# EmoSTL: Formal Spatial-Temporal Verification of Emotion Specifications in Computer Games

Saba Gholizadeh Ansari*, I. S. W. B. Prasetya*, Mehdi Dastani*, Frank Dignum†, and Gabriele Keller*
**Utrecht University, Utrecht, The Netherlands*, {s.gholizadehansari, s.w.b.prasetya, m.m.dastani, g.k.keller}@uu.nl
†*Umeå University, Umeå, Sweden*, frank.dignum@umu.se

***Abstract*—As the game industry continues to evolve in popularity, testing the experience of players becomes crucial for attracting and retaining players in the highly competitive market. However, the absence of automated methods for articulating and verifying player experience (PX) specifications led us to introduce EmoSTL, a specialized language that extends Linear Temporal Logic with spatial and time-interval expressions, enabling the capture of complex temporal and spatial aspects of players' emotions and their experiences within games. We conducted a user study to collect suggestive PX requirements for a game under test to assess the capabilities of EmoSTL. Findings reveal that the language formalizes 92 percent of the set PX requirements, and with runtime verification, several PX design issues are identified in the game. Moreoever, EmoSTL performance evaluation demonstrates its linear execution time, showcasing the language potential usage in automated PX testing of games.**

***Index Terms*—formal verification, player experience testing, emotional experience, playtesting**

**Note:** This is a preprint of a paper with the same title, accepted in the Proceedings of the 17th IEEE International Conference on Software Testing, Verification and Validation (ICST), 2024.

## I. Introduction

Game User Research (GUR) [1] is a rapidly growing phenomenon that focuses on understanding the experience of players and their interaction with games. Such knowledge plays an essential role in the acquisition and retention of players in the competitive environment of the game industry. GUR involves analysis of psychology and human factors such as emotions that regulate the general experience of players [2]. This ultimately helps game designers to create engaging and immersive experiences. While maintaining positive emotions is connected to factors such as enjoyment and fun [3], [4], negatively valenced emotions contribute to engaging player experiences (PX) and facilitate player involvement [5], [6].

Turning this into a testing problem with verifiable emotion specifications would provide assistance in the automated quality control process by identifying PX-related flaws of the game design such as repetitive game-play, and eventually improving the overall game experience. The potential application of players' emotions verification during the game-play extends into broader domains such as a better human-computer interaction design [2] and game AIs development which let the AIs adapt and respond more naturally and believably to human players.

Formal verification [7] is a correctness technique that uses a mathematical interpretation to define and check properties, making it more precise and robust than non-formal techniques. Moreover, using formulas, rather than concrete values, allows us to formulate the specifications generally and consistently. Linear temporal logic (LTL) is a formalism technique that is introduced for specifying properties of systems that evolve over time [8], [9]. The language is applicable to design-time and run-time verification. While the former employs model-checking, the latter checks whether the system's current actions are aligned with the desired behavior [10]. Design-time verification techniques such as model-checking enable errors to be detected at an early development stage. However, obtaining reasonable models for real-life systems is often not feasible. On the other hand, run-time verification allows the detection of inconsistencies in the system's actual execution that may remain undetected by model-based verification [11]. The use of formal verification has become pervasive for assessing the conformance of software systems to their specification. However, the dominant focus of this methodology is on the functional correctness rather than utilizing a formal approach to verify the desired user experience, whereas for systems such as computer games the latter is just as important. In particular, testing gameplay is more than verifying the functional aspects of games; it is also about the evaluation of PX to help game developers address PX-issues during development and reduce the likelihood of costly and time-consuming post-deployment fixes [12]–[14] to deliver a well-received product.

**Contribution I.** This paper proposes a Domain Specific Language (DSL) named EmoSTL based on a subset of LTL and extended with spatial and time-interval expressions to express PX specifications of computer games with a semantic that allows run-time verification of specifications written in this DSL. In some aspects, the language offers more expressiveness than LTL, as it captures complex temporal and spatial properties of emotions, as well as aggregates of such properties over time or over a specific time interval. Furthermore, the language provides the capability for spatial verification, which targets a specific logic in a defined region, allowing verification of properties such as the coverage ratio of a certain emotion in a given area. Given the challenge of understanding LTL formulas, we strive to propose a more intuitive language for game designers. The proposed language would enable game developers to perform automated testing of emotion specifications.

**Contribution II.** Moreover, we conduct a user study to gather PX requirements designed by software engineers for

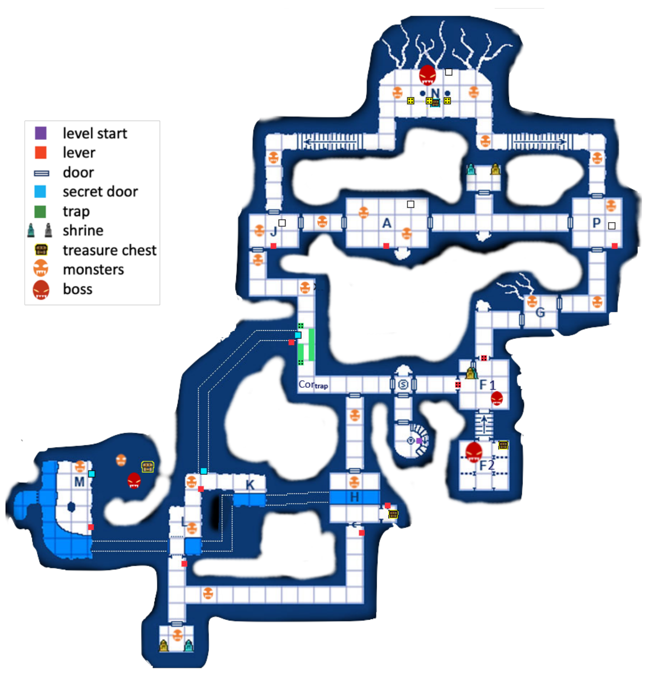

Fig. 1: 'Castle Zenopus' level in Dungeons & Dragons [15].

a game under test. We explore how our approach formulates such requirements into formal specifications and verifies them. Additionally, we evaluate the performance of EmoSTL.

The paper is organized as follows. Section II provides motivating examples. Section III and IV introduce EmoSTL and its formal semantics. Section V explains the user study and the application of EmoSTL in a case study alongside a performance evaluation. Section VI discusses related work. Section VII concludes the paper and outlines future work.

## II. A Motivating example

This section shows several examples of envisaged use of our proposed language for defining PX specifications to aid PX testing. To present these examples, a few notations are explained here informally to provide some insights. The language syntax, however, is given in Section III. Figure 1 shows a running example of a level in *Dungeons & Dragons Online* named *Castle Zenopus*. The castle consists of rooms and corridors filled with traps and monsters. There are also levers to unlock doors, and shrines to heal the player. The player's goal is to explore the castle and defeat the boss monster in room F2 [15]. Imagine a setup where some players play the level, and data are collected, capturing the game behavior and the players' emotions during playing. Alternatively, an emotional AI agent can play the level and report emotions it feels over time [16]. Assume that $S$ consists of the traces of emotional experience and $\phi$ expresses a property over traces. We write $S$ valid $\phi$ to check if $\phi$ is satisfied by all traces in $S$ and $S$ sat $\phi$ to check if at least one trace satisfies $\phi$. Such specifications in our language, are exemplified as follows:

**Example-1** Every player would always feel a sense of hope during playing the designed level:

$$S \text{ valid always } (h > 0) \tag{1}$$

where $h$ denotes the player's intensity (strength) of hope.

**Example-2** When a player passes the corridor containing traps ($cor_{trap}$), they would feel distressed at least once:

$$S \text{ valid ( seq } [cor_{trap}] \rightarrow \text{seq } [cor_{trap}.\mathbf{D}'] \text{ )} \tag{2}$$

where $\mathbf{D}'$ denotes a rise in the intensity of distress.

**Example-3** There is at least one trace in which the player experienced a rise in fear in 20% of the area of room N.

$$S \text{ sat covered } (N.\mathbf{F}', 20\%) \tag{3}$$

where $\mathbf{F}'$ denotes the presence of a rise in the intensity of fear.

**Example-4** Here, we would like to check the effect of monsters and a boss in rooms A and N on players' emotional experience during a specific time interval. The specification below searches for a trace in which the player initially experiences a rise in fear when passing room A and never experiences fear afterward until it reaches N where the player becomes distressed before experiencing a rise in hope within 20 time units:

$$S \text{ sat seq}^{[1,20]} \text{ [ } A.\mathbf{F}'; \text{absence } \mathbf{F}'; N.\mathbf{D}'; N.\mathbf{H}' \text{ ]} \tag{4}$$

where $\mathbf{F}'$, $\mathbf{D}'$ and $\mathbf{H}'$ denote the presence of an increase in the player's fear, distress, and hope.

## III. EmoSTL: The Language Syntax

The proposed language is based on finite-trace LTL [17], extended to incorporate time intervals and spatial expressions, and interpreted over enriched states that are assumed to carry emotion-related information. It is also more restrictive than LTL, as it does not permit arbitrary nesting of temporal operators. For convenience, we call the language EmoSTL: Spatial Temporal Language of Emotions. This section introduces the syntax. Section IV gives the formal semantics.

### A. Rich State Formula

The basic building blocks of every specification are state formulas that are evaluated at a particular state in a given emotion trace. An 'emotion trace' is a finite sequence of states that records the player's emotions along with relevant state variables as the player plays a game, assuming these traces are available. Section V discusses options to obtain these traces.

A state formula might specify a particular variable should have a certain value or bound, e.g. $h{>}0$. Some variables have specific PX-related meaning, e.g. $h$ represents the player's hope intensity, as recorded in the emotion trace. Formulas can also express spatial properties, differential properties towards the previous state, and aggregation over an entire trace. The precise syntax of rich state formulas is given below:

$$\begin{array}{lll} P & ::= & \neg P \;\mid\; P_1 \wedge P_2 \\ & & \mid\; A.P \;\mid\; \textsf{covered}\;(A.P\;,\;\mathit{constant}) \\ & & \mid \mathit{variable}\;\mathit{rel}\;\mathit{constant} \\ \mathit{rel} & ::= & = \mid \neq \mid \leq \mid < \mid > \mid \geq \\ \mathit{variable} & ::= & \mathit{ordinary\text{-}variable} \\ & & \mid \mathit{px\text{-}variable} \\ & & \mid \mathit{primed\text{-}variable} \end{array} \tag{5}$$

- Negation and conjunction have the usual meaning.
- $A$ is a spatial expression that describes an area, such as a room. The syntax for $A$ is shown in Eq.(7). $A.P$ describes a state where the player is in $A$ and the formula $P$ holds. We allow simply $A$ to be treated as a state formula, as an abbreviation of $A$.true.
- covered *(A. P , constant)* expresses the proportion of geospatial points (constant value) within areas A where $P$ is present (further in Def. 2).
- *variables*: There are several types of variables. An $\mathit{ordinary\text{-}variable}$ refers to the corresponding part of a state in the trace, e.g. the player's health point ($hp$). Ordinary variables include time and location that correspond to the time when the current state is sampled, and the player position in that state.
  There are also $\mathit{px\text{-}variables}$, e.g. $h$, which we use to refer to the intensity of the emotion hope recorded in the trace. Examples of $\mathit{px}$-v$\mathit{ariables}$ are as listed below:
  $$\mathit{px\text{-}variable}\;::=\;h \mid j \mid s \mid f \mid d \mid p$$
  representing hope, joy, satisfaction, fear, distress, and disappointment. Here, the choice of these variables is based on the well-known psychological structure of emotions presented by Ortony, Clore & Collins [18] as the OCC structure. However, any emotion type can be utilized as $\mathit{px\text{-}variables}$ as long as the traces expose their values. For every variable, e.g. $h$, a primed-version $h'$ refers to the difference between its values in the current and the previous state in the trace. Thus, an emotion rise can be expressed a positive primed value. Notice that these rich state formulas are actually trace formulas referring to two states. The degree of the rise is expressed with a certain threshold $c$. The following derived notations are introduced for the aforementioned examples:
  $$\begin{array}{lll} \mathbf{H'}_c = \mathsf{h}' > c & ,\; \mathbf{J'}_c = \mathsf{j}' > c & ,\; \mathbf{F'}_c = \mathsf{f}' > c \\ \mathbf{D'}_c = \mathsf{d}' > c & ,\mathbf{S'}_c = \mathsf{s}' > c & ,\mathbf{P'}_c = \mathsf{p}' > c \end{array} \tag{6}$$
  When $c$ is 0, it is removed from the notation; e.g. $\mathbf{H'}_0$ becomes $\mathbf{H'}$.

*Spatial Expression:* An area can represent a room, a corridor, or any fine-grained area as small as a $1 \times 1$ square. An essential property of the expression is to check whether a location is inside an area, denoted as $\ell \in A$, where $\ell$ is a location and $A$ is an area. $A$ is defined as follows:

$$A ::= \mathit{circle} \mid \mathit{rectangle} \mid A_i \cup A_j \mid A_i \cap A_j \mid \overline{A} \tag{7}$$

where $A_i$ and $A_j$ represent two distinct area. Area $A$ can be defined as, e.g. a rectangle described by the bottom left vertex and the top right vertex of the rectangle as $(x_i, y_i)$ , $(x_j, y_j)$.

## B. Temporal Expression

A temporal formula is used to express a property over an emotion trace, as opposed to state formulas, which express a property on a single state. While LTL can be used to express such a formula, LTL is notoriously hard to understand [9]. Requiring game designers to write plain LTL formulas appears unproductive and error-prone. We, therefore, propose an alternative language, which in the temporal aspect is more restricted than LTL, but makes use of the richer state formula as defined in the section III-A. The language of temporal formulas is defined recursively as follows; below, $\phi$ and $\psi$ are temporal formulas, and $P$ is a state formula.

$$\begin{array}{lll} \phi & ::= & P \;\mid \neg\phi \mid \textsf{and}(\phi,\psi) \\ & & \mid \textsf{always}(\phi) \\ & & \mid \textsf{seq}\;\;[\;e_1\;;\;...\;;\;e_n\;] \end{array} \tag{8}$$

$$e_i \quad ::= \quad P \mid P^I \mid \textsf{absence}\;P$$

- $\textsf{and}(\phi,\psi)$ and $\textsf{always}(\phi)$ denote conjunction and the temporal 'always' property [19], with their usual meaning.
- seq [ $e_1$ ; ... ; $e_n$ ] expresses the order of states, satisfying the specified 'elements'. An element $e$ can be either a formula $P$ with an optional time interval $I$ or an *absence* element, expressing the absence of a certain state (rather than its occurrence). For example, seq $[P;$ absence $P;$ $Q]$ means $P$ should eventually occur, and then it should be absent until $Q$ occurs.
- $I$ is a time interval, either relative or absolute. A relative $I$ is denoted by a pair of lower and upper time-bounds $\sim[a, b]$ (or with variations such as $[a, b)$ and $[a..]$), intended to put a constraint on when a certain thing should happen. An absolute interval is denoted by $[a, b]$.

For convenience, we also introduce some *derived operators:*

- $\textsf{seq}^I$ [ $e_1; ...; e_n$ ] is a time-interval variation of seq which requires that $e_1; ...; e_n$ should happen within the given time interval. The definition is given later in Section IV-B.
- sustain $P$ = absence $\neg P$, to express that we want to sustain a state rather than just having it occurring once.
- $\textsf{or}(\phi,\psi)$ and $\phi \rightarrow \psi$ with the usual meaning.

## IV. The Language Semantics

### A. State Formula Semantics

To define the semantics of the introduced state formulas, we first need to define *state* and *trace* of a game under test.

- A *state* $\sigma_i$ is a vector of values such that $\sigma_i.v$ denotes the value of the variable $v$ in the state $\sigma_i$.
- A game *trace* is $\sigma = \sigma_0, \sigma_1, ..., \sigma_{n-1}$ is a *finite*, *non-empty* sequence of states that corresponds to a finite game execution of a player. Traces are assumed to be maximal, i.e. they end with the completion of a quest, or a defeat.

**Definition 1.** The semantics of state formulas in Eq.(5) is defined as follows, where $v$ is a *variable* and $c$ is *constant*:

$$\begin{array}{lcl}
[\![\neg P]\!]\ \sigma_i & = & \text{not } [\![P]\!]\ \sigma_i \\
[\![\ P_1\ \wedge\ P_2]\!]\ \sigma_i & = & [\![P_1]\!]\ \sigma_i \wedge [\![P_2]\!]\ \sigma_i \\
[\![v]\!]\ \sigma_i & = & \sigma_i.v \text{ (the value of } v \text{ in the state } \sigma_i) \\
[\![v\ rel\ c]\!]\ \sigma_i & = & [\![v]\!]\ \sigma_i\ \ rel\ \ c \\
[\![v']\!]\ \sigma_i & = & \sigma_i.v\ -\ \sigma_{i-1}.v, \text{ if } i>0 \text{ else undefined}
\end{array}$$

where $rel$ is one of $=\ |\ \neq\ |\ \leq\ |\ <\ |\ >\ |\ \geq$.

**Definition 2.** The semantic of spatial formula $A.P$:

$$[\![A.P]\!]\ \sigma_i\ \ =\ \ [\![P]\!]\ \sigma_i\ \wedge\ \sigma_i.\mathsf{location} \in A$$

Imagine the level is divided into small unit squares of $u{\times}u$. Given a location $l$, $extra(l)$ denotes the unique square in the level such that $l$ is in the square, so spatial-coverage is:

$$[\![\mathsf{covered}\ (A.P, c)]\!]\ \sigma_i\ = \\ \frac{|\ \{extra(\sigma_k.\mathsf{location})\ |\ 0{\leq}k{\leq}i\ \wedge\ [\![A.P]\!]\ \sigma_k\ \}\ |}{|\ \{extra(\ell)\ |\ \ell{\in}A\}\ |}\ \geq\ c$$

covered $(A.P, c)$ asserts that the proportion of unit squares in $A$ where $P$ is true, compared to the total number of unit squares in $A$ is at least $c$. E.g. covered $(A.(h{>}0), 0.8)$ means the presence of hope being witnessed in 80% of the area $A$.

## *B. Base Temporal Language Semantics*

EmoSTL can be translated into LTL. To facilitate translation, we offer an overview of classic LTL and time-bounded LTL, followed by defining mappings between the constructs in our language and the corresponding LTL formulae.

*1) Classic LTL:* LTL is a modal logic that incorporates temporal operators, relating events happening at different times over a linear timeline [8]. LTL formulas are generally interpreted over *infinite* traces. However, some applications [20]–[22] employ offline verification, which involves using a variant of LTL synthesis named LTL$_f$ where specifications are interpreted over *finite* traces [19], [23]. Let $\sigma$ be a trace of length $n{>}0$. Then, $\phi$ holds on $\sigma$ is defined as $\sigma[0..n) \models_f \phi$ which means the satisfaction of $\phi$ on the trace segment $\sigma[0..n)$ can be either **true** or **false**. The LTL$_f$ semantics is inductively defined similar to Baier and McIlraith in [19] as follows.

**Definition 3.** let $0 \leq i < n$:

$$\begin{array}{lcl}
\sigma_{[i..n)}\ \models_f P & = & [\![P]\!]\ \sigma_i \\
\sigma_{[(n-1)..n)}\ \models_f \mathbf{X}\phi & = & \mathbf{false} \\
\sigma_{[i..n)} \models_f \mathbf{X}\phi & = & \sigma_{[i+1..n)} \models_f \phi, \text{ if } i < n{-}1 \\
\sigma_{[i..n)}\ \models_f \neg\phi & = & \text{not } (\sigma_{[i..n)}\ \models_f \phi) \\
\sigma_{[i..n)} \models_f \phi\, \mathbf{U}\, \psi & = & (\exists k : i \leq k < n : \sigma_{[k..n)} \models_f \psi \\
 & & \text{ and } (\forall \ell : i \leq \ell < k : \sigma[\ell..n) \models_f \phi)) \\
\sigma_{[i..n)} \models_f \phi \wedge \psi & = & \sigma_{[i..n)} \models_f \phi \text{ and } \sigma_{[i..n)} \models_f \psi
\end{array}$$

Standard derived operators like *always* $\Box$, *eventually* $\Diamond$ and *implication* $\rightarrow$ are defined as usual [8], [19].

*2) LTL with time interval constrain:* In certain contexts, achieving a specific system state in a certain time interval can be of interest, e.g. when the timing of events determines behavior correctness. Metric Temporal Logic (MTL) [17] is an extension of LTL that allows specifying real-time constrains for timed systems. Unlike classic LTL, that expresses ordering of events in time qualitatively, MTL can quantitatively express real-time constraints. Incorporation of such time interval constraints on $LTL_f$ in this paper is inspired by MTL:

**Definition 4.** Relative-timed until operators

$$\begin{array}{lcl}
\sigma_{[i..n)} \models_f \phi\, \mathbf{U}^{\sim[a,b]}\, \psi & = & (\exists k :\ i{\leq}k{<}n : \sigma_{[k..n)} \models_f \psi, \\
 & & \mathsf{time}_k - \mathsf{time}_i \in [a, b], \\
 & & (\forall \ell : i{\leq}\ell{<}k :\ \sigma_{[\ell..n)} \models_f\ \phi)) \\
\Diamond^{\sim[a,b]}\phi & = & \mathbf{true}\, \mathbf{U}^{\sim[a,b]}\, \psi
\end{array}$$

So, interpreting over $\sigma_{[i..n)}$, $\phi\, \mathbf{U}^{\sim[a,b]}\, \psi$ requires $\psi$ occurs within a time interval $[a, b]$, relative to the time sampled at $\sigma_i$. However, sometimes it is desired to specify time constraints in terms of *absolute* time. With *absolute timing*, events are described in relation to an external reference point, simplifying understanding the occurrence of events. The variation of the until-operator $\mathbf{U}$ with *absolute timing* is as follows:

**Definition 5.** Absolute-timed until operator

$$\phi\, \mathbf{U}^{[a,b]}\, \psi\ \ =\ \ \phi\, \mathbf{U}\ (\psi \wedge \mathsf{time} \in [a, b])$$

The $\Diamond^{[a,b]}\phi$ with *absolute timing* is analogous to the relative time variant in Def. 4. Lower-bound-only $I{=}[a, \infty]$, and upper bound-only $I{=}[0, b]$ can also be applied analogously. From now on, we refer to a time interval as $I$ in formulas.

## *C. EmoSTL Temporal Expression Semantics*

Here, we define the semantics of seq and its timed-variation seq$^I$ which are used to express an order in which properties, or their absence, are expected to occur. Definition of other temporal operators remains the same as LTL$_f$. seq is introduced to simplify emotional specification writing. The primary goal of designers is to identify the occurrence of emotions and the triggers that cause them [24], [25]. With seq syntax, designers more effectively write spatial-temporal testing specifications while maintaining order and addressing changes in emotional intensity. Intuitively, seq$[P_1; P_2; P_3]$ means $P_1$ is expected to happen in the future, followed by (not necessarily immediately) $P_2$, and then by $P_3$. However, for an 'absence' element, expressing the absence of a positive element, seq would mean that the absence should persist until the next element in the sequence occurs. For example, seq[absence $P_1; P_2$] means $\neg P_1\ \mathbf{U}\ P_2$ rather than $\Diamond(\neg P_1 \wedge \mathbf{X}\Diamond P_2)$. This explains why in the syntax in Eq.(8) only non-absence elements are allowed to have a time interval $I$ –As observed in the example, the until-operator $\mathbf{U}$ allows for specifying a time interval for *when* $P_2$ should occur, but this concept doesn't make sense for $\neg P_1$. The precise definition of seq is given below.

**Definition 6.** Let $T$ be a sequence $[e_1; ...; e_n]$; the semantic of $\mathsf{seq}\ [e_1; ...; e_n]$ with *no* time constraint on elements $e_i$ is:

$$[\![\mathsf{seq}\ T]\!] \ = \ \begin{cases} \Diamond[\![T]\!] & \text{, if } T \text{ starts with a non-absence element} \\ [\![T]\!] & \text{, if } T \text{ starts with an absence element} \end{cases}$$

The semantic of the inner structure of the sequence $[\![T]\!]$ is defined recursively, as follows:

$$[\![P;T]\!] = \begin{cases} P \wedge \mathbf{X}\Diamond[\![T]\!], \text{if } T \text{ starts with a non-absence } e \\ P \wedge \mathbf{X}[\![T]\!], \text{ if } T \text{ starts with an absence element} \end{cases}$$

$$[\![\mathsf{absence}\ P\ ;\ T]\!] \ = \ \neg P\ \mathbf{U}\ [\![T]\!]$$
$$[\![\mathsf{absence}\ P]\!] \ = \ \Box\neg P$$
$$[\![P]\!] \ = \ P$$

Below are two examples of sequences with three formulas, one starting with a non-absence element and the other with the absence element, with their translations into LTL$_f$:

$$\begin{array}{lcl} \mathsf{seq}[P_1; \mathsf{absence}\ P_2; P_3] & = & \Diamond(P_1 \wedge \mathbf{X}\ (\neg P_2\ \mathbf{U}\ P_3)) \\ \mathsf{seq}[\mathsf{absence}\ P_1; P_2; \mathsf{absence}\ P_3] & = & \neg P_1\ \mathbf{U}\ \Diamond(P_2 \wedge \mathbf{X}\ \Box\neg P_3) \end{array}$$

**Definition 7.** Consider a formula $\mathsf{seq}[e_1; ..; e_k; ..; e_n]$, where $e_k$ may have a time constraint. Such a constraint is only possible if $e_k$ is a non-absence element. Then, the definition remains the same as the case without time constraint (Def. 6), with the following extension:

- If $T$ starts with a time-constrained element $P^I$, $[\![\mathsf{seq}\ T]\!] = \Diamond^I[\![T]\!]$. Otherwise, $[\![\mathsf{seq}\ T]\!]$ is the same as in Def. 6.
- For the inner structure of sequence, if $T$ starts with a time-constrained element $e_1^I$, then:
  - $[\![P;T]\!] = P\ \wedge\ \Diamond^I[\![T]\!]$,
  - $[\![\ \mathsf{absence}\ P\ ;\ T]\!] = \neg P\ \mathbf{U}^I\ [\![T]\!]$.
- For other cases, $[\![T]\!]$ is the same as in Def. 6.

For instance, $\mathsf{seq}\ [P_1^{I_1}; \mathsf{absence}\ P_2;\ P_3^{I_2}]$, where $I_1$ and $I_2$ are not in conflict, is translated into LTL$_f$ as:

$$\mathsf{seq}\ [P_1^{I_1}; \mathsf{absence}\ P_2;\ P_3^{I_2}] = \Diamond^{I_1}(P_1 \wedge \mathbf{X}(P_2\ \mathbf{U}^{I_2}\ P_3))$$

Finally, the outer time interval over a seq, written as $\mathsf{seq}\ ^I[e_1; ...; e_n]$, can be defined by distributing $I$ over every non-absence element in the sequence.

### D. Satisfiability and Validity Check

Let $S$ be the set of game traces collected from players. Given a temporal formula $\phi$, we are interested in three types of verdict for $\phi$: whether there is no counter-example of $\phi$ can be found within the traces in $S$, whether there is no trace that satisfies $\phi$, and whether there is at least one trace (or with a certain minimum ratio of traces $c$ ) satisfying $\phi$:

$$\begin{array}{lcl} S \models\ \mathsf{valid}\ \phi & = & (\forall\sigma \in S ::\ \sigma \models \phi) \\ S \models\ \mathsf{unsat}\ \phi & = & (\forall\sigma \in S ::\ \sigma \models \neg\phi) \\ S \models\ \mathsf{sat}\ \phi & = & (\exists\sigma \in S ::\ \sigma \models \phi) \\ S \models\ \mathsf{sat_c}\ \phi & = & (\frac{\mid \{\sigma \in S ::\ \sigma \models \phi\} \mid}{\mid S \mid} \geq c) \end{array} \tag{9}$$

## V. Case Study

This section describes two research questions and the approach we employ to conduct our study to answer them. As explained earlier, PX testing lacks a formal language to accurately capture and articulate the nuances of emotions. The absence of a standardized language makes the formal verification of emotional experience requirements challenging. We investigate the following research questions for the proposed language EmoSTL (Section III) that allows the formulation of emotion requirements as formal specifications.

***RQ1:*** *Is it possible to formulate emotional requirements given by developers into formal specifications using* EmoSTL*?*

***RQ2:*** *Can we validate* EmoSTL *specifications crafted for assessing emotional experiences?*

### A. Formulating Emotion requirements: User Study

We conduct a user study with skilled software engineers to address the **RQ1**. Here, we explain the experimental setup and its findings. The experiment is structured as 6 stages:

1) *Game under study.* We selected a game called Lab Recruits [26], a configurable 3D game developed with the Unity game engine, designed specifically for software engineers (SEs) to test their automated game testing approaches on test scenarios. This study uses a newly crafted level that still needs emotional experience assessment and possible adjustment before the release.
2) *Tutorial.* We designed a tutorial session in which participants were trained on how to play the game and received a summary of what emotion and game experience are.
3) *Task design.* The tasks focus on two dimensions: first on their experience as players and later on their perspective as testers to write requirements for emotions.
4) *Participants.* We asked five skilled software engineers in testing and verification to participate in our study.
5) *Pilot.* We perform a pilot study to identify issues with the tutorial, tasks, and session.
6) *Experiment session.* We designed group sessions to let the participants play the game, perform the tasks and collect the data to answer the first research question.

Below, we provide further elaboration on the stages.

*Game under study:* Figure 2 shows scenes of Lab Recruits game which presents a maze environment with rooms, doors, and buttons, sometimes accompanied by zombies and fire hazards. The primary objective is to reach the end-game healing flag, which is protected behind doors and by zombies. To have access, players must uncover the associations between buttons and doors in different areas which forms a chain of dependencies. Without preceding knowledge, the player needs to reveal the correlation between buttons and doors to find a path to the final goal flag. Points can be earned by opening doors, but players must avoid fire hazards, contact with zombies, and falling from buildings, designed to make navigation of certain areas difficult. If the player either falls down or reaches zero health points, the game ends in failure.

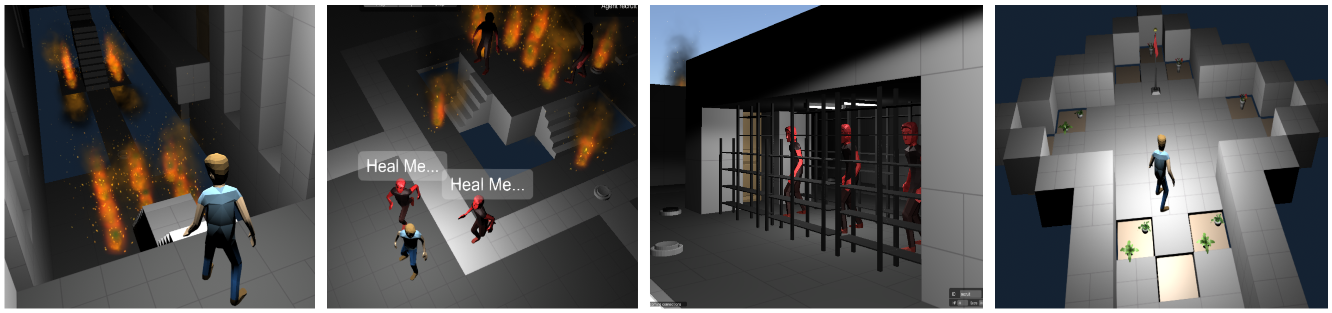

Fig. 2: Scenes of the Lab Recruits game.

The Lab recruits level under test for the upcoming experiment highly resembles the *Castle Zenopus* level in the world of *Dungeons & Dragons* as shown earlier in Figure 1. The crafted level is a large level ($217m \times 208m$) with 9 rooms, 23 buttons, 27 doors, 16 zombies, and over 50 fire hazards.

*Tutorial sessions:* Participants receive an email with a download link for a standalone version of the game. The email also includes the assigned level, a summary of the emotional experience, an overview of the experiment and its objectives, as well as a set of experiment rules and ethical principles. On the day of the experiment, participants are given a tutorial session on the game mechanics, ensuring they understand how to play it. Additionally, any questions they may have are answered before the start of the experimental session.

*Task design:* Three tasks are defined for participants: $T1$- after the tutorial, participants individually play the game level for 10 minutes, to reflect on the impact of the game design on their emotional experience and essential emotional requirements for subsequent tasks. $T2$- is to answer the question based on their personal experience and feelings as players. The purpose of this task is to evaluate participants' overall perception of the game. $T3$- involves participants taking on the role of the game designer, utilizing their skills in testing and verification to write 5 requirements of emotions without any constraint in a natural language; These requirements should capture emotions and patterns they find important to know as testers or anticipate to occur or never occur within the game. The goal of this task is to answer the **RQ1** by evaluating the strength of EmoSTL to formulate the participants' set requirements. We aim to effectively capture and articulate the expert-like requirements with the language. To complete all these three tasks, participants are provided with the layout of the level with the room labels, a table of emotions with their psychological meaning, and real-life examples of emotions based on the definition of emotions by Ortony, Clore, and Collins (**OCC**) [18]. Additionally, participants receive a table, showcasing examples of simple emotional requirements.

*Participant:* An open invitation is sent to researchers and developers in a software division of a university, target people with at least 2 years experience in software testing and verification. In total, we selected 5 volunteers for the experiment: four Ph.D. students and one bachelor-level software engineer to run the pilot study. All had prior experience as game players. In the sequel, we refer to the participants as SEs.

*Pilot:* A pilot is performed with a bachelor level software engineer to assess the clarity of the experiment's tasks and optimize the workload. After the pilot, questions are adjusted and the session is reduced to 50 minutes.

*Experiment session:* Upon completing a 10-minute game-play, participants are required to answer the questionnaire, which includes writing requirements using Wooclap [27].

**Results.** First, we present the participants' personal emotional experience results for the game($T2$). Following that, we offer formalization of written requirements as EmoSTL specifications. Given a list of emotions, participants are asked to express their general experience during the game-play with up to two most significant emotions. Emotions such as excitement, fear, and frustration are reported which predominantly lean toward negative polarity. To graphically show the reported emotions in a two-dimensional space, we rely on the definition of emotion in terms of *Pleasure*, *Arousal*, and *Dominance* (PAD) dimensions [28] to map emotions. Mehrabian and Russel [28], [29] present a model in which every single emotion can be quantitatively expressed in terms of three dimensions as *Pleasure*, *Arousal*, and *Dominance*. A series of studies [29], [30] suggest the polarity and values of various emotions in three dimensions of PAD model using verbal-report, physiological, and behavioral measuring of emotions done which serve as evidence of the representations of emotions into the PAD dimensions. Figure 3 shows the emotion diversification of the player's experience of the game, mapped on Pleasure-Arousal dimensions. The result shows that despite having positive emotions such as joy and hope, the level mostly elicits negatively-pleasured emotions such as fear which can potentially be the intended feature of the game. All reported emotions are positively aroused emotions. The presence of arousal reported emotions shows that the crafted game level evokes emotional responses in players and therefore we chose this game level to evaluate the ability of EmoSTL in capturing emotion requirements as specifications.

**Pre-processing of requirements.** The purpose of the expert study is to evaluate the ability of EmoSTL to formulate real software engineers' set emotion requirements as indicated in **RQ1**. To do so, the requirements first undergone a pre-processing procedure to remove e.g. fragments that pose unrealistic requirements or redundant fragments that only serve as providing clarification or explanation towards an asserted

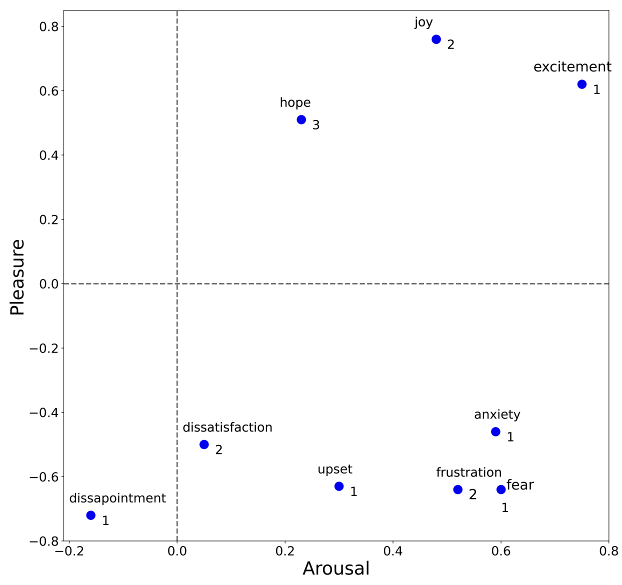

Fig. 3: Emotion diversification of players' experience for the 'Castle Zenopus' level in Lab Recruits game.

main specification. An example of such requirements is R3: "every player would feel a bit worried about going through area G <u>since there is fire there</u>". The underlined fragment only provides an explanation for the non-underlined part; so it is dropped by the pre-processing. Additionally, fragments that refer to information that is technically infeasible to obtain and is not relevant for the purpose of this study, are also dropped. R2 is an example of this: "every player would feel anticipation on how to go through the fire in area $G$ without dying, as well as <u>if it is worth to press the button in that area</u>". The underlined part refers to the value of doing an action, which is hard to infer. So, that part is omitted. There are also three cases in which the participant describes their unwanted experience with a game setup rather than outlining a requirement. For example, they express their dissatisfaction with the lack of challenge posed by the zombies, as in R15: "Most players will feel dissatisfied that the zombies do not really pose a challenge". However, instead of dropping it, we address such cases by checking its negation which in this case is turned into the revised assertion: "Some players will not feel dissatisfied with the challenge posed by the zombies". We also applied the same approach for R5 and R11, expressing undesired behavior, as agreed upon by the requirement writer. In general, there are only a few requirements among 25 requirements that have been adjusted; The rest remain unchanged.

Table I lists a subset of the requirements set by the five software engineers, and their formalization using EmoSTL. We choose to interpret a presence of an emotion e.g. feeling excited as observing an increase in that specific emotion from its baseline state. The main reason for such translation is that a presence of an emotion such as excitement, often implies a state of transition from a baseline state (considered as zero) to some intensity value, suggesting a rising level of excitement. Thus, the presence of emotion and an increase in the level of present emotion are both denoted by primed-variables e.g. $\mathbf{Exct}'$. Collected dataset along with the translation of all requirements into EmoSTL formal specifications are available[1].

Since the focus of this part of the study is to assess EmoSTL's ability in formulating emotion requirements, the specifications are written with the assumption that referred emotion types (e.g. fear) can be captured within the traces on which the specifications are to be verified, e.g. by employing data collection approaches such as player self-reports, facial expression recognition, or intelligent emotional agent modeling. As can be seen in Table I, not only can spatial and temporal specifications be captured, but the DSL also allows functional and emotional properties to be combined within a specification (e.g. specification R2 in Table I) as long as the traces encompass the necessary information which can be treated as *ordinary-variable* in the DSL.We will refer to these specifications as mixed specifications. Among 25 set requirements, there are two requirements that need access to data at two arbitrary times in the past for comparison. One is "The player feels more nervous for the first time that they pass room G compared to their first time in room F1 for all gameplays." and the other is "Every player feels decreasingly anxious every time that they pass room G". Formulation of them involves access to two distinct moments which goes beyond the expressiveness of the current version of the DSL, which provides access only to the current and the preceding state at any given time. It is technically possible to add this feature. However, how to do this efficiently with low time complexity is a topic for future work. Figure 4 shows the statistics of EmoSTL terms used for the translation of the 23 set requirements into specifications. In the figure, 'functional term' is a fragment that expresses a condition over the functionality of the game. As can be seen, seq and spatial terms are the most frequently employed terms in the translations, appearing in 21 out of 23 instances (0.91) and 17 out of 23 instances (0.74), respectively, indicating the usefulness of such terms. Another interesting finding from the user study is that experts tend to often formulate mixed requirements that target both functional and emotional aspects of a game. This occurs in 11 out of 23 instances(0.47). Additionally, despite relatively limited use of absence and sustain terms in specifications, their inclusion enriches the language's expressiveness, allowing it to address critical special cases. Furthermore, we evaluate the length of specification in EmoSTL and compare it with the equivalent specification in $LTL_f$. The length of the specification is defined as the number of all operators. Figure 5 illustrates the lengths of the specifications written in $EmoSTL$ are 17.5% shorter than those in plain $LTL_f$, with a greater reduction of 22.5% for plain $LTL_f$ formulas larger than four. It's important to note that this reduction in length not only enhances usability and ease of specification writing but also improves understanding, as $LTL_f$ specifications become challenging and harder to understand as they grow in length.

[1]https://zenodo.org/record/8409957

TABLE I: *Emotion requirements from the user study with their formalization as* EmoSTL *specifications.*

| No. | Emotion requirements | EmoSTL formalization |
|---|---|---|
| R1 | Every player would feel excited at the beginning of the game, in area P. | valid $\phi = \mathsf{seq}^{[0,b]}$ $[\ P.\mathbf{Excit'}\ ]$, where $\mathbf{Excit'}$ refers to the excitement increase in the collected trace and $[0, b]$ is the duration considered to be the beginning of the game by the SE. |
| R2 | Every player would feel anticipation on how to go through the fire in area G without dying. | valid $\phi = \mathsf{seq}\ [\ G\ ]\ \rightarrow\ \mathsf{seq}\ [G\ ;\ G.(\mathbf{Antcp'} \wedge hp{>}0)]$, where $\mathbf{Antcp'}$ refers to anticipation increase and $hp$ is the health point which must be greater than zero for the player to remain alive. |
| R4 | There is at least a game-play in which a player would feel hopeful to complete the game as soon as they reach room F2. | sat $\phi = \mathsf{seq}\ [\ F2\ ;\ F2.\mathbf{H'}\ ^{\sim[0,b]}]$, where $\mathbf{H'}$ refers to the hope increase in the traces and $b$ is some reasonable time upper bound that is decided by SE to capture "as soon as". |
| R5 | There is no game-play in which a player would feel a bit disappointed after walking for a while in rooms P, G and F1. | unsat $\phi$ = and( seq$[P]$, seq$[F1]$, seq$[G]$, seq$[\mathbf{P'}]$ ), where $\mathbf{P'}$ refers to the disappointment increase in the traces. |
| R6 | When every player begins playing the game, they should be pleased to accomplish something right away like pressing a button ($b0$ or $b1$). | valid $\phi = \mathsf{seq}\ [\ (pressed_{b0} \vee pressed_{b1})^{[0,a]}\ ;\ \mathbf{Plsd'}^{[0,b]}\ ]$, where $a$ and $b$ represent upper bounds on the time considered as the beginning of the game ('right away') by SEs, $pressed_{b0}$ and $pressed_{b1}$ are the status of buttons and $\mathbf{Plsd'}$ refers to an increase in pleasure emotion in the traces. |
| R7 | Every player should feel a sense of fear as well as joy at least once in the whole game even if they lose the level fairly quickly. | valid $\phi$ = and( seq $[\ \mathbf{J'}\ ]$, seq $[\ \mathbf{F'}\ ]$), where $\mathbf{J'}$ and $\mathbf{F'}$ are the increase in joy and fear in the traces. |
| R8 | There is a game-play in which the player keeps hopes that fire flames in room G can be avoided easily. | sat $\phi = \mathsf{seq}\ [\ G\ ;\ \mathsf{sustain}(G.h > 0)\ ;\ \neg G\ ]$, where $h$ refers to hope variable in the traces. Note that the fire flames are the only elements within room P that, when touched, change the level of hope in a player to win the level. |
| R10 | There is at least one game-play in which the player should pass a challenge to reach a closed door $d1$ and within the next 10 seconds realize that they need to come back and feel nervous. | sat $\phi = \mathsf{seq}[\ closed_{d1} \wedge reached_{d1}\ ;\mathbf{Nrv'}^{\sim[0,10sec]}\ ]$, where $closed_{d1}$ is a variable representing whether the door $d1$ is closed, and $reached_{d1}$ means means that the player is nearby that door. $\mathbf{Nrv'}$ refers to the increase in nervousness in the traces. |
| R12 | In at least one game-play, a player will be confused and disappointed after pressing the button at the bridge in room G which controls the door between rooms G and F1, if they have not learned yet that pressing a button twice will close the door again. | sat $\phi = \mathsf{seq}[\ \neg pressed_{bBridge};\ pressed_{bBridge};\ \neg pressed_{bBridge}; G.\mathbf{Conf'} \wedge G.\mathbf{P'}\ ]$, where $pressed_{bBridge}$ refers to the status of the bridge button, $\mathbf{Conf'}$ and $\mathbf{P'}$ to respectively to confusion and disappointment increase in the traces. |
| R13 | Some players will feel delighted after they pass through the fire in room G and gain their health back by interacting with a healing flag in room P. | sat$_c$ $\phi$ = and(seq$[\ G; interacted_{flagp} \wedge \mathbf{HP'}\ ]$, seq $[\ P.\mathbf{Deltd'}\ ]$), where $interacted_{flagp}$ is the status of the specific flag ($ordinary-\ variable$), $\mathbf{HP'}$ refers to the rise in health point after the last state, $\mathbf{Deltd'}$ is the increase in emotion delight in the traces and $c$ is the ratio of traces that need to satisfy the specification. |
| R14 | Some people may feel fear when encountering the zombies in room F1. | sat$_c$ $\phi$ = seq$[\ F1.nearEnemy; F1.\mathbf{F'}\ ]$, where $nearbyEnemy$ shows whether the player got close to the enemy and $\mathbf{F'}$ is the increase in the fear variable in the traces. |
| R15 | Some players will not feel dissatisfied with the challenge posed by the zombies. | sat$_c$ $\phi$ = always ( $nearEnemy \rightarrow \neg\ \mathbf{Dissat'}$ ), where $nearbyEnemy$ shows whether the player got close to the enemy and $\mathbf{Dissat'}$ show an increase in dissatisfaction emotion in the traces. |
| R16 | Players always feel a sense of fear when faced with the zombies in room F2. | valid $\phi$ = always ( $F2.nearEnemy \rightarrow$ seq $[\ F2.F'\ ]$ ) , where $\mathbf{F'}$ is the increase in the fear variable in the traces. |
| R18 | If the player first moves towards F1 via corridor GF1, and meets the closed door there, they will typically feel anticipation for going the other way, finding out how to open this door. | valid $\phi$ = seq$[\ GF1 \wedge closed_{doorF1}\ ]\rightarrow$seq$[\ GF1.\mathbf{Antcp'} \wedge closed_{doorF1}]$, where $closed_{doorF1}$ refers to the status of the door and $\mathbf{Antcp'}$ is an increase in anticipation emotion in the traces. |
| R19 | Some player always feels joyful when reaching the finish flag alive. | sat$_c$ $\phi$ = seq$[\ Finish \wedge hp > 0 \wedge \mathbf{J'}\ ]$, where $Finish$ is the status of the game, $hp$ is the health point and $\mathbf{J'}$ is the increase in joy field in the traces. |
| R22 | Every player feels nervous when they enter room F1 for the first time. | valid $\phi$ = seq$[F1]\rightarrow$seq[absence $F1\ ;\ F1\ ;\ F1.Nrv'$], where $\mathbf{Nrv'}$ is the increase in nervousness in the traces. |

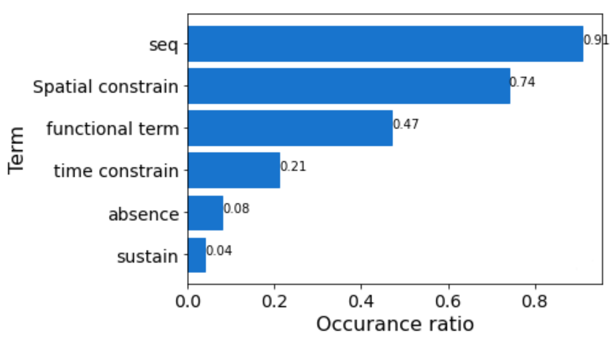

Fig. 4: Statistics of terms in set specifications.

**RQ1 summary. The result of our user study shows the proposed DSL allows us to formally capture the important properties of set emotion requirements.**

## *B. Verification of Emotion Specification*

In this section, we explain the result of the formal verification of the set emotion specifications in Table I to answer the **RQ2**. To monitor a specification in offline runtime verification, acquiring execution traces is crucial. For emotion verification, these traces can be acquired by different methods such as self-reported emotions [31], facial expression

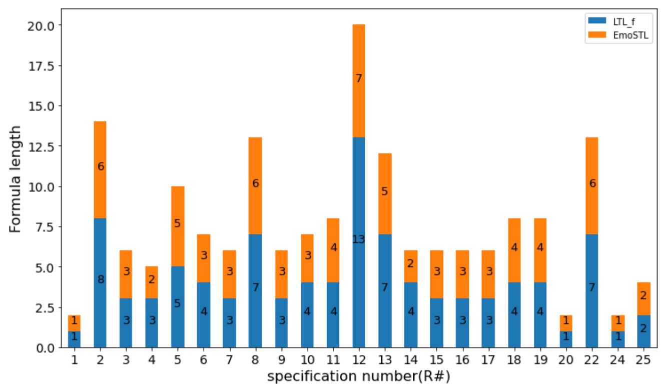

Fig. 5: LTL$_f$ and EmoSTL formula length comparison.

TABLE II: *Characteristics of traces gathered from agent executions.*

| No. | Game interactions # | Execution time (ms) | Game status |
|---|---|---|---|
| 1 | 999 | 97,295 | Lose |
| 2 | 1,872 | 174,495 | Win |
| 3 | 2,124 | 208,881 | Win |
| 4 | 3,002 | 222,818 | Lose |
| 5 | 3,028 | 247,107 | Win |
| 6 | 3,113 | 278,615 | Win |
| 7 | 3,177 | 218,184 | Lose |
| 8 | 4,091 | 355,332 | Win |

TABLE III: *Emotions and their tokens based on OCC theory [18]*

| Primary emotion | Tokens |
|---|---|
| **Hope** | anticipation, excitement, expectancy,hopeful |
| **Fear** | worried, anxious, fright, nervous, scared, terrified |
| **Joy** | delighted, pleased, feeling good, happy,glad, joyful |
| **Disappointment** | despair, frustration, heartbroken |

recognition [32], [33], physiological measurement [34], [35], or emotion reasoning using an intelligent agent [36]–[39]. We employ the latter approach, proposed by Ansari et.al [16], [38] that utilizes a computational model of emotions using the OCC emotion theory to reason about the emotional state of an agent during the game based on perceived events and the agent's goal. To have an intelligent agent, we employ a Belief-Desire-Intent (BDI) [40] technique for the agent's decision-making process to explore and pursue its goal to win the game level. While the strategy is outlined by the developer at a higher level, the agent's BDI decision-making mechanism determines how to pursue the game within the provided strategy. We use Aplib [41], an agent programming library, that is developed to facilitate this particular approach. Here, we use eight distinct strategies, including one that takes the shortest path to the goal and another that explores all rooms before reaching the finish point. These strategies are designed to align with the sensical approaches typically employed by human players when they play a Lab Recruits level. This led to eight distinct game executions, each leading the agent on a different path that may either result in victory or defeat (see Table II). These game executions can represent a broader player base, as some players make similar choices and traverse similar paths during their play . Throughout the execution, the agent not only makes the decision for its next action upon observing events but also computes its emotional state (using the model in [38]) which is then recorded in a trace file. The resulting eight trace files are then used to verify the earlier formulated emotion specifications in Table I. The specifications are designed independently of the emotions that a system can identify. However, the ability of offline verification relies on the trace files. In this context, some emotions mentioned in Table I, e.g., nervousness, do not correspond to OCC's primary emotion categories and, therefore, are not computed using the model [38]. We, nevertheless, tackle this problem by mapping such emotions to the primary OCC emotion types. This approach is grounded in the OCC theory, where certain emotions are perceived as tokens or synonyms of the primary emotions [18]. Among 23 formal specifications in our case, there are only two of them that cannot be mapped via OCC emotion tokens, namely confusion and dissatisfaction in R12 and R15, which are associated with secondary/complex emotions, resulting from a combination of other emotions and not categorized as primary emotions. The rest can be accommodated within the OCC tokens as shown in Table III which are then mapped to the corresponding primary emotions for the verification. R17 is also excluded as it refers to an emotional response triggered by the game's sound effect; for technical reasons, it was not possible to trace the sound effect.

Table IV shows the verification results of the 20 specifications. The constant values of the specifications, such as time boundaries and satisfaction ratio $\mathsf{sat_c}$ (here, 0.25) are chosen by the game designer based on the reasonableness of the game's configuration. The pre-conditions of implicative specifications are also tested separately and they occur at least once in the trace files. The results show that 16 of 20 specifications have passed the verification, and only R6,R7, R18 and R25 are failed. The main reason behind failing R6,R7, R18 is due to the current game design. This shows that the specifications can actually detect issues within the current game design. Note that imposing valid-type of assertions for some requirements such as R18 may actually be overly-restrictive as individual players may respond emotionally differently to the game dynamism. Detecting such issues is valuable to help in negotiations between game designers and testers on potential adjustments to either the game design or the requirements. EmoSTL [2], the user study, the agent simulations, along the trace files are publicly available.

[2]https://github.com/SabaGholizadehAnsari/ltl-pxevaluation.git

TABLE IV: *Emotion Specification Verification: 'Castle Zenopus' Level Results.*

| No. | Emotion Specification | Verified |
|---|---|---|
| R1 | valid $\phi = \text{seq}^{[0,\ 7sec]}\ [\ P.H'\ ]$ | ✔ |
| R2 | valid $\phi = \text{seq}\ [\ G\ ]\ \rightarrow\ \text{seq}\ [G\ ;\ G.(H' \wedge hp > 0)]$ | ✔ |
| R3 | valid $\phi = \text{seq}\ [\ G\ ]\ \rightarrow\ \text{seq}\ [\ G.F'\ ]$ | ✔ |
| R4 | sat $\phi = \text{seq}\ [\ F2\ ;\ F2.H'\ ^{\sim[0,\ 2sec]}]$ | ✔ |
| R5 | unsat $\phi = \text{and}(\ \text{seq}[\ P\ ], \text{seq}[\ F1\ ], \text{seq}[\ G\ ], \text{seq}[\ P'\ ]\ )$ | ✔ |
| R6 | valid $\phi = \text{seq}\ [\ (pressed_{b0} \vee pressed_{b1})^{[0,\ 30sec]}\ ;\ J'^{[0,\ 50sec]}\ ]$ | ✗ |
| R7 | valid $\phi = \text{and}(\ \text{seq}\ [\ \mathbf{J}'\ ], \text{seq}\ [\ \mathbf{F}'\ ]\ )$ | ✗ |
| R8 | sat $\phi = \text{seq}\ [\ G\ ;\ \text{sustain}(G.h > 0)\ ;\ \neg G\ ]$ | ✔ |
| R9 | valid $\phi = \text{seq}[\ F2\ ] \rightarrow \text{seq}[\ F1.F'\ ]$ | ✔ |
| R10 | sat $\phi = \text{seq}[\ closed_{d1} \wedge reached_{d1}\ ; F'^{[0,10sec]}\ ]$ | ✔ |
| R11 | $\phi$ sat $= \text{and}(\ \text{seq}[\ G\ ], \text{seq}[\ \text{absence}\ G.P'\ ]\ )$ | ✔ |
| R13 | sat $_{0,25}\phi = \text{and}(\ \text{seq}[\ G; interacted_{flagp} \wedge \mathbf{HP}'\ ], \text{seq}\ [\ P.\mathbf{J}'\ ]\ )$ | ✔ |
| R14 | sat $_{0.25}\phi = \text{seq}[\ F1.nearEnemy; F1.F'\ ]$ | ✔ |
| R16 | valid $\phi = \text{always}\ (\ F2.nearEnemy \rightarrow \text{seq}\ [\ F2.F'\ ]\ )$ | ✔ |
| R18 | valid $\phi = \text{seq}[GF1 \wedge closed_{dF1}] \rightarrow \text{seq}[GF1.H' \wedge closed_{dF1}]$ | ✗ |
| R19 | sat $_{0.25}\phi = \text{seq}[\ Finish \wedge hp > 0 \wedge J'\ ]$ | ✔ |
| R20 | sat $_{0.25}\phi = \text{seq}[\ G.F'\ ]$ | ✔ |
| R22 | valid $\phi = \text{seq}[F1] \rightarrow \text{seq}[\text{absence}\ F1\ ;\ F1\ ;\ F1.F']$ | ✔ |
| R24 | sat $\phi = \text{seq}[\ F2.J']$ | ✔ |
| R25 | unsat $\phi = \text{seq}[\ P.h \leq 0]$ | ✗ |

**RQ2 summary. The verification results confirm that the formulated emotion specifications can be monitored in a real-game scenario and EmoSTL contributes to the detection of game design issues.**

*Threat to Validity.:* As external threats, performing the experiment on one game may not be generalizable. However, we believe EmoSTL can be used for formal verification of emotion-inducing games such as role-playing games like *Dungeons & Dragons Online*, provided that emotions are captured via a data collection method. To mitigate errors in translating informal requirements into formal specifications due to some pre-processing as an internal threat, we reduce pre-processing and two authors reviewed both the pre-processing and the formal specifications and modify those on which they do not agree. Moreover, the emotion traces generated by an artificial agent using the OCC emotion model might not align closely enough with human experiences. but using the OCC emotion model as ground truth still allows for rapid feedback.

### C. Time Complexity of EmoSTL

We examine the performance of the proposed language EmoSTL. Figure 6 illustrates the performance of EmoSTL with respect to different trace and formula lengths. We are applying two formulas: $Formula1 = \text{seq}[\ A1.\mathbf{H}'; \text{absence}\ \mathbf{F}'; \mathbf{H}'\ ]$ and one with a time constraint: $Formula2 = \text{seq}^{[5000,50000]}[\ A1.\mathbf{H}'; \text{absence}; \mathbf{F}'; \mathbf{H}'\ ]$ to evaluate the execution time of our approach across different trace lengths ranging up to 100 thousand. Moreover, to see how the formula influences performance we test it using formulas with increasing length, up to 20. Families $Formula3$ and $Formula4$ are introduced that each consists of a sequence with a repeated $\mathbf{H}'; \text{absence}\ \mathbf{F}'$ pattern as $\text{seq}[\ \mathbf{H}'; \text{absence}\ \mathbf{F}'; \mathbf{H}'; \ldots\ ]$ up to length 20, noting that $Formula4$ includes an outer time constraint of $[0, 5000]$. Both graphs in Figure 6 demonstrate the linear execution time of EmoSTL over different trace and formula lengths. While implementing the language literally as its semantics would result in polynomial execution time, our optimized implementation makes the execution time linear.

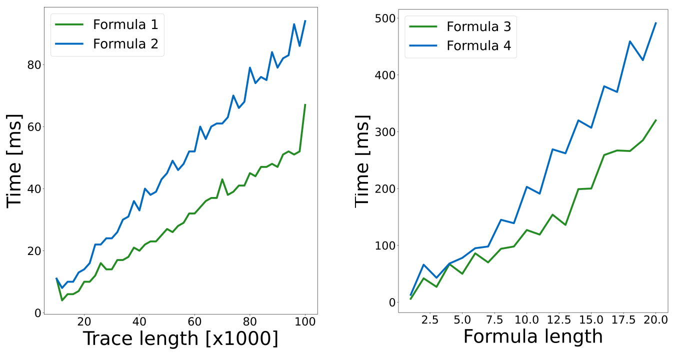

Fig. 6: EmoSTL performance tests

## VI. Related Work

In the realm of verifying functional aspects in games, some approaches have been proposed [42]–[44]. The recent shift from conventional functional testing to the evaluation of player experience has transformed this domain within game research. Kim and Doh [25] explore a new method to model players' emotional response patterns to story events in video games, by combining the OCC emotion structure and D. Price's emotional intensity equation [45]. The study compares the emotional response patterns of players in successful and unsuccessful video games. Successful games elicit more frequent emotion transitions, diverse emotion types, and higher intensity of emotions in players compared to unsuccessful ones. The authors introduce emotion transition pattern graphs and emotion transition distance graphs as visual tools to help game designers improve the structure of story events to evoke desired emotional responses in players. Gow et al. [46] evaluate player experience using the post-game commentaries. It involves players providing verbal reports by watching the playback of a recorded video of their gameplay. Commentaries are categorized into seven player experience dimensions, including challenging (easy-hard) and engaging (interested-bored) experiences. The approach allows scene-by-scene evaluation of the player experience and their overall share, in a single level of a third-person shooter game. Despite similarities between [46] and our research in capturing player experience feedback, the first does not include a specification language for expressing and verifying certain experiences. To address this issue, we establish a formal language for more accurate evaluation, helping in identifying design issues.

Towards expressing emotional requirements in the game domain, Callelle et al. [47] particularly focus on capturing emotional requirements in terms of the designer's intent. They employ emotional terrain maps in which emotions are expressed in terms of color codes in the game world, emotional intensity maps using luminance, and emotion timelines to visually express emotional requirements. The approach was later improved in [24] to provide a more comprehensive approach that includes e.g. the explicit identification of potential locations for emotion markers to ultimately create libraries of emotion prototypes to identify emotion patterns in the future.

However, their approach lacks a language for expressing spatial-temporal requirements with an automated verification mechanism to ensure the game meets its requirements. Miguéis et al. [48] propose a design of a DSL to model emotion requirements for the video game industry. The paper outlines the development of a meta-model though it lacks formal semantics, so the formal verification will not be possible. Miller et al. [49] introduce a notation of emotional goals into requirements models. They differentiate between personal emotional goals, which model the emotional desires of users, and context-specific emotional goals, which represent the desired effects of a system on its users. Results on the improved design of an emergency system that addresses the emotional needs of users show greater user satisfaction. Nevertheless, the study does not encompass spatial-temporal aspects, nor does it possess formal semantic support.

## VII. Conclusion & Future work

This paper introduces EmoTL, a Domain Specific Language based on LTL, extended with spatial information and time intervals to let game developers formally specify requirements of emotions and verify them on the game in the early development stage to receive fast feedback. The approach is evaluated with the software engineer-designed PX requirements which are then formalized and tested on a specific game. The study demonstrates EmoTL fulfills the majority of demands in translating designed requirements into specifications. Such an intuitive language for automated emotion verification assists in evaluating players' emotions, fine-tuning a game's difficulty, and ultimately contributing to the creation of a well-designed gaming experience. We aim to use EmoSTL for the evaluation of commercial games. *Acknowledgement:* This project is funded by the EU H2020 Programme grant nr. 856716.

## References

[1] A. Drachen, P. Mirza-Babaei, and L. E. Nacke, *Games user research*. Oxford University Press, 2018.

[2] G. N. Yannakakis and A. Paiva, "Emotion in games," *Handbook on affective computing*, vol. 2014, pp. 459–471, 2014.

[3] N. Lazzaro, "Why we play: affect and the fun of games," *Human-computer interaction: Designing for diverse users and domains*, vol. 155, pp. 679–700, 2009.

[4] E. D. Mekler, J. A. Bopp, A. N. Tuch, and K. Opwis, "A systematic review of quantitative studies on the enjoyment of digital entertainment games," in *Proceedings of the SIGCHI conference on human factors in computing systems*, 2014, pp. 927–936.

[5] M. Montola, "The positive negative experience in extreme role-playing," *The Foundation Stone of Nordic Larp (2010)*, vol. 153, 2010.

[6] M. V. Birk, I. Iacovides, D. Johnson, and R. L. Mandryk, "The false dichotomy between positive and negative affect in game play," in *Proceedings of the 2015 Annual Symposium on Computer-Human Interaction in Play*, 2015, pp. 799–804.

[7] J. Woodcock, P. G. Larsen, J. Bicarregui, and J. Fitzgerald, "Formal methods: Practice and experience," *ACM computing surveys (CSUR)*, vol. 41, no. 4, pp. 1–36, 2009.

[8] A. Pnueli, "The temporal logic of programs," in *18th Annual Symposium on Foundations of Computer Science (sfcs 1977)*. ieee, 1977, pp. 46–57.

[9] C. Baier and J.-P. Katoen, *Principles of model checking*. MIT press, 2008.

[10] F. M. Maggi, M. Westergaard, M. Montali, and W. M. van der Aalst, "Runtime verification of ltl-based declarative process models," in *International Conference on Runtime Verification*. Springer, 2012, pp. 131–146.

[11] I. Lee, S. Kannan, M. Kim, O. Sokolsky, and M. Viswanathan, "Runtime assurance based on formal specifications," *Departmental Papers (CIS)*, p. 294, 1999.

[12] A. Agarwal and A. Meyer, "Beyond usability: evaluating emotional response as an integral part of the user experience," in *CHI'09 Extended Abstracts on Human Factors in Computing Systems*. ACM New York, NY, USA, 2009, pp. 2919–2930.

[13] R. Alves, P. Valente, and N. J. Nunes, "The state of user experience evaluation practice," in *Proceedings of the 8th Nordic Conference on Human-Computer Interaction: Fun, Fast, Foundational*, 2014, pp. 93–102.

[14] A. P. Vermeeren, E. L.-C. Law, V. Roto, M. Obrist, J. Hoonhout, and K. Väänänen-Vainio-Mattila, "User experience evaluation methods: current state and development needs," in *Proceedings of the 6th Nordic conference on human-computer interaction: Extending boundaries*, 2010, pp. 521–530.

[15] (2006) Dungeons & dragons online: Back to basics. [Online]. Available: https://ddowiki.com/page/Back_to_Basics

[16] S. G. Ansari, I. Prasetya, D. Prandi, F. M. Kifetew, M. Dastani, F. Dignum, and G. Keller, "Model-based player experience testing with emotion pattern verification," in *International Conference on Fundamental Approaches to Software Engineering*. Springer Nature Switzerland Cham, 2023, pp. 151–172.

[17] R. Koymans, "Specifying real-time properties with metric temporal logic," *Real-time systems*, vol. 2, no. 4, pp. 255–299, 1990.

[18] A. Ortony, G. Clore, and A. Collins, "The cognitive structure of emotions. cam (bridge university press," *Cambridge, England*, 1988.

[19] J. A. Baier and S. A. McIlraith, "Planning with first-order temporally extended goals using heuristic search," in *AAAI*, 2006, pp. 788–795.

[20] W. M. Van Der Aalst and M. Pesic, "Decserflow: Towards a truly declarative service flow language," in *Web Services and Formal Methods: Third International Workshop, WS-FM 2006 Vienna, Austria, September 8-9, 2006 Proceedings 3*. Springer, 2006, pp. 1–23.

[21] M. C. Mayer, C. Limongelli, A. Orlandini, and V. Poggioni, "Linear temporal logic as an executable semantics for planning languages," *Journal of Logic, Language and Information*, vol. 16, pp. 63–89, 2007.

[22] G. De Giacomo, M. Y. Vardi *et al.*, "Synthesis for ltl and ldl on finite traces," in *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015*. AAAI Press, 2015, pp. 1558–1564.

[23] U. Sammapun, I. Lee, O. Sokolsky, and J. Regehr, "Statistical runtime checking of probabilistic properties," in *Runtime Verification: 7th International Workshop, RV 2007, Vancover, Canada, March 13, 2007, Revised Selected Papers 7*. Springer, 2007, pp. 164–175.

[24] D. Callele, E. Neufeld, and K. Schneider, "Visualizing emotional requirements," in *2009 Fourth International Workshop on Requirements Engineering Visualization*. IEEE, 2009, pp. 1–10.

[25] M. Kim and Y. Y. Doh, "Computational modeling of players' emotional response patterns to the story events of video games," *IEEE Transactions on Affective Computing*, vol. 8, no. 2, pp. 216–227, 2017.

[26] (last accessed October, 2023.) Lab recruits wiki. [Online]. Available: https://github.com/iv4xr-project/labrecruits/wiki

[27] (last accessed October, 2023.) Wooclap. [Online]. Available: https://www.wooclap.com/

[28] A. Mehrabian and J. A. Russell, *An approach to environmental psychology.* the MIT Press, 1974.

[29] J. A. Russell and A. Mehrabian, "Evidence for a three-factor theory of emotions," *Journal of research in Personality*, vol. 11, no. 3, pp. 273–294, 1977.

[30] A. Mehrabian, "Pleasure-arousal-dominance: A general framework for describing and measuring individual differences in temperament," *Current Psychology*, vol. 14, pp. 261–292, 1996.

[31] K. R. Scherer and B. Meuleman, "Human emotion experiences can be predicted on theoretical grounds: Evidence from verbal labeling," *PloS one*, vol. 8, no. 3, p. e58166, 2013.

[32] S. Asteriadis, K. Karpouzis, N. Shaker, and G. N. Yannakakis, "Towards detecting clusters of players using visual and gameplay behavioral cues," *Procedia Computer Science*, vol. 15, pp. 140–147, 2012.

[33] K. Kutt, D. Drkażyk, L. Żuchowska, M. Szelkażek, S. Bobek, and G. J. Nalepa, "Biraffe2, a multimodal dataset for emotion-based personalization in rich affective game environments," *Scientific Data*, vol. 9, no. 1, p. 274, 2022.

[34] C. van Reekum, T. Johnstone, R. Banse, A. Etter, T. Wehrle, and K. Scherer, "Psychophysiological responses to appraisal dimensions in

a computer game," *Cognition and emotion*, vol. 18, no. 5, pp. 663–688, 2004.

[35] C. Bassano, G. Ballestin, E. Ceccaldi, F. I. Larradet, M. Mancini, E. Volta, and R. Niewiadomski, "A vr game-based system for multimodal emotion data collection," in *Proceedings of the 12th ACM SIGGRAPH Conference on Motion, Interaction and Games*, 2019, pp. 1–3.

[36] S. Mascarenhas, M. Guimarães, R. Prada, P. A. Santos, J. Dias, and A. Paiva, "Fatima toolkit: Toward an accessible tool for the development of socio-emotional agents," *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 12, no. 1, pp. 1–30, 2022.

[37] S. C. Marsella and J. Gratch, "Ema: A process model of appraisal dynamics," *Cognitive Systems Research*, vol. 10, no. 1, pp. 70–90, 2009.

[38] S. G. Ansari, I. Prasetya, M. Dastani, F. Dignum, and G. Keller, "An appraisal transition system for event-driven emotions in agent-based player experience testing," in *International Workshop on Engineering Multi-Agent Systems*. Springer, 2021, pp. 156–174.

[39] M. Shvo, J. Buhmann, and M. Kapadia, "An interdependent model of personality, motivation, emotion, and mood for intelligent virtual agents," in *Proceedings of the 19th ACM international conference on intelligent virtual agents*, 2019, pp. 65–72.

[40] A. S. Rao and M. P. Georgeff, "Decision procedures for bdi logics," 1998.

[41] I. Prasetya, M. Dastani, R. Prada, T. E. Vos, F. Dignum, and F. Kifetew, "Aplib: Tactical agents for testing computer games," in *International Workshop on Engineering Multi-Agent Systems*. Springer, 2020, pp. 21–41.

[42] A. Yacoub, M. E. A. Hamri, and C. Frydman, "Dev-promela: modeling, verification, and validation of a video game by combining model-checking and simulation," *Simulation*, vol. 96, no. 11, pp. 881–910, 2020.

[43] R. Rezin, I. Afanasyev, M. Mazzara, and V. Rivera, "Model checking in multiplayer games development," in *2018 IEEE 32nd International Conference on Advanced Information Networking and Applications (AINA)*. IEEE, 2018, pp. 826–833.

[44] O. Tekik, E. Surer, and A. B. Can, "Verifying maze-like game levels with model checker spin," *IEEE Access*, vol. 10, pp. 66 492–66 510, 2022.

[45] D. D. Price and J. J. Barrell, "Some general laws of human emotion: Interrelationships between intensities of desire, expectation, and emotional feeling," *Journal of personality*, vol. 52, no. 4, pp. 389–409, 1984.

[46] J. Gow, P. Cairns, S. Colton, P. Miller, and R. Baumgarten, "Capturing player experience with post-game commentaries," in *Proc. 3rd Int. Conf. on Computer Games, Multimedia & Allied Technologies*, 2010.

[47] D. Callele, E. Neufeld, and K. Schneider, "Emotional requirements in video games," in *14th IEEE International Requirements Engineering Conference (RE'06)*. IEEE, 2006, pp. 299–302.

[48] G. Miguéis, J. Araujo, and A. Moreira, "Towards a requirements language for modeling emotion in videogames," in *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, 2019, pp. 1878–1880.

[49] T. Miller, S. Pedell, A. A. Lopez-Lorca, A. Mendoza, L. Sterling, and A. Keirnan, "Emotion-led modelling for people-oriented requirements engineering: the case study of emergency systems," *Journal of Systems and Software*, vol. 105, pp. 54–71, 2015.