existence of enemies and gaps seem to be important for predicting frustration since most of the sequential patterns of  $P$ ,  $I$  and  $C_g$  contain these events. The placement of stairs around gaps or the changes in platform height within a close distance to a gap appear to have an influence on how frustrating the game perceived even with moderate to small width gaps (e.g. the pattern  $(\blacksquare, \blacksquare, \blacksquare)$ ). Another element that factors in the perceived frustration is the placement of several game content events within a small time window as can be seen from the example patterns:  $(\text{🍄}, \text{🍄}, \text{🍄})$  and  $(\text{🍄}, \text{🍄})(\text{🏠})$ . It can be observed from the most frequent patterns of players' actions and the correlation between them and reported frustration that the frustrated player rapidly switches between simple actions of moving right (without speeding up), standing still and performing simple jumps.

3) *Challenge*: Challenge can be best predicted using a subset of direct features with significantly better performance than all other models constructed on sequential features. The models constructed on direct features also outperform the other models when excluding the controllable features. The best performing model from sequential features is based on multimodal patterns with a very close performance to the models constructed on patterns from players' actions,  $A$ , and the models constructed on patterns of game content,  $C_g$ .

The direct features selected for predicting challenge (see

Table VI) reveal the importance of gaps and enemies since five of them relate to gaps width, placement and killing of enemies and avatar death (all positively correlated). An interesting and somehow expected feature is enemy placement, which is negatively correlated with challenge and adds to the difficulty of the game, in particular, when enemies are placed around gaps making it more challenging to jump over and also when placed around blocks making item collection more difficult.

Selected direct features can be better explained when analysing the selected sequential patterns. The presence of the standing still item in the same pattern with moving right and/or jumping suggests an existence of a challenging situation in which the player has to pause and spend sometime thinking before taking a simple action (e.g. patterns like  $(\uparrow)(\blacktriangleright)(S)$  or  $(\blacktriangleright)(S)(\text{🏠})$ ). While challenge is positively correlated with the pattern  $(\blacksquare)(\blacksquare, \blacksquare)$ , a negative correlation has been observed between challenge and the pattern  $(\blacksquare, \blacksquare, \blacksquare)$ . This can be explained by the complex situation that arises in the first case and makes jumping over a gap more challenging since the player does not have enough space to speedup before jumping; instead she has to move carefully towards the edge and press a set of combined keys in order to reach the other edge.

One should expect that the models constructed on multimodal data of content and gameplay ( $M$ ) should achieve the best performance. Surprisingly, the performances obtained from these models are as high or slightly lower than the performance of the best sequential models constructed. A possible explanation is that frequent patterns of length three are rather small to capture patterns across different data streams and longer pattern lengths should be considered. (We would likely need more data in order to effectively use longer subsequences for analysis). Another critique is the wide diversity of players' actions when encountering the same in game situation which enlarges the size of the feature space and complicates the mining of the resulted sequences.

### C. Comparison with Player Experience Models in the Literature

Since player experience models based on direct features using the same methodology but with smaller dataset and longer game sessions has been constructed in our previous work [17], [19], it is worth comparing the models' accuracies and selected features for the three emotional states and investigate how well the methodology proposed scales for a much larger dataset and smaller game sessions. Note that in our previous work, the three emotional states investigated were fun, frustration and challenge. Even though not entirely accurate we assume that players' reported fun is consistent to the level of reported engagement for comparison purposes.

Table VIII presents the features selected, model topologies and prediction performance for the models presented in [19]. For engagement, three out of the four features selected in the previous models have also been selected in the current model along with seven other features. Despite the expansion



TABLE VIII

THE SUBSET OF DIRECT FEATURES USED FOR PREDICTING PLAYERS’ REPORTED EXPERIENCE AND THE CORRESPONDING MODELS’ TOPOLOGIES AND PERFORMANCE AS PRESENTED IN [19]

	<i>Fun</i>	<i>Frustration</i>	<i>Challenge</i>
	$t_{comp}$	$t_{comp}$	$n_{coin}$
	$t_{big}$	$F$	$t_{comp}$
	$t_{run}$	$n_{powerups}$	$n_f$
	$k_{unleash}$	$n_{mode}$	$F$
		$t_{play}$	$d_f$
		$\bar{G}_w$	$\bar{G}_w$
		$n_{jump}$	
$MLP_{top}$	7-10	10-4-2	9-3
$MLP_{perf}$	69.66%	89.33%	74.66%

in the dataset size and the use of smaller time sessions, the methodology proposed for constructing players’ experience models of engagement appears to be consistent since the two models are able to predict engagement with a relatively similar accuracy.

Three out of seven features selected for the frustration model in [19] are common for predicting frustration in this paper. Comparing the models performance indicates that frustration can be predicted with higher accuracy from the smaller dataset and longer session time. This can be in part explained by the difficulty in expressing a clear emotional preference of frustration on different short game variants since data collection resulted in 169 pairs of unclear preferences compared to 103 and 71 for engagement and challenge, respectively.

Only two features generalise for the two datasets for challenge, namely, the number of collected coins,  $n_{coin}$  and the average gaps width,  $\bar{G}_w$ . Some of the other features are somehow related, more specially, the time spent during last life,  $t_{play}$  correlate with the time needed to complete the level,  $t_{comp}$  and the number of death,  $d_{total}$  is a generalisation of the number of times the player killed because of a cannon bullet,  $d_f$ . Overall, despite the huge increase in the dataset size, challenge is predicted with larger subset of features and higher accuracy from shorter game sessions.

To check for the efficiency of the feature selection approach, the impact of the selected subset of features on the prediction accuracy, the influence of the size of the game session and the generality of the proposed methodology, we evaluated the previous models on the dataset used to construct the current models and vice versa. The obtained accuracies for the three player experience states are presented in Table IX. As can be seen from the table, the best performance is obtained when the old model evaluates challenge on the new dataset (67.25%) despite that these two models share only two features. Unsurprisingly, that cross-validation performance on challenge is lower than the two corresponding models constructed and evaluated on the same dataset. While reported frustration models are the most accurate for the two datasets — and although four features

TABLE IX

THE PERFORMANCE OF THE MODELS OF [19] ON THE NEW DATASET ( $P_{old/new}$ ) COMPARED TO THE PERFORMANCE OF THE NEW MODELS ON THE DATASET OF [19] ( $P_{new/old}$ )

	Engagement (Fun)	Frustration	Challenge
$P_{old/new}$	58.98%	40.68%	67.25%
$P_{new/old}$	57.33%	58.18%	45.36%

are found in common between these two models — none of them managed to generalise well when evaluated on the unseen dataset. The two models for predicting reported engagement achieved similar results when evaluated on the unseen dataset. In summary, it appears that longer game sessions are more relevant for predicting frustration while challenge can be predicted better from short game sessions.

## IX. DISCUSSION

The computational aesthetics approach presented in this paper is based on several short Infinite Mario Bros game levels played over the Internet in a crowdsourcing user survey that yielded a large dataset of 780 pairs of played and annotated (self-reported) games. Direct and sequential features describing game content and players’ in-game behaviour have been extracted and were used for the analysis of the relationship between the content of the game, the players’ playing style and the reported experience of three different states of player experience. Data mining techniques have been implemented to extract useful patterns from sequential features and sequential forward feature selection has been employed to extract a subset of features that have predictive capabilities with respect to reported player experience. Based on the selected feature subsets, highly accurate models of player experience have been constructed and used for an in-depth analysis of the factors that contribute to player experience, and thereby aesthetics, in platform games.

The thorough analysis followed shows some generic aspects of level design aesthetics that relate to the three reported emotional states: engagement, frustration and challenge. Overall, an engaging Infinite Mario level is the one that provides enough space for running, changes in platform height, items to be collected as well as it contains challenging elements presented in the placement of enemies around collectable items, the existence of gaps and the placement of not easily collectable items. It also appears that the level of challenge should match the player’s level of expertise for the game to be engaging for a particular player.

The number of gaps and their average width play major roles in perceived frustration. The more gaps and the wider they are, the more frustrating the game is specially when the gaps are combined with changes in platform height. The number of enemies has less direct influence on frustration. It is interesting to note that frustration can be predicted up to a good degree just by the changes in platform height; this can be the result of more player concentration required when height changes rapidly leading to frequent changes of

performed actions. It appears that, in general, the placement of a sequence of items after each other within a small distance leads to a more frustrating game as it most likely increases the level of player confusion, cognitive load and level complexity.

Challenge appears to be affected more by the characteristics of particular features rather than the frequency of their appearance in the level: the width of gaps, the placement of stairs around them, the placement of enemies, the frequent changes in platform height, and the placement of items within a small distance to each other contribute to a more challenging game as they imply a higher probability of game failure [37]. It would be interesting to validate the methodology proposed and its findings with designers' knowledge of what makes a level engaging, frustrating or challenging, and check to which extent these findings add to what we know about game design.

We venture that, as Super Mario Bros more or less defines the platform game genre, and as a clone of this game is used as a testbed game for this study, the results obtained can be applied, by extension, to Super Mario Bros in particular, and to the majority of platform games, to some extent. The generality of the selected patterns allow the use of them to design and analyse other levels with different graphical representation. The use of the selected patterns as the main building blocks for designing levels provide a promising alternative to other rhythm based approaches [12], [23] specially when the purpose is to alter a particular affective state of the player. Extending this study, and validating the methodology and the findings in other games from the same genre or from other genres constitute a future direction. The methodology proposed could potentially be used to find new design insights if used on a less-known game genre.

The approach presented provides the underlying basis for game adaptation techniques that could be employed to automatically generate game content that optimises particular aspects of player experience [16]. To this end, the use of the extracted patterns of players' actions and game content as controllable features — instead of item frequencies — constitutes a promising future direction. This also implies the use of more powerful search algorithm to find the optimal set of controllable features that will be used to generate the new personalised level.

The proposed approach and the analysis presented could also be used as an assistant tool in a mixed-initiative level design process [12]. A level can be crafted by a human designer and models constructed from game content and reported player experience could be used to encourage the designer to include or modify features or patterns based on the experience the designer wishes to provide.

Although a thorough analysis has been conducted, the conclusions drawn are rather general leaving plenty of rooms for further investigations.

For the experiments presented in this paper we used only sequences of length three that are rather small to draw general conclusions. Frequent sequences of longer length have been

investigated; although these sequences are more expressive, a performance drop has been observed using these sequences. Longer sequences tend to capture more specific patterns across multiple modalities of player input in which we expect larger data variation due to variant playing styles. A solution might be to cluster the resulting sequences and construct models for each cluster, or consider sequences of different length as inputs to ANN models. A step towards clustering players' behaviour based on sequence patterns of actions has already been taken and preliminary results indicate the promise of the approach. One could also investigate the use of other sequence mining techniques such as Hidden Markov Model to classify the resulted sequences and to extract sequential pattern.

The performance increase obtained, in some cases, when combining the controllable features with the selected sequential features suggest that models of higher performance could be constructed by presenting the direct features and the sequential features as inputs to SFS. It is also worth investigating whether including features from different sequences type and different pattern length would have a positive impact on model's accuracy. Doing so, however, would expand the feature space and more efficient feature selection methods will most likely be required.

## X. CONCLUSION

This paper presents an computational, data-driven, approach for a thorough analysis of aesthetics in games via the investigation of the relationship between game content, players' playing style and reported player experience (engagement, frustration and challenge) in the Infinite Mario Bros game. The approach is based on large sets of crowd-sourced gameplay data and annotated data of player experience via self-reported (pairwise) ranks. Direct and sequential features of content and gameplay were explored and sequence mining techniques were implemented to extract useful game environment and player's behavioural patterns. The features have been analysed in terms of their linear and nonlinear relationship to each reported state of player experience and revealed a wealth of interconnections among them. Furthermore, neuroevolutionary preference learning was used to construct player experience models of high accuracies based on dissimilar types of extracted features. Using the proposed approach we are able to draw general conclusions about the interaction between the player and the game and mine patterns of level content — that yield sequences of players' actions — and their corresponding effect on player experience and game aesthetics. The methodology proposed and the findings can be potentially applied to other less well-known games from the same genre or to other game genres.

## ACKNOWLEDGMENTS

The authors would like to thank all subjects that participated in the experiments. The research was supported, in part, by the Danish Research Agency, Ministry of Science, Technology and Innovation: project "AGameComIn" (274-09-0083).

## REFERENCES

- [1] T. Malone, "What makes computer games fun? (abstract only)," in *Proceedings of the joint conference on Easier and more productive use of computer systems. (Part - II): Human interface and the user interface - Volume 1981*, ser. CHI '81. New York, NY, USA: ACM, 1981, p. 143.
- [2] R. Koster, *A theory of fun for game design*. Paraglyph press, 2004.
- [3] B. Magerko, C. Heeter, J. Fitzgerald, and B. Medler, "Intelligent adaptation of digital game-based learning," in *Proceedings of the 2008 Conference on Future Play: Research, Play, Share*. ACM, 2008, pp. 200–203.
- [4] G. Yannakakis and J. Hallam, "A generic approach for generating interesting interactive pac-man opponents," in *In Proceedings of the IEEE Symposium on Computational Intelligence and Games*, 2005, pp. 94–101.
- [5] J. Togelius and J. Schmidhuber, "An experiment in automatic game design," pp. 111–118, 2009.
- [6] S. Björk, S. Lundgren, and J. Holopainen, "Game design patterns," in *Proceedings of Level Up-1st International Digital Games Research Conference*. Citeseer, 2003.
- [7] K. Hullett and J. Whitehead, "Design patterns in fps levels," in *FDG '10: Proceedings of the Fifth International Conference on the Foundations of Digital Games*. New York, NY, USA: ACM, 2010, pp. 78–85.
- [8] D. Moura, M. el Nasr, and C. Shaw, "Visualizing and understanding players' behavior in video games: discovering patterns and supporting aggregation and comparison," in *ACM SIGGRAPH 2011 Game Papers*. ACM, 2011, p. 2.
- [9] D. Milam and M. El Nasr, "Design patterns to guide player movement in 3d games," in *Proceedings of the 5th ACM SIGGRAPH Symposium on Video Games*. ACM, 2010, pp. 37–42.
- [10] M. Jennings-Teats, G. Smith, and N. Wardrip-Fruin, "Polymorph: A model for dynamic level generation," in *Sixth Artificial Intelligence and Interactive Digital Entertainment Conference*, 2010.
- [11] G. Smith, M. Cha, and J. Whitehead, "A framework for analysis of 2d platformer levels," in *Sandbox '08: Proceedings of the 2008 ACM SIGGRAPH symposium on Video games*. New York, NY, USA: ACM, 2008, pp. 75–80.
- [12] G. Smith, J. Whitehead, and M. Mateas, "Tanagra: A mixed-initiative level design tool," in *Proceedings of the International Conference on the Foundations of Digital Games*, 2010.
- [13] L. D. Riek, M. O. Connor, and P. Robinson, "Guess what? a game for affective annotation of video using crowd sourcing," in *Proceedings Affective Computing and Intelligent Interaction*, 2011.
- [14] A. Drachen, A. Canossa, and G. N. Yannakakis, "Player Modeling using Self-Organization in Tomb Raider: Underworld," in *Proceedings of the IEEE Symposium on Computational Intelligence and Games*. Milan, Italy: IEEE, September 2009, pp. 1–8.
- [15] C. Thurau and C. Bauckhage, "Analyzing the Evolution of Social Groups in World of Warcraft," in *Proceedings of the 2010 IEEE Conference on Computational Intelligence and Games*. Copenhagen: IEEE, August 2010, pp. 170–177.
- [16] G. N. Yannakakis and J. Togelius, "Experience-Driven Procedural Content Generation," *IEEE Transactions on Affective Computing*, 2011.
- [17] C. Pedersen, J. Togelius, and G. N. Yannakakis, "Modeling player experience in super mario bros," in *CIG'09: Proceedings of the 5th international conference on Computational Intelligence and Games*. Piscataway, NJ, USA: IEEE Press, 2009, pp. 132–139.
- [18] —, "Modeling player experience for content creation," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 2, no. 1, pp. 54–67, 2010.
- [19] N. Shaker, G. N. Yannakakis, and J. Togelius, "Towards Automatic Personalized Content Generation for Platform Games," in *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*. AAAI Press, October 2010.
- [20] —, "Feature Analysis for Modeling Game Content Quality," in *IEEE Transactions on Computational Intelligence and AI in Games (CIG)*, 2011.
- [21] P. A. Mawhorter and M. Mateas, "Procedural level generation using occupancy-regulated extension," in *Proceedings of the IEEE Conference on Computational Intelligence and Games (CIG)*, 2010, pp. 351–358.
- [22] N. Sorenson and P. Pasquier, "Towards a generic framework for automated video game level creation," in *Proceedings of the European Conference on Applications of Evolutionary Computation (EvoApplications)*, vol. 6024. Springer LNCS, 2010, pp. 130–139.
- [23] N. Sorenson, P. Pasquier, and S. DiPaola, "A generic approach to challenge modeling for the procedural creation of video game levels," 2011.
- [24] D. Perez, M. Nicolau, M. O'Neill, and A. Brabazon, "Evolving behaviour trees for the mario ai competition using grammatical evolution," in *Applications of Evolutionary Computation*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2011, vol. 6624, pp. 123–132.
- [25] S. Bojarski and C. Congdon, "Realm: A rule-based evolutionary computation agent that learns to play mario," in *Computational Intelligence and Games (CIG), 2010 IEEE Symposium*, 2010, pp. 83–90.
- [26] N. Shaker, J. Togelius, G. N. Yannakakis, B. Weber, T. Shimizu, T. Hashiyama, N. Sorenson, P. Pasquier, P. Mawhorter, G. Takahashi, G. Smith, and R. Baumgarten, "The 2010 Mario AI championship: Level generation track," *IEEE Transactions on Computational Intelligence and Games*, 2011.
- [27] C. Pedersen, J. Togelius, and G. N. Yannakakis, "Modeling player experience for content creation," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 2, no. 1, pp. 54–67, 2010.
- [28] G. Yannakakis and J. Hallam, "Entertainment modeling in physical play through physiology beyond heart-rate," *Affective Computing and Intelligent Interaction*, pp. 254–265, 2007.
- [29] G. N. Yannakakis, M. Maragoudakis, and J. Hallam, "Preference learning for cognitive modeling: a case study on entertainment preferences," *Trans. Sys. Man Cyber. Part A*, vol. 39, pp. 1165–1175, November 2009. [Online]. Available: <http://dx.doi.org/10.1109/TSMCA.2009.2028152>
- [30] K. Höök, "Affective loop experiences - what are they?" in *PERSUA-SIVE*, ser. Lecture Notes in Computer Science, vol. 5033. Springer, 2008, pp. 1–12.
- [31] M. Li, X. Chen, X. Li, B. Ma, and P. Vitányi, "The similarity metric," *IEEE Transactions on Information Theory*, vol. 50, no. 12, pp. 3250–3264, 2004.
- [32] M. J. Zaki, "Spade: An efficient algorithm for mining frequent sequences," *Mach. Learn.*, vol. 42, pp. 31–60, January 2001. [Online]. Available: <http://portal.acm.org/citation.cfm?id=599609.599626>
- [33] R. Srikant and R. Agrawal, "Mining sequential patterns: Generalizations and performance improvements," *Advances in Database TechnologyEDBT'96*, pp. 1–17, 1996.
- [34] H. Martinez and G. Yannakakis, "Mining multimodal sequential patterns: A case study on affect detection," in *Proceedings of the 13th International Conference in Multimodal Interaction, ICMI 2011, Alicante*. ACM Press, November 2011.
- [35] G. N. Yannakakis and J. Hallam, "Entertainment modeling through physiology in physical play," *Int. J. Hum.-Comput. Stud.*, vol. 66, pp. 741–755, October 2008. [Online]. Available: <http://dl.acm.org/citation.cfm?id=1410473.1410682>
- [36] G. N. Yannakakis, M. Maragoudakis, and J. Hallam, "Preference learning for cognitive modeling: a case study on entertainment preferences," *Trans. Sys. Man Cyber. Part A*, vol. 39, pp. 1165–1175, November 2009.
- [37] V. Nicollet, *Difficulty in dexterity-based platform games*. [Online]. Available: <http://www.gamedev.net/reference/design/features/platformdiff>, Mar. 2004.



**Noor Shaker** is a Ph.D. candidate at the IT University of Copenhagen. She received a 5-year BA in IT Engineering in 2007 from Damascus University, and an M.Sc. in Artificial Intelligence in 2009 from Katholieke Universiteit Leuven. Her research interests include player modeling, procedural content generation, affective computing and player behavior imitation.



**Georgios N. Yannakakis** (S'04; M'05) is an Associate Professor at the University of Malta (UoM). He received the Ph.D. degree in Informatics from the University of Edinburgh in 2005. Prior to joining the Department of Digital Games, UoM, in 2012 he was an Associate Professor at (and still being affiliated with) the Center for Computer Games Research at the IT University of Copenhagen.

He does research at the crossroads of AI (computational intelligence, preference learning), affective computing (emotion detection, emotion annotation), advanced game technology (player experience modeling, procedural content generation, personalisation) and human-computer interaction (multimodal interaction, psychophysiology, user modeling). He has published over 100 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 CIS Task Force on Player Satisfaction Modeling.



**Julian Togelius** is Associate Professor at the Center for Computer Games Research, IT University of Copenhagen, Denmark. He works on all aspects of computational intelligence and games, on geometric generalization of stochastic search algorithms and on evolutionary reinforcement learning. His current main research directions involve search-based procedural content generation in games, game adaptation through player modelling, automatic game design, and fair and relevant benchmarking of game AI through competitions.

He is also the chair of the IEEE CIS Technical Committee on Games, and an associate editor of IEEE Transactions on Computational Intelligence and Games. Togelius holds a BA from Lund University, an MSc from the University of Sussex, and a PhD from the University of Essex.