Intelligent Verification/Validation for XR Based Systems

Research and Innovation Action

Grant agreement no.: 856716

D4.1 – 1st prototype of SETAs

iv4XR – WP4 – D4.1

Version 1.5

December 2020

Project Reference	EU H2020-ICT-2018-3 - 856716
Due Date	31/12/2020
Actual Date	30/12/2020
Document Author/s	Manuel Lopes, Marta Couto, Pedro Fernandes, Carlos Martinho, Rui Prada (INESC-ID), Ian Saunter (GW), Davide Prandi (FBK)
Version	1..5
Dissemination level	Public
Status	Final

This project has received funding from the European Union's Horizon 2020 Research and innovation programme under grant agreement No 856716

Document Version Control			
Version	**Date**	**Change Made (and if appropriate reason for change)**	**Initials of Commentator(s) or Author(s)**
1.0	03/12/2020	Initial document structure and contents	ML
1.1	10/12/2020	Added details about the appraisal model for UX assessment	PF
1.2	11/12/2020	Improvements to the document's structure and introduction	RP, MC
1.3	14/12/2020	Added details about the cognitive load	MC, CM
1.4	21/12/2020	Improvements in the document structure	RP
1.5	30/12/2020	Final arrangements before submission	RP, MC

Document Quality Control			
Version QA	**Date**	**Comments (and if appropriate reason for change)**	**Initials of QA Person**
1.3	16/12/2020	Document review	DP
1.3	18/12/2020	Document review	IS

Document Authors and Quality Assurance Checks		
Author Initials	**Name of Author**	**Institution**
ML	Manuel Lopes	INESC-ID
MC	Marta Couto	INESC-ID
PF	Pedro Fernandes	INESC-ID
CM	Carlos Martinho	INESC-ID
RP	Rui Prada	INESC-ID
DP	Davide Prandi	FBK
IS	Ian Saunter	GWE

TABLE OF CONTENTS

1. EXECUTIVE SUMMARY

This document is a software deliverable. We provide an overview of the state of the work on WP4, software links, and its description. We annexed papers and reports that describe in more detail the development and results of the software.

Acronyms and Abbreviations

SETA	Socio-Emotional Test Agent
SUT	System Under Test
XR	Extended Reality
UX	User Experience
RL	Reinforcement Learning

2. INTRODUCTION

Work Package 4 focuses on developing socio-emotional test agents (SETAs) to aid the systematic assessment of user experience (UX) of extended reality (XR) systems. The SETA's will use well-established emotion appraisal theories and models to assess the emotional state; they will cover social variability to simulate a wide range of users; and use a progression model allowing it to appraise UX-relevant user states over time.

To achieve these goals we need to build several components including, understanding of the social and emotional experience of a user, creating profiles of the social properties of different types of users, studying the impact of the different scenarios/interactions in different types of people, and then using such profiles to test software automatically.

We are developing methods for the automatic creation of typical profiles from user interaction. These profiles can then be employed either for personalisation of the interaction or to be able to generate automatic tests that are more representative of the different profiles of users.

During the first period of the project, two tasks are active. Task 4.1 SETA Design and Development and Task 4.2 Socio Components. Task 4.5 has just begun, however, we do already have some processes regarding the integration with the Framework.

One of WP4 primary outcomes is the SETA Design and Development, which aims to develop the core Socio-Emotional Test Agents. Work on this task is on track, and we have integrated prototypal SETAs with the iv4XR Framework. Originally, the SETAs were to be based on the FAtiMA Toolkit to make use of its cognitive model of emotions. However, we are first exploring the use of simpler dimensional theories of emotion, such as the PAD and the Core Affect Model. We felt the need to complement the FAtiMA toolkit with dimensional emotional models since the OCC model (Clore & Ortony, 2013), on which the FAtiMA toolkit is based, requires a considerable understanding of the relations between agents and objects. This understanding might not always be available to the agent, depending on the system under test (SUT). Dimensional emotional models can more easily be used to work without such understanding, and their continuous nature is well suited for modelling the evolution of emotional states through time and location.

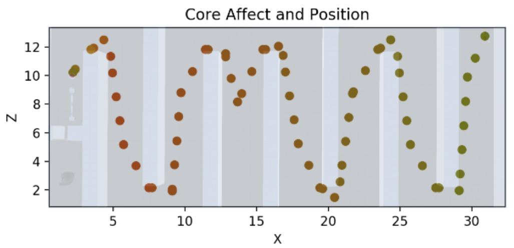
In this period the work was divided into three main components:

- study and development of models of user experience and social components of interaction
- study and development of models of difficulty estimation of scenarios
- study and development methods for modelling user profiles

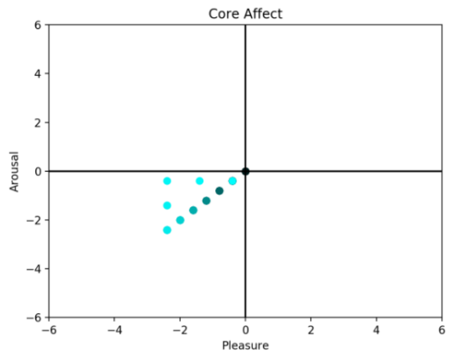
3. SOFTWARE INCLUDED IN THIS DELIVERABLE

3.1 Appraisal model for UX assessment

To understand and develop an appraisal model we carried out a systematic survey of user experience evaluation [2] and informed with such research we created our first prototype described in [3] with the code in [1] integrated with the currently developed Framework and the main Lab Recruits demo (developed as a first testbed for the iv4XR Framework). In its current version, the software uses a rule-based approach to model the two dimensions of the Core Affect of an agent that traverses a level/scenario. This approach allows designers and testers to have an estimation of UX through time and position of a scenario, allowing them to pinpoint troublesome areas that require improvements. Currently, we are integrating bio-sensors to be able to measure physiological parameters during an interaction, allowing us to improve and validate this approach. Figure 1 shows a maze in lab recruits and we see in Fig.1a) the locations where the affect was higher and in Fig.2b) the evolution of arousal/pleasure during the trajectory.



(a)



(b)

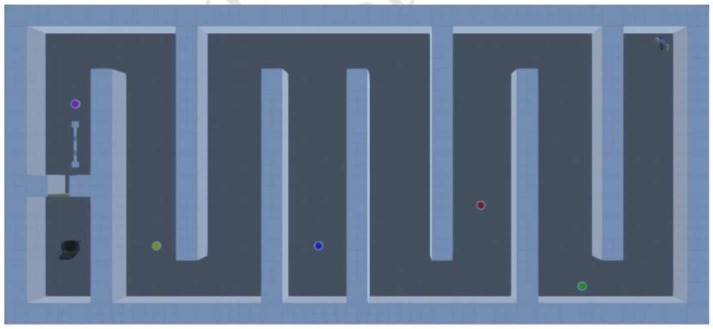


Figure 1: (Top) Estimation of affect variables (Arousal and Pleasure), (Bottom) Level.

Links and References

Source code: [1] https://github.com/iv4xr-project/userexperienceeval

Video: https://www.youtube.com/watch?v=UOdwZX2h11o&t=12s

Expected Publications:

[2] Fernandes, P., Ansari, S. G., Prasetya,I. S. W. B., Lopes, M., Martinho, C., Dignum, F., Prada, R. (2021) *Automated UX Testing: A Call for Research*. Manuscript in preparation. Annex A1.

[3] Fernandes, P., Lopes, M, Prada, R. (2021) *Agents for Automatic User Experience Testing*. Manuscript in preparation. Annex A2.

3.2 Difficulty estimation

The difficulty of a level, and more importantly, the progression of difficulty of a series of levels, can have a significant impact on the user experience and learnability of a game. We are developing methods to rank a series of levels in terms of the difficulty. As an example Figure 2 shows a character (rectangle) that needs to jump over an obstacle, the length of the obstacle and the difference in height will make this obstacle more or less difficult. Preliminary results can be seen in [4] (software available at [5]) and show that difficulty can be estimated in many cases. This is currently being tested and developed in a platform game.

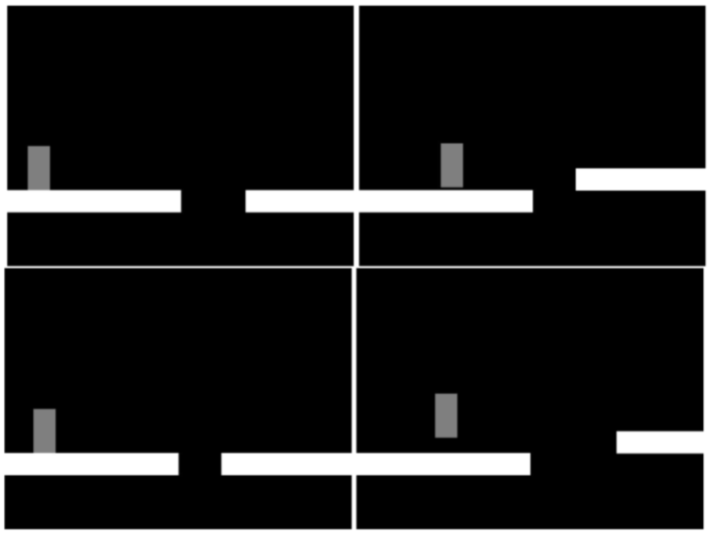


Figure 2: Levels used for testing the estimation of difficulty. Length and difference in heights will make an obstacle more difficult. We aim at automatically order the levels in terms of difficulty.

Links and References

Source code: [5] https://github.com/iv4xr-project/difficultysch

Expected Publications:

[4] Miguel Reis, Manuel Lopes, Rui Prada, Automatic Evaluation and Ordering of Level Difficulty in Games. Manuscript in preparation.

3.3 Automated Cognitive-Load Estimation

A significant part of XR user experience results from the interaction of each user with the system while solving a task in the environment. During interaction with an XR system, each user has different exchanges while navigating through the design space of the system, which will create distinct user experiences (UX). Aside from the emotions and social motivations, cognitive abilities also impact appraisal, and, indirectly, how the user navigates the space. Being able to understand how the interactions are a reflection of these dimensions of the experience would help designing more personalised systems, resulting in better UX.

This work focuses on Cognitive Load, meaning the amount of information a person is conscientiously processing at a given moment. The cognitive load has a close relationship with attention mechanisms. We aim to create a toolset in the context of automatic play-testing that we can extend to other types of systems. The toolset, based on the TBRS (Time-Based Resource Sharing) [6] memory model, aims at providing a measure of the cognitive load (a percentage) that the user is expected to experience while going through a particular task in a specific context. Autonomous agents navigating and exploring a virtual environment need some parameterisation of what grabs the agents' attention (e.g., interacting with an object or dodging enemies' attacks). We call these "attention-grabbing events", measured in seconds (duration of the event). The toolset will provide a set of methods (API) that need to be added/called from the code of the software undergoing testing, each time an attention-grabbing event occurs.

After finalising a task, the toolset will compute an estimate for the cognitive load experienced by the user, based on the sum of the attention-grabbing events, and the total duration of the task. We can integrate this value into automated testing procedures by using assertions in the code that check whether the estimated value of the cognitive load is within a specific desired range. If not, the toolset will be able to present a short report to identify possible problems with the current implementation (based on the data gathered) and shed some light on the direction to take in future development.

To test and evaluate our model, we created the game "Way-out" (Figure 3), in which the player is escaping from a small underground complex. We designed the game to allow for the parameterisation and control of several attention-grabbing events, and the manipulation of several dimensions of the experience, such as the time required to navigate through the game, the complexity of the task based on the number of interactions required to overcome them, as well as the number of items a player needs to keep in mind to solve the different puzzles. By comparing the reported cognitive load of the participants to the value computed by the TBRS theoretical model, we aim at evaluating whether this model is an appropriate predictor of the cognitive load reported by the players for the whole experience.

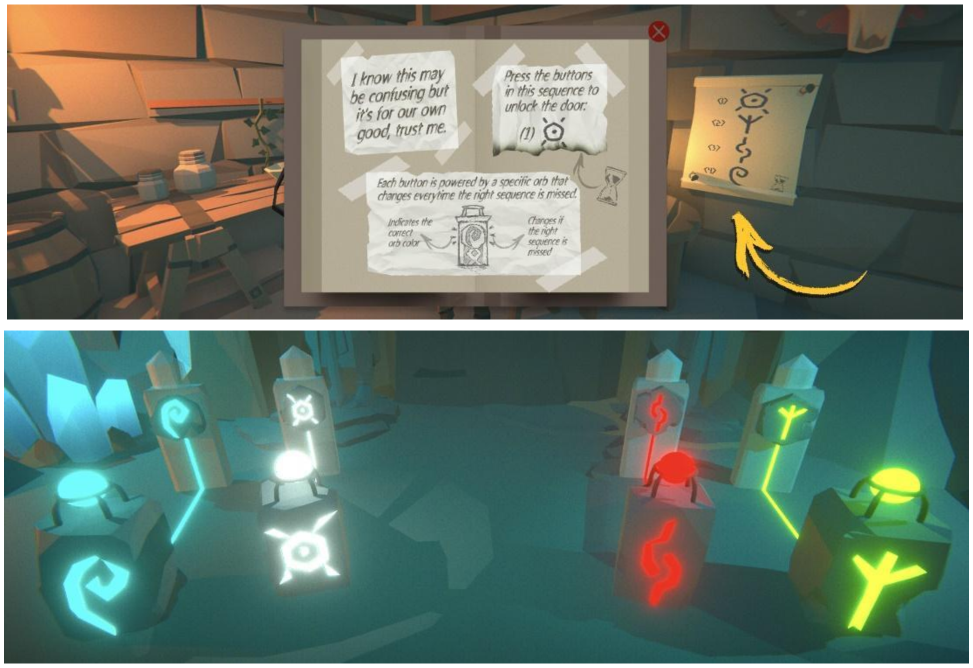

I know this may be confusing but it's for our own good, trust me.
Press the buttons in this sequence to unlock the door:
(1)
Each button is powered by a specific orb that changes everytime the right sequence is missed.
Indicates the correct orb color
Changes if the right sequence is missed

Figure 3: The 'Way out' game

Links and References

References:

[6] Barrouillet, P., & Camos, V. (2007). The time-based resource-sharing model of working memory. *The cognitive neuroscience of working memory*, *455*, 59-80.

Project Github:
https://github.com/albertoramos1997/WayOut
Video(s):
Vídeo Way Out (Lever Puzzle): The video shows the four versions of the "lever puzzle". In this puzzle, the player needs to collect the missing levers, add them to a machine, and activate the correct levers to open the door to the next room. The video also shows the inventory (backpack) and the notebook, where players can store hints about the different puzzles.

https://www.youtube.com/watch?v=6LSi81yiB28&t=18s&ab_channel=AlbertoRamos

Vídeo Way Out (Orb Puzzle): The video shows the four versions of the "orb puzzle", which is the final puzzle of the game. Throughout the rooms, the player finds different orbs in stands. Each orb has a different colour and each stand has a symbol. The player has four stands with symbols on the final room, and they need to match the symbols to the colours. Once all orbs are in the correct stand, the player needs to activate the buttons (by pressing the symbols) in a predefined order to win the game.

https://www.youtube.com/watch?v=8ge565wPE9I&t=21s&ab_channel=AlbertoRamos

Expected Publications:
[7] Ramos, A., Couto, M., Martinho, C. (2020). *Assessing Players' Cognitive Load in Games*. Manuscript in preparation. Annex A3

3.4 Verification of interaction properties

When a designer creates a level, there is an intention to produce specific properties. For instance, a game for training teams must verify that each scenario enables and allows teamwork. We developed the first version of a methodology to verify properties of scenarios that can be described based on desired behaviours or example behaviours. The approach uses machine learning to learn the best behaviour of a set of agents automatically and then compare the estimated policy with a set of baseline policies to decide which of the baseline policies the agents are using. The baseline policies define the design goals of the system under test.
We focused on determining if a scenario promotes cooperation [7] and plan to consider other types of properties [8]. We are conducting a user study to verify the simulation results.

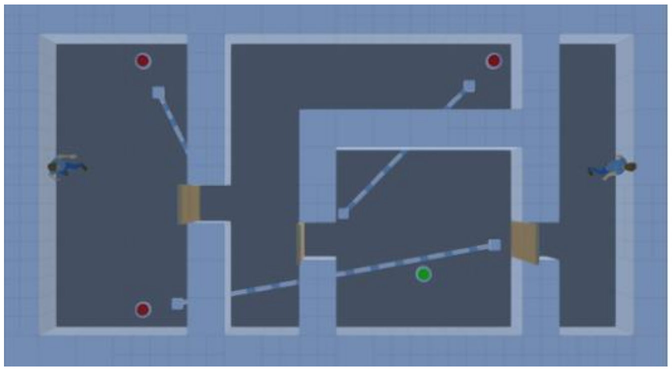


Figure 4: Evaluation of how a level enables collaboration between agents.

Links and references

Project Github:

https://github.com/carreirabruno/Tese_iv4XR_Pessoal

Expected publications

[7] Bruno Carreira, Manuel Lopes, Rui Prada Collaboration analysis in a multi-player based simulation. Manuscript in preparation. Annex A4.

[8] Luis Fernandes, Manuel Lopes, Learning User Profiles for Automatic Test of Games. Manuscript in preparation.

4. CONCLUSIONS AND FUTURE WORK

In this period we developed several methods to measure and evaluate various aspect of the experience of the user during gameplay. We considered cognitive workload and affect/arousal. Even before gameplay we developed methods to validate a set of scenarios in terms of gameplay properties such as capability to play as a team, learnability and difficulty.

For the second period of the project we plan to integrate all the components developed so far into the main Framework of the project. We will expand the specification of emotional experience and its automatic verification on more complex scenarios.

ANNEXES

ANNEX A1

Automated UX Testing: A Call for Research

PEDRO M. FERNANDES*, INESC-ID and Instituto Superior Técnico, Univ. de Lisboa, Portugal
SABA GHOLIZADEH ANSARI*, Utrecht University, Netherlands
I. S. W. B. PRASETYA, Utrecht University, Netherlands
MANUEL LOPES, INESC-ID and Instituto Superior Técnico, Univ. de Lisboa, Portugal
CARLOS MARTINHO, INESC-ID and Instituto Superior Técnico, Univ. de Lisboa, Portugal
FRANK DIGNUM, Umea University, Sweden
RUI PRADA, INESC-ID and Instituto Superior Técnico, Univ. de Lisboa, Portugal

User eXperience (UX) is highly important in any application that interacts directly with the end user and merits being an integral part of the iterative development process. Functional testing has been highly automated and GUI testing is taking steps on that direction, yet UX testing is largely dependent on user based testing. This leads to financial and time constraints that turn UX testing into a bottleneck, forcing developers to mostly exclude UX concerns from frequent test runs. Thus, we believe it would be greatly beneficial to automate UX testing. In this paper we begin by presenting a systematic review on automated UX testing based on agents and affective/cognitive models, finding no works that satisfy our inclusion criteria. Considering that this represents a research gap, we further conduct a scoping review and present 6 areas of research that could serve as the foundation for the development of affective/cognitive testing agents.

CCS Concepts: • **Human-centered computing** → **Human computer interaction (HCI)**; *User models*; • **General and reference** → *Surveys and overviews*; • **Computing methodologies** → *Artificial intelligence.*

Additional Key Words and Phrases: user experience, automation, software testing, automated testing, affective agents

ACM Reference Format:
Pedro M. Fernandes, Saba Gholizadeh Ansari, I. S. W. B. Prasetya, Manuel Lopes, Carlos Martinho, Frank Dignum, and Rui Prada. 2020. Automated UX Testing: A Call for Research. 1, 1 (December 2020), 16 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

User eXperience (UX) is a hard to define concept. A survey approach to the problem of user experience [42], asking 275 researchers and practitioners from academia and industry which definition of UX they most agreed with, found the most agreed upon definition to be: "A consequence of a user's internal state (predispositions, expectations, needs, motivation,

*Both authors contributed equally to this research.

Authors' addresses: Pedro M. Fernandes, pedro.miguel.rocha.fernandes@ist.utl.com, INESC-ID and Instituto Superior Técnico, Univ. de Lisboa, Portugal, Lisboa; Saba Gholizadeh Ansari, s.gholizadehansari@uu.nl, Utrecht University, Netherlands; I. S. W. B. Prasetya, Utrecht University, Netherlands; Manuel Lopes, INESC-ID and Instituto Superior Técnico, Univ. de Lisboa, Portugal, Lisboa; Carlos Martinho, INESC-ID and Instituto Superior Técnico, Univ. de Lisboa, Portugal, Lisboa; Frank Dignum, Umea University, Sweden; Rui Prada, INESC-ID and Instituto Superior Técnico, Univ. de Lisboa, Portugal, Lisboa, rui.prada@tecnico.ulisboa.pt.

mood, etc.) the characteristics of the designed system (e.g. complexity, purpose, usability, functionality, etc.) and the context (or the environment) within which the interaction occurs (e.g. organisational/social setting, meaningfulness of the activity, voluntariness of use, etc.) [25]". In most situations, system designers have no control over the user's internal state prior the interaction with the system nor the context (environment) in which the system is used. As such, for the purpose of this work, when we mention UX we are referring to the consequences the characteristics of a system have on the internal state of a user. This is in accordance with the ISO 9241-210:2019 definition of UX: "user's perceptions and responses that result from the use and/or anticipated use of a system, product or service." [15]. The first note to that definition clarifies that "Users' perceptions and responses include the users' emotions, beliefs, preferences, perceptions, comfort, behaviours, and accomplishments that occur before, during and after use.".

User questionnaires, and more broadly user testing, are commonly used when evaluating UX [1, 66]. When done properly, these methods can provide numerous insights to developers and designers and be a fundamental part of the iterative cycle of product development. Unfortunately, they also prove to be a bottleneck. User testing is a relatively slow endeavour in a world of sprint runs and agile development. Advances in automatic software testing have allowed the functional testing of software to rely less on user testing and more on software tools, accomplishing in minutes what previously took days. UX testing lags behind and so is left behind. The financial and temporal costs inherent to current UX testing methods force many companies and developers to ignore UX concerns, which is neither beneficial for the companies nor the end user.

We, therefore, argue that there is a strong need to add automation to UX testing. By this we do not mean that testing methods that rely on users (user testing) should be completely discarded. Developers should, however, be empowered with tools that allow them to quickly estimate certain dimensions of UX, allowing for quick iterative development cycles that also take UX into consideration. For example, in game development, if a new version of the software inadvertently makes a key action become much more complex and cumbersome, makes a location harder to reach or makes the game more repetitive, the developer should be alerted of a negative UX effect when doing routine testing.

To automatically assess how certain environments affect UX, we can employ an internal model of users. This can be done by artificially emulating a user or using narrower models that focus on a single interaction metric. The autonomous and self-motivated behaviour of artificial agents make them strong contenders for automated UX experience testing. The field of Artificial Intelligence (AI) is constantly evolving. We are able to program agents which are capable of accomplishing increasingly complex tasks, some of which were believed to be impossible a few decades ago [5]. Other areas of computer science have also taken large leaps forward. How have such advances impacted UX testing? We now have agents that are able to drive cars in complex and unpredictable environments. Using these technological improvements, can we have agents that interact with a system and give us an estimate of UX measures? Can we apply affective and cognitive models, developed in the area of psychology, to estimate the expected emotional states or cognitive load of users?

This paper was motivated by these questions. Our first step was to understand if any attempts had been made to use agents and affective/cognitive models to estimate UX. To do so, we conducted a systematic review (Sec. 4.1). Throughout this review, we were unable to find any publications that met our inclusion criteria (Tab. 1). As far as we could find, there have been no attempts to implement artificial users or affective/cognitive models in order to automatically estimate UX measures.

We believe agent based UX testing to be a very promising field that can greatly improve software development, especially in interactive applications areas, such as gaming, simulation and extended reality. As such, we decided to further conduct a scoping review [55] in order to find which current areas of research are most aligned with the objective of creating automated methods for UX testing. By doing so, we strive to collect and present a strong foundation that can be used for future research and development in the area of automated UX testing.

This paper is organized as follows. In Sec. 2 we give a brief overview of previous surveys on the area of UX research. In Sec. 3 we provide an overview of non-automated UX testing and give a short description of the affective and cognitive models that will be mentioned further in the paper. In Sec. 4 we describe the methodology used to conduct our review process. In Sec. 5 we present the findings of our review, enumerating 6 relevant areas of research and providing short descriptions for a number of relevant publications. Finally, in Sec. 6 we summarize the state of the art, discuss the results and present our position regarding future research in the area.

2 PREVIOUS SURVEYS

Previous surveys have provided an overview of the current state of the art of UX research, although we found none that focused on automated methods exclusively. Vermeeren et al. [66] identified 96 UX evaluation methods until 2010, characterized according to, among others, type of collected data, information source, evaluated type of application, location (lab or field), and type of the method used. Because psycho-physiological data may not be enough for UX evaluation, methods that use these data together with other types of data such as users' comments are also included in the survey. About one-third of these investigated methods used quantitative data, one-third qualitative data and one-third both. The study indicates that although one-third of all methods (37 methods) could be used in the early development phases, there is still a demand for more effective early stage UX evaluation methods, multi-method approaches, improving the validity of measured-based methods and establishing practicability and scientific quality.

Later, Rivero and Conte [57] continue the work on UX methods by performing a systematic mapping study on technologies developed between 2010 and 2015. It interestingly points out the need for gathering UX data of application usage at different time stamps, while considering specific features or types of applications. Assessing UX before, during and after episodic experience of the user provides richer information on the aspects that affect their experience, helping developers identify user requirements. The users or the development team, however, may not have available time to carry out a long period evaluation.

Alves et al. [1] conducted an online survey in which participants were asked about the UX evaluation techniques they use, UX evaluation characterization, and impacts on different level of software engineering and field of their expertise. The survey demonstrates that informal, low cost methods, such as observation, think aloud, contextual inquiry, interview and experience prototyping are widely used. Methods that involve high cost and technological challenges such as eye-tracking are rarely applied to UX evaluation by the participants. This study also illustrates that only in 50% of cases, end-users are always involved in evaluating UX. However,in the remaining half, subjects are designers, software engineers and directors. This implies that end-users are not elaborated enough on UX assessment. Alves et al. highlight that the way the end-users perceive the products and interact with software is more important than the actual usability of the software.

Considerable research has been performed in order to study attitudes toward UX measurement. Law [41] argues that UX researchers and practitioners may roughly be divided into two camps: the "design-based UX camp" that focuses more on qualitative approaches, and the "model-based camp" with a focus on quantitative approaches. Employing quantitative measures with complete exclusion of qualitative accounts of user experiences may result, however, in wrong implications. To enhance the acceptance of UX measures, UX modelling should, therefore, be grounded in psychological theories to link experiential qualities with outcomes, such as UX theoretical frameworks investigating the relationship between affect, action, and cognition[43].

Given the current lack of surveys on automated UX testing, we believe a survey on automated UX testing could pave the way for future research, being a valuable source of information and inspiration, in an area we argue requires more research.

3 BACKGROUND

Under the UX umbrella, various types of experiential qualities such as flow, immersion, long-term engagement, learning and usability can be measured. To measure these factors, methods like self-reporting, physiological responses, user modeling and objective assessment techniques have been used used.

We begin this section by giving a quick overview of what non-automated approaches are commonly used for UX testing. We then briefly describe the affective/cognitive model theories that will be relevant to understand some of the UX testing methods that will be mentioned further ahead in the paper.

3.1 Non-automated UX testing

Classically, UX testing is mostly done manually, employing user testing (testing done with real users) and self-reporting techniques. Examples of these non-automated self-reporting methods are 3E [64], MAX[9], Feeltrace[12], and UX Curve[38].

The 3E (Expressing Experiences and Emotions) [64] method is used during field studies to collect information about users' experiences and emotions in a semi-structured way. Users follow a template where they draw and write their experiences and emotions regarding their interaction with the evaluated application during the field study.

MAX [9] is a post-use method that uses a set of cards and a board for evaluating the general experience. MAX can be applied after the use of mock-ups, prototypes, interactive systems, or any artifact that user can interact with. The MAX cards allow evaluating UX in terms of four categories: emotion, ease of use, usefulness and intention to use, which are represented on the board by questions that guide the user at the evaluation. After experimenting with the evaluated artifact, the user selects cards and then places them on the board. Each card contains a human cartoon to portray the emotional reaction of users to the system. In the end, the evaluator analyzes the cards selection and conducts an interview to gather further information regarding the reason of choices.

Another approach to assess UX based on self-reporting is to observe the emotional dynamism over speech with support tools such as Feeltrace [12]. This tool records the perceived emotional content of speech. The software tool is designed to collect self-reports from observers watching the stimulus to be evaluated by allowing users to continuously label the change of affective expressions.

Unlike other methods evaluating UX in a short period of time, UX Curve method [38] was proposed to facilitate the assessment of UX changes over time. The UX Curve method aims at assisting users in retrospectively reporting how and why their experience with a product has changed over time. In particular, it investigates how specific memories of user experiences with mobile phones guide their behavior and their willingness to recommend the product to others. The participant draws one or more curves to describe how the experience about a product has changed over time. The curve drawing area is formed by a timeline and a horizontal line that divides positive and negative experiences. Additional horizontal and vertical lines can be used as more precise scales for the quality of experience and for time periods. Users are also asked to fill in a questionnaire giving the overall evaluation of the phone as well as willingness to recommend the phone model to a friend. According to statistic analysis, satisfied users draw an improving trend in UX curve and they report a higher likelihood of recommending the products. While UX Curve is a pen-and-paper method, there is a sketching tool called iScale [32] which follows the same technique to reconstruct longitudinal users experience data.

3.2 Cognitive Models

As cognitive models will be the foundation of a number of a number of works we will later discuss, in this section we shortly present the basis of cognitive modelling and give an overview of the specific approaches that will be used in works presented in Sec. 5.

Emotional experiences and their connections to other components of UX have been investigated over the years using physiological sensing to evaluate user experience including emotional reactions, stress levels, cognitive performance, and user engagement. For example, Mahlke and Thüring [46] proposed a method which integrates major components of user experience and their possible interrelations to investigate the influence of instrumental as well as non-instrumental qualities on emotional reactions, employing a combination of physiological methods, questionnaires and behavioral data. While instrumental qualities focus on the amount of support the system provides, such as the controllability of system behavior and the effectiveness of its functionality, non-instrumental qualities concern the look and feel of the system, such as, visual aesthetics or haptic feel. The results indicated that instrumental and non-instrumental qualities influence emotional reactions with respect to subjective feelings, facial expressions and physiological responses.

Bargas-Avila and Hornbæk [3] described eight dimensions for user experience assessment in a review on empirical studies of UX. Among them, emotions, enjoyment and aesthetics are the most frequently assessed UX dimensions. To confirm the impact of emotion on UX, Saariluoma has conducted several research on conceptualizing the emotional dimension of UX in psychological terms. In [59], Saariluoma and Jokine investigated the psychology of user experience based on an emotional theory to study which emotional dimensions are most relevant for user experience and should be addressed in UX testing. The essential role of emotion, however, is so far neglected in automated UX testing frameworks.

In addition to physiological approaches, it is possible to address the emotional dimension in UX testing by providing an emotional model to recognize and understand emotions. Emotional models can be formulated through computational cognitive modeling. Computational cognitive models define the essence of cognition and various human cognitive functions, such as perception and emotion, by representing them in computational models of mechanisms and processes[63].

Among various cognitive models, only two types, according to our review, have been used in assessing user experience: 'emotional models' and 'human processing models'. While emotional models are used to assist the assessment of the emotional dimensional user experience, Human Processing Models evaluate the usability in the early steps of the development process by estimating the cognitive burden needed to execute tasks on an interface. Both these cognitive models are going to be explained in the subsections below. Later on, we will present the current use of these models in UX testing (Sec. 5) and discuss that user emotion modeling has the potential to be used in automated UX testing, opening avenues for further research (Sec. 6).

3.2.1 Emotional Models. The causal analysis between cognition and emotion supports the theory that emotional reactions depend on cognitive appraisal. There have been various theories that have examined the appraisal criteria leading to emotions and the cognitive functions involved. Appraisal theory [44] suggests that before the occurrence of emotions, there are certain cognitive processes that analyze stimuli. Appraisal theories argue that *appraisal* process refers to the subjective perception, interpretation and evaluation of an event which results in emotions. This, then, guides the *coping* process to give appropriate responses based on subject's coping strategies. For instance, a human might "feel" distress regarding a specific event (appraisal). This leads to a shift of blame (coping), which results in anger (re-appraisal). Needless to say, people respond to events differently depending on how they are appraised. Computational models of emotion can ease further research on systems that try to model or influence human behavior [22].

A popular appraisal-based emotional model is the **OCC model** by Ortony, Clore and Collins [54]. This model proposed a hierarchy that classifies 22 type of emotions. At the most abstract level, OCC model considers every emotion as a valenced reaction, which can either be positive or negative. These emotions emerge as a consequences of events (e.g., joy and pity), actions of agents (e.g., pride and reproach), and aspects of objects (e.g., love and hate). This model is commonly used in developing emotional agents and user modeling, making the OCC model a good candidate to model emotional dimensions of user experience.

3.2.2 Human Processing Models. Several cognitive models have been used to evaluate usability by estimating the cognitive processes needed to execute tasks on an interface. The most commonly used methods are the Model Human Processor (MHP), the Goals, Operators, Methods, and Selection rules (GOMS), and the Key-stroke Level Model (KLM), which are all related and were proposed by Card, Moran, and Newell [8]. MHP is used to estimate the time for performing a specific task. It is defined by a set of specifications and principles of operation over models consisting of several types of interconnected processors (perceptual, cognitive, and motor) and memory systems (visual and auditory image storage, working memory, and long-term memory).

MHP formed the basis for developing GOMS (Goals, Operators, Methods, and Selection rules) family. GOMS were first proposed to describe human information processing in terms of basic perceptual, cognitive, and motor abilities. Like MHP, the family of GOMS models aim at predicting the execution time of perceptual-motor processing tasks by summing the time requirement of an operator sequence for completing a task with a system at hand. GOMS models could be used to predict some aspects of usability, such as, consistency and efficiency of designed interactive web forms and may effectively complement user testing in early design stages, reducing the amount of testing that may be necessary for usability evaluation as well as development costs [34].

The simplest and most frequently used GOMS technique is the Key-stroke level model (KLM) which predicts the time that will take for a skilled user to execute routine tasks with a User Interface (UI). KLM contains six operators which are: key-stroking; pointing with a mouse to a target; homing the hand at a keyboard; drawing a line segment on the grid; mentally preparing for executing actions; and the response time of the system. Other major GOMS techniques (CMN-GOMS, NGOMSL, and CPM-GOMS) require extensive training and familiarity with Human-Computer Interaction principles to perform an analysis [34].

4 REVIEW METHODOLOGY

4.1 Systematic Review

We began our review process by doing a systematic review, following the existent guidelines for systematic reviews in software engineering [36]. A systematic review is motivated by a precise research question, follows a defined search algorithm, examines all documents found by the search algorithm and has explicit criteria of paper inclusion/exclusion. This allows the review process to be thorough and as replicable as current search engines allow.

The first step of our systematic review was to define a concrete research question. We aimed to find research that automatically estimated UX without relying on user testing. During preliminary searches, we became aware that physiological and behavioural approaches often described their work as "automated UX testing" despite heavily relying on user testing. Such works were not our main focus, so we decided to narrow our research to approaches that modelled the ways users think and feel to avoid works that rely on real users during testing. With these details in mind, we defined the following research question: "What is the state of the art of automatic user experience testing using affective or cognitive models?".

Having defined our research question, the second step of our systematic review was to define a search algorithm to find relevant publications. Our aim was to define a research string, that is, a string of keywords and operators that could be used as a query in search engines. To do so, we began by doing preliminary searches looking for previous reviews. None were found that related to our particular research question. We then did several trial searches with combinations of many different keywords trying to assess which keywords and keyword combinations allowed us to find the greater number of potentially relevant publications.

We tried many combinations and variations of the terms: "automatic"; "user experience"; "testing"; "affective"; "cognitive"; and "model", judging such combinations by the number of potentially relevant papers they retrieved. We also experimented with using both "user experience" and its acronym, "UX". In the end, we settled for the following query:

> ("(emotional **OR** cognitive **OR** affective **OR** processing **OR** human) model") **AND** ("(testing **OR** test **OR** measure **OR** measuring **OR** estimate **OR** evaluate **OR** evaluating) user experience") **AND** (automated **OR** automatic)

Running this query through Google Scholar resulted in 133 hits, whereas running the same query through IEEExplore or the ACM Digital library lead to no results.

Having defined the research string, we needed inclusion/exclusion criteria in order to filter the publications found. We decided to only include works that automatically evaluated UX, employed an internal model of users and had no reliance on user testing (Tab. 1). Scanning the 133 papers found, we didn't find any publication that matched our criteria. This wasn't too surprising: throughout our preliminary searches we were unable to find any work that fully satisfied our research question. To our knowledge, there have been no documented attempts to test UX using implementations of affective/cognitive models.

Table 1. Criteria used for inclusion of publications for the systematic review.

Inclusion Criteria
-Automatically evaluates user experience
-Has no reliance on user testing (except for either training or validation)
-Employs a model of users

4.2 Scoping Review

Having been unable to find any publication that fully satisfied our inclusion criteria, we decided to find areas of research that could be a valuable inspiration for future research and development of automatic UX testing. We had found several works that didn't fully satisfy our criteria but had certain characteristics that made them useful resources for possible future developments in the area. Some works attempted to automatically assess a single component of user experience without modeling users and others created models of users but with different objectives in mind. We considered that having a catalog of all such works we could find to be a useful resource for future research.

To achieve this goal, we began a new review process. Our new research question was "What works and related areas of research could be a valuable inspiration for future research and development of automatic UX testing?" Given the horizontal nature of this new research question, we decided to conduct a broad scoping review instead of a systematic one, hoping in this way to find areas of research that a focused systematic review could miss.

5 RELATED RESEARCH AREAS

As previously described (Sec. 4.1), our systematic review, following strict inclusion criteria (Tab. 1), led us to conclude that there have been no documented attempts to automatically test UX using implementations of affective/cognitive models. The works presented in this section, divided in 6 different research areas, were found by our broad scoping review. We thus strive to combine in a single place all the works that we believe can provide meaningful insights and methodology for the development of automatic UX testing.

5.1 Automated Usability Testing

One example is Cogtool [4], a UI prototyping tool that produce quantitative predictions of skilled users' execution time. CogTool allows UI designers and developers to deploy keystroke-level models in substantially less time by using a cognitive model which predicts how long it will take a skilled user to complete the tasks. In addition, UI designers and developers can interpret their CogTool models to extract design recommendations directly supported by the psychological science underlying the models. It is one of the most widely used tools for automatic evaluation of usability.

The accuracy of CogTool is disputed, with some authors claiming CogTool is accurate enough to be a valuable asset for developers [53], whereas others defend CogTool is not reliable enough to be of use [30]. Whatever may be the case, CogTool makes a case for the promising advantages of automatic usability testing and shows how such tools can be highly beneficial for agile development [61].

Other GOMSL based models have been used to evaluate various aspects of usability of new interface design of high-throughput screening application [31] and estimating the usability of interactive web forms has been done using KLM models [33].

Another approach has been not to automate the usability testing itself, but its evaluation [2]. By recording user actions on an XML file, Fiora et al. were able to compare them with the expected actions, estimating a number of usability metrics. The Tracking Real-Time User Experience (TRUE) system also focuses on recording a broad amount of information during user testing in order to better evaluate the overall user experience and accurately detect problematic events or locations in video games, automating the process of relating actions and events to the experiences of users [35].

An overview of the literature discussed in this section can be found in Table 2.

Table 2. Overview of reviewed UX papers in automated usability testing

Ref	Approach	Method
[4, 30, 33, 53, 61]	Human processor modeling	KLM
[31]	Human processor modeling	GOMSL
[2]	User behavior recording	Heuristics
[35]	Real-time user behavior recording	Indexing videos to events

5.2 Automated GUI Testing

Outside the realm of human processing models, testing the usability of software is often interconnected with testing the Graphical User Interface (GUI), ensuring everything is working as supposed. If a button has unexpected behaviour or a user is unable to exit a certain state, usability will be negatively impacted. TESTAR is a framework that automatically tests the GUI by mapping every possible action on each state using a widget tree model that is automatically derived from the GUI through the accessibility API, traversing the interface by continuously choosing actions [67]. The choice of actions can be done randomly or using more sophisticated choice methods [17, 18].

Tackling the same issue of GUI testing, Memon et al. [48] define a formal model of the interface, expressed as sets of objects, object properties and actions, in order to generate all possible states of the system. Using test cases, they proceed by automatically interacting with the GUI, verifying if the reached state is the same as the one expected by the formal model. If the GUI obtained state is different, a problem is detected.

Other approaches have focused on the aesthetics of interfaces. Miniukovich and Angeli have taken a metric approach to evaluate the aesthetics of websites and mobile apps [49]. Their approach was based on metrics like visual clutter and color variability. Their results seem to indicate the approach is better suited for websites.

An overview of the literature discussed in this section can be found in Table 3.

Table 3. Overview of reviewed UX papers in automated GUI testing

Ref	Approach	Method
[67]	Automatic test case generation	Widget tree (Random action selection)
[17]	Automatic test case generation	Widget tree (Q-learning-based action selection)
[18]	Automatic test case generation	Widget tree (GP-based action selection)
[48]	Automatic test case generation	Formal model of interface
[49]	Aesthetics evaluation	Metrics of aesthetics

5.3 Artificial Emotions

We did not find any publication that explored the use of emotional models with the intent of testing UX. However, there is a wealth of research in the use of computational emotional models for different purposes.

A number of reviews have been done in the area. Rumbell et al. conducted a review of emotional mechanisms in twelve different emotional agents and present a series of recommendations for designing emotion mechanisms within artificial agents [58]. Kowalczuk and Czubenko analyse twelve computational approaches to modeling artificial emotion, discussing how they reflect the existent psychological theories of emotion [37]. Focusing on reinforcement learning, Moerland et al. did a survey on the use of emotional models on this machine learning architecture [50]. A state of the art review regarding affective games, which are games where an assessment of the player's emotional state influences the gameplay, was also done [40].

Following a significantly different approach, Wang et al. describe how features and shapes of images can influence emotions aroused in human beings, describing a model that can interpolate emotional arousal and valence from images [45, 68]. Such an approach could be a very strong complement to emotional models based on psychological theories of emotion.

An overview of the literature discussed in this section can be found in Table 4.

5.4 User Modelling

User Modelling is a sub area of Human Computer Interaction that focuses on modelling users, mostly with the intent of adapting the response of systems to particular users. This is done based on user models, that is, "models that systems

Table 4. Overview of reviewed papers in modeling of artificial emotion

Ref	Approach	Method
[58]	Computational modeling of emotion	Different emotional mechanisms
[37]	Computational modeling of emotion	Artificial emotion modeling
[50]	Model emotion in RL	Several (Review)
[40]	Affective games	Several (Review)
[68]	Recognize emotion from images	Shape modelling

have of users that reside inside a computational environment" [19]. Such modelling approaches resonate with the idea of cognitive and affective modelling, but most works focus on system specific criteria and do not attempt to use a more general modelling of the user's internal state. Most of the research in the area is focused on learning systems, that is, systems that aim to teach a certain task or set of skills to the user [14]. User modelling is relevant in such systems as it allows the system to adapt to the learning ability and prior knowledge of the user, providing a tailored experience that stimulates optimal learning. Whereas User Modelling is used in ways that improve UX, we were unable to find any work that used User Modelling in order to estimate UX.

Martinho et al. propose a user moddeling framework based on the OCC cognitive theory of emotions (Sec. 3.2.1), an approach that is highly aligned with our research question [47]. The framework allows to make predictions about the affective state of the user based on system events. The end goal of this work was not UX testing, but the framework could be modified with that goal in mind.

Also based on the OCC cognitive theory of emotions, Conati and Maclaren "present a probabilistic model of user affect designed to allow an intelligent agent to recognise multiple user emotions during the interaction with an educational computer game" [11]. Their approach, like many others in the User Modelling community, is system dependant, hindering the adaptation of the framework to different systems, something a UX testing framework would require. This is one of the open problems of User Modelling: how to create a user modelling framework that can be easily adapted to different systems. If this problem is solved, then the solution would be a very promising approach to automatic UX testing.

Taking a different approach, the Lumière Project uses probabilistic methods to model the intentions of users [28]. The authors create Bayesian models that allow the system under use to extrapolate the user's intentions from observed actions. With UX testing agents in mind, it could be interesting to invert this approach, creating models that extrapolate the most probable user actions given a goal.

Extended reality systems can present new challenges to UX and expand the real of that which can be modelled. Cheema et al. have proposed a Deep Reinforcement Learning model to predict the fatigue of user when interacting with a system using arm movement [10]. Such fatigue could prove to be fundamental for predicting long term use UX. Similar approaches could be required to complement cognitive or emotional models to ensure accurate UX automated testing of extended reality systems.

An overview of the literature discussed in this section can be found in Table 5.

5.5 Automatic Playtesting

Changing the focus from what users feel to what users do, the research area of Automatic Playtesting focuses on creating agents that play games in a similar manner to a human user. Such agents can be valuable assets both for

Table 5. Overview of reviewed papers in user modeling

Ref	Approach	Method
[14]	Learner modeling	Several (Review)
[47]	Emotion recognition	OCC model of emotion
[11]	Emotion recognition	OCC model of emotion
[28]	Goal extrapolation	Bayesian models
[10]	Fatigue modelling	Arm mechanics model

creating and testing games. For UX testing, having agents that act as humans is also valuable, as appraising UX on a sequence of actions that no user would ever do can fail to provide meaningful information.

Trying to mimic human cognitive processes, Stahlke et al. propose a framework to create agents that suffer from the same limitations as humans, like a field of view and limited short-term memory [62]. The goal choice of such agents is based on heuristics that try to mimic different human behaviours, like hazard avoidance, exploration, aggression, etc... These traits are called play-styles which lead to different goal prioritization in the heuristics.

Using a contrasting black-box approach, Gudmundsson et al. use deep learning to train a model to choose the most human-like action on the "Candy Crush Saga" game [23]. Despite being done on a relatively simple game, the idea of training a model to choose the most "human-like" action instead of the best action could prove useful for designing UX testing agents.

One could argue that not all people choose in the same way, and for that very reason Holmgård et al. defined different personas for testing maps for a 2D game [27]. Based on psychological decision theory, the authors implement different personas using a variation of Monte Carlo Tree Search where each persona has an altered node selection criteria developed using evolutionary computation. This approach is later used in playtesting of Match-3 games to represent various play-styles [52].

Automatic playtesting has also been used in order to test board games [13], where four different types of intelligent agents were used to play a board game, finding "bugs" in the game rules and providing a wealth of information that could previously only be obtained after many playtesting sessions with users. Card games were also tested using automatic playtesting [20]. The authors used evolutionary algorithms to create new decks of cards, testing if some novel strategy or deck could be prejudicial to the balance of the game. Moreover, [24] proposed to use a team of artificial general intelligent agents with different goals and skill levels to assess games, using expected performance of each agent as well as how the experience of the agents evolves in the game.

To assist designers with automated playtesting, several agent-based frameworks have recently been proposed. Zhao et. al. [6] trained playtesting AI agents to receive feedback in the game development process as well as game-playing AI agents in order to have an interaction with human players to shape gameplay experience. They presented four case studies, two on creating playtesting agents and two on creating game-playing agents. The game-playing agents learn behavior policies from the game designers instead of player behaviors. The training algorithms of these agents are chosen based on the style and skill requirements in each case studies.

With ICARUS, Pfau et al. propose a framework for autonomous video game playing, testing and bug reporting [56]. Their approach, working for all games created using a specific game engine, uses reinforcement learning combined with both volatile short-term memory and persistent long-term memory to learn how to traverse the game. The framework is able to play games in real-time, allowing the evaluation of game performance metrics like frames per second and CPU usage.

An overview of the literature discussed in this section can be found in Table 6.

Table 6. Overview of reviewed papers in automated playtesting

Ref	Approach	Method
[62]	AI testing agent	Mimicing players' styles by heuristics
[23]	AI testing agent	CNN
[27, 52]	AI testing agent with Persona	MCTS and evolutionary computation
[13]	AI testing agent	Heuristic
[20]	AI testing agent	Evolutionary algorithms
[24]	AI testing agent	General AI algorithms
[6]	AI testing agent	DQN, Rainbow, DNN, Deep RL
[56]	AI testing agent	RL and memory

CNN: Convolutional neural network, MCTS: Monte-Carlo tree search, DQN: Deep Q-network, DNN: Deep network, RL: Reinforcement learning

5.6 Physiological and Behavioural Measurement of UX

In this research area, we change the focus from how to develop automated UX testing to how to test and validate the developed solutions. We believe a truly automatic UX testing approach should not rely on real users when testing a system. However, user testing could be an integral part of both development and validation of UX testing solutions. It has been long noted that there is a correlation between physiological responses and emotional states. William James went so far as to propose that emotions were no more than our perception of those physiological responses [29]. For James, fear is our mind's perception of an increased adrenaline level, accelerated heartbeat, etc.

Based on the physiological and emotional correlation, research has been conducted in order to be able to predict certain characteristics of UX based on a number of different physiological and behavioural responses. Such approaches can provide information about UX over time and help evaluate the accuracy of developed automatic solutions. We found two review papers which are valuable resources for understanding the state of the art of the area [7, 51].

For example, zygomaticus muscle facial electromyography has been used as a predictor of emotional valence of boys playing video games [26] and Drachen et al. explored the correlation between heart rate, electrodermal activity and the self reported experience of users of first-person shooter games, finding a significant correlation [16].

Avoiding specialized hardware, Shaker et al. combined facial expressions, head movement and behavioural analysis to create a machine leaning model able to predict player experience when interacting with a single player game [60]. They further built upon the model in order to be able to generate personalized game maps with the intent of inducing certain levels of engagement, frustration, and challenge. Also relying on video feed, Tan et al. propose an approach to automate playtesting, using facial expression recognition software to attribute a degree of fun to the playing experience [65].

An overview of the literature discussed in this section can be found in Table 7.

Table 7. Overview of reviewed papers in UX evaluation using physiological and behavioral response

Ref	Approach	Method
[29]	Emotion recognition by physiological response	ECG, GSR, EMG
[7, 51]	Emotion recognition by physiological response	EEG, fMRI, HR, etc
[26]	Physiological measurement of UX	Facial electromyography
[16]	Physiological response for UX evaluation	HR, EDA, reports
[60]	Model UX for prediction	Facial expressions, behavioral anaylsis with ML
[65]	Emotion recognition by physiological response	Facial expression recognition

6 CONCLUSION AND DISCUSSION

We began this study with a systematic review, searching for works that implemented cognitive/affective models with the intent of automatically estimating UX. Doing so, we were unable to find any works that satisfied our inclusion criteria. We argue that this represents a research gap, which should be tackled in light of the recent technological advances in AI and automated testing. Even if not completely, UX testing needs to be automated or it will remain a bottleneck on development.

We thus continued our study by searching for research works that could provide meaningful insights and techniques to serve as the foundation to the development of automated UX testing. We presented the works we believed to be more representative and significant for the goal of automated UX testing, dividing them in 6 different research areas.

In making this paper, we strive to motivate and support research towards the automation of UX testing. We do this by: presenting a lack of research on UX testing based on agents and cognitive/affective models (Sec. 4); and by organizing and presenting literature that can be used as the foundation or inspiration of such research (Sec. 5).

We believe that the foundations, methods and technology required to make automatic UX testing possible can be found in literature. However, it is necessary to join the different venues of knowledge with the intent of testing UX. This can be done in several different ways, focusing on different components of UX.

Inferring highly complex human emotional states can be extremely challenging. Such states can depend on a myriad of external and internal details, many of which are not yet understood. Creating agents that can accurately simulate the internal experience of seeing a beautiful scenery is not, at the moment, feasible. For this reason, we believe a part of UX testing will continue to rely on user testing in the foreseeable future. This doesn't mean, however, that we shouldn't attempt to automate parts of the process. Some mental states might be easier to model and predict than others. If a user is forced to repeat the same simple task a thousand times in a row, it isn't absurd to suppose the user will become bored. If when playing a game, a user becomes trapped on a certain location, it is predictable that she will become frustrated. If accomplishing a task requires keeping track of a dozens objects at the same time, it is likely the user will become confused.

We believe UX testing automation should begin with easier to predict affective states. Being able to automatically detect the aforementioned situations would already be extremely useful for both developers and designers. It wouldn't replace user testing, but it would reduce its burden and allow for UX testing to be constantly present during iterative runs of development.

We consider agent based approaches to be the most promising. As presented in Sec. 5.5, there have already been successful implementations of agents for playtesting. Endowing those agents with internal cognitive/affective models could allow them not only to estimate certain UX measures but also allow them to behave in a more "human-like" way.

These agents could be used to test not only UX but also functional aspects of the environment, allowing developers to have a great number of such agents testing the environments in parallel, providing countless hours of gameplay testing.

Coverage is always a concern when testing a system or environment. Full coverage in testing would correspond to visiting every single possible state of the system in search for problems. When such an approach is feasible, game-playing agents that search the entirety of the search space can be a solution. However, there are several situations where that is not feasible given the large number of possible states. In those situations, agents that behave as closely to humans as possible become a valuable asset. When covering all possible states is impossible, covering the greatest number of states that are most likely to be reached by humans can be the next best option.

It might be argued that to have such testing agents, we'd have to develop Artificial General Intelligence (AGI) [21], defined in contrast to "narrow AI" [39] as AI that is able to generalize knowledge learnt in solving a task to help solving different tasks, equalling or surpassing humans. Having AGI would make the task much easier and the agents more complete, but as previously said, we do not argue that agents can, for now, completely replace human users in UX testing. What we defend is that they could automatically test some essential components of UX that do not require human expertise, reducing the need for human labour. Using as example an industry that is already highly automated, car factories implement robots and AI to assess the quality of the developed vehicles, but no car reaches the road without being inspected by a human. This does not mean the automated quality control solutions were not worth the effort for not being able to fully do the job of quality control. They allow the human operators to focus on those things that require a human mind to accomplish. They also allow the car manufacturers to do more complete quality control inspections that perhaps wouldn't be economically viable if they had to pay human operators to do them. Furthermore, what automatic solutions can test, they can do so more exhaustively and reliably than humans.

REFERENCES

[1] Rui Alves, Pedro Valente, and Nuno Jardim Nunes. 2014. The state of user experience evaluation practice. In *Proceedings of the 8th Nordic Conference on Human-Computer Interaction: Fun, Fast, Foundational.* 93–102.

[2] Fiora TW Au, Simon Baker, Ian Warren, and Gillian Dobbie. 2008. Automated usability testing framework. In *Proceedings of the ninth conference on Australasian user interface-Volume 76.* 55–64.

[3] Javier A Bargas-Avila and Kasper Hornbæk. 2011. Old wine in new bottles or novel challenges: a critical analysis of empirical studies of user experience. In *Proceedings of the SIGCHI conference on human factors in computing systems.* 2689–2698.

[4] Rachel Bellamy, Bonnie John, and Sandra Kogan. 2011. Deploying CogTool: integrating quantitative usability assessment into real-world software development. In *Proceedings of the 33rd international conference on software engineering.* 691–700.

[5] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. 2016. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316* (2016).

[6] Igor Borovikov, Yunqi Zhao, Ahmad Beirami, Jesse Harder, John Kolen, James Pestrak, Jervis Pinto, Reza Pourabolghasem, Harold Chaput, Mohsen Sardari, et al. 2019. Winning isn't everything: Training agents to playtest modern games. In *AAAI Workshop on Reinforcement Learning in Games.*

[7] Mauro Callejas-Cuervo, Laura Alejandra Martínez-Tejada, and Andrea Catherine Alarcón-Aldana. 2017. Emotion recognition techniques using physiological signals and video games-Systematic review. *Revista Facultad de Ingeniería* 26, 46 (2017), 19–28.

[8] Stuart K Card, Thomas P Moran, and Allen Newell. 1983. The psychology of human-computer interaction. 1983.

[9] Emanuelle Cavalcante, Luis Rivero, and Tayana Conte. 2015. Evaluating the feasibility of max: a method using cards and a board for assessing the post-use UX. *International Journal of Software Engineering and Knowledge Engineering* 25, 09n10 (2015), 1759–1764.

[10] Noshaba Cheema, Laura A Frey-Law, Kourosh Naderi, Jaakko Lehtinen, Philipp Slusallek, and Perttu Hämäläinen. 2020. Predicting Mid-Air Interaction Movements and Fatigue Using Deep Reinforcement Learning. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems.* 1–13.

[11] Cristina Conati and Heather Maclaren. 2009. Empirically building and evaluating a probabilistic model of user affect. *User Modeling and User-Adapted Interaction* 19, 3 (2009), 267–303.

[12] Roddy Cowie, Ellen Douglas-Cowie, Susie Savvidou*, Edelle McMahon, Martin Sawey, and Marc Schröder. 2000. 'FEELTRACE': An instrument for recording perceived emotion in real time. In *ISCA tutorial and research workshop (ITRW) on speech and emotion.*

[13] Fernando de Mesentier Silva, Scott Lee, Julian Togelius, and Andy Nealen. 2017. AI-based playtesting of contemporary board games. In *Proceedings of the 12th International Conference on the Foundations of Digital Games.* 1–10.

[14] Michel C Desmarais and Ryan SJ d Baker. 2012. A review of recent advances in learner and skill modeling in intelligent learning environments. *User Modeling and User-Adapted Interaction* 22, 1-2 (2012), 9–38.

[15] ISO DIS. 2010. 9241-210: 2010. Ergonomics of human system interaction-Part 210: Human-centred design for interactive systems (formerly known as 13407). *International Standardization Organization (ISO). Switzerland* (2010).

[16] Anders Drachen, Lennart E Nacke, Georgios Yannakakis, and Anja Lee Pedersen. 2010. Correlation between heart rate, electrodermal activity and player experience in first-person shooter games. In *Proceedings of the 5th ACM SIGGRAPH Symposium on Video Games*. 49–54.

[17] Anna I Esparcia-Alcázar, Francisco Almenar, Mirella Martínez, Urko Rueda, and T Vos. 2016. Q-learning strategies for action selection in the TESTAR automated testing tool. *6th International Conferenrence on Metaheuristics and nature inspired computing (META 2016)* (2016), 130–137.

[18] Anna I Esparcia-Alcázar, Francisco Almenar, Tanja EJ Vos, and Urko Rueda. 2018. Using genetic programming to evolve action selection rules in traversal-based automated software testing: results obtained with the TESTAR tool. *Memetic Computing* 10, 3 (2018), 257–265.

[19] Gerhard Fischer. 2001. User modeling in human–computer interaction. *User modeling and user-adapted interaction* 11, 1-2 (2001), 65–86.

[20] Pablo García-Sánchez, Alberto Tonda, Antonio M Mora, Giovanni Squillero, and Juan Julián Merelo. 2018. Automated playtesting in collectible card games using evolutionary algorithms: A case study in hearthstone. *Knowledge-Based Systems* 153 (2018), 133–146.

[21] Ben Goertzel. 2014. Artificial general intelligence: concept, state of the art, and future prospects. *Journal of Artificial General Intelligence* 5, 1 (2014), 1–48.

[22] Jonathan Gratch and Stacy Marsella. 2005. Evaluating a computational model of emotion. *Autonomous Agents and Multi-Agent Systems* 11, 1 (2005), 23–43.

[23] Stefan Freyr Gudmundsson, Philipp Eisen, Erik Poromaa, Alex Nodet, Sami Purmonen, Bartlomiej Kozakowski, Richard Meurling, and Lele Cao. 2018. Human-like playtesting with deep learning. In *2018 IEEE Conference on Computational Intelligence and Games (CIG)*. IEEE, 1–8.

[24] Cristina Guerrero-Romero, Simon M Lucas, and Diego Perez-Liebana. 2018. Using a team of general ai algorithms to assist game design and testing. In *2018 IEEE Conference on Computational Intelligence and Games (CIG)*. IEEE, 1–8.

[25] Marc Hassenzahl and Noam Tractinsky. 2006. User experience-a research agenda. *Behaviour & information technology* 25, 2 (2006), 91–97.

[26] Richard L Hazlett. 2006. Measuring emotional valence during interactive experiences: boys at video game play. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*. 1023–1026.

[27] Christoffer Holmgard, Michael Cerny Green, Antonios Liapis, and Julian Togelius. 2018. Automated playtesting with procedural personas with evolved heuristics. *IEEE Transactions on Games* (2018).

[28] Eric J Horvitz, John S Breese, David Heckerman, David Hovel, and Koos Rommelse. 2013. The Lumiere project: Bayesian user modeling for inferring the goals and needs of software users. *arXiv preprint arXiv:1301.7385* (2013).

[29] William James. 2007. *The principles of psychology*. Vol. 1. Cosimo, Inc.

[30] Wiard Jorritsma, Peter-Jan Haga, Fokie Cnossen, Rudi A Dierckx, Matthijs Oudkerk, and Peter MA van Ooijen. 2015. Predicting human performance differences on multiple interface alternatives: KLM, GOMS and CogTool are unreliable. *Procedia Manufacturing* 3 (2015), 3725–3731.

[31] David B Kaber, Rebecca S Green, Sang-Hwan Kim, and Noa Segall. 2011. Assessing usability of human–machine interfaces for life science automation using computational cognitive models. *Intl. Journal of Human–Computer Interaction* 27, 6 (2011), 481–504.

[32] Evangelos Karapanos, Jean-Bernard Martens, and Marc Hassenzahl. 2010. On the retrospective assessment of users' experiences over time: memory or actuality? In *CHI'10 Extended Abstracts on Human Factors in Computing Systems*. 4075–4080.

[33] Christos Katsanos, Nikos Karousos, Nikolaos Tselios, Michalis Xenos, and Nikolaos Avouris. 2013. KLM Form analyzer: automated evaluation of web form filling tasks using human performance models. In *IFIP Conference on Human-Computer Interaction*. Springer, 530–537.

[34] David E Kieras. 1999. A guide to GOMS model usability evaluation using GOMSL and GLEAN3. *University of Michigan* 313 (1999).

[35] Jun H Kim, Daniel V Gunn, Eric Schuh, Bruce Phillips, Randy J Pagulayan, and Dennis Wixon. 2008. Tracking real-time user experience (TRUE) a comprehensive instrumentation solution for complex systems. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. 443–452.

[36] Barbara Kitchenham, Stuart Charters, et al. 2007. Guidelines for performing systematic literature reviews in software engineering version 2.3. *Engineering* 45, 4ve (2007), 1051.

[37] Zdzisław Kowalczuk and Michał Czubenko. 2016. Computational approaches to modeling artificial emotion–an overview of the proposed solutions. *Frontiers in Robotics and AI* 3 (2016), 21.

[38] Sari Kujala, Virpi Roto, Kaisa Väänänen-Vainio-Mattila, Evangelos Karapanos, and Arto Sinnelä. 2011. UX Curve: A method for evaluating long-term user experience. *Interacting with computers* 23, 5 (2011), 473–483.

[39] Ray Kurzweil. 2005. *The singularity is near: When humans transcend biology*. Penguin.

[40] Raúl Lara-Cabrera and David Camacho. 2019. A taxonomy and state of the art revision on affective games. *Future Generation Computer Systems* 92 (2019), 516–525.

[41] Effie Lai-Chong Law. 2011. The measurability and predictability of user experience. In *Proceedings of the 3rd ACM SIGCHI symposium on Engineering interactive computing systems*. 1–10.

[42] Effie Lai-Chong Law, Virpi Roto, Marc Hassenzahl, Arnold POS Vermeeren, and Joke Kort. 2009. Understanding, scoping and defining user experience: a survey approach. In *Proceedings of the SIGCHI conference on human factors in computing systems*. 719–728.

[43] Effie Lai-Chong Law, Paul Van Schaik, and Virpi Roto. 2014. Attitudes towards user experience (UX) measurement. *International Journal of Human-Computer Studies* 72, 6 (2014), 526–541.

[44] Richard S Lazarus and Richard S Lazarus. 1991. *Emotion and adaptation.* Oxford University Press on Demand.

[45] Xin Lu, Poonam Suryanarayan, Reginald B Adams Jr, Jia Li, Michelle G Newman, and James Z Wang. 2012. On shape and the computability of emotions. In *Proceedings of the 20th ACM international conference on Multimedia.* 229–238.

[46] Sascha Mahlke and Manfred Thüring. 2007. Studying antecedents of emotional experiences in interactive contexts. In *Proceedings of the SIGCHI conference on Human factors in computing systems.* 915–918.

[47] Carlos Martinho, Isabel Machado, and Ana Paiva. 1999. A cognitive approach to affective user modeling. In *International Workshop on Affective Interactions.* Springer, 64–75.

[48] Atif M Memon, Martha E Pollack, and Mary Lou Soffa. 2000. Automated test oracles for GUIs. *ACM SIGSOFT Software Engineering Notes* 25, 6 (2000), 30–39.

[49] Aliaksei Miniukovich and Antonella De Angeli. 2015. Computation of interface aesthetics. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems.* 1163–1172.

[50] Thomas M Moerland, Joost Broekens, and Catholijn M Jonker. 2018. Emotion in reinforcement learning agents and robots: a survey. *Machine Learning* 107, 2 (2018), 443–480.

[51] Seong-Eun Moon and Jong-Seok Lee. 2016. Implicit analysis of perceptual multimedia experience based on physiological response: a review. *IEEE Transactions on Multimedia* 19, 2 (2016), 340–353.

[52] Luvneesh Mugrai, Fernando Silva, Christoffer Holmgård, and Julian Togelius. 2019. Automated playtesting of matching tile games. In *2019 IEEE Conference on Games (CoG).* IEEE, 1–7.

[53] Nihan Ocak and Kursat Cagiltay. 2017. Comparison of cognitive modeling and user performance analysis for touch screen mobile interface design. *International Journal of Human–Computer Interaction* 33, 8 (2017), 633–641.

[54] Andrew Ortony, Gerald L Clore, and Allan Collins. 1990. *The cognitive structure of emotions.* Cambridge university press.

[55] Jessica Peterson, Patricia F Pearce, Laurie Anne Ferguson, and Cynthia A Langford. 2017. Understanding scoping reviews: Definition, purpose, and process. *Journal of the American Association of Nurse Practitioners* 29, 1 (2017), 12–16.

[56] Johannes Pfau, Jan David Smeddinck, and Rainer Malaka. 2017. Automated game testing with icarus: Intelligent completion of adventure riddles via unsupervised solving. In *Extended Abstracts Publication of the Annual Symposium on Computer-Human Interaction in Play.* 153–164.

[57] Luis Rivero and Tayana Conte. 2017. A systematic mapping study on research contributions on UX evaluation technologies. In *Proceedings of the XVI Brazilian Symposium on Human Factors in Computing Systems.* 1–10.

[58] Timothy Rumbell, John Barnden, Susan Denham, and Thomas Wennekers. 2012. Emotions in autonomous agents: comparative analysis of mechanisms and functions. *Autonomous Agents and Multi-Agent Systems* 25, 1 (2012), 1–45.

[59] Pertti Saariluoma and Jussi PP Jokinen. 2014. Emotional dimensions of user experience: A user psychological analysis. *International Journal of Human-Computer Interaction* 30, 4 (2014), 303–320.

[60] Noor Shaker, Stylianos Asteriadis, Georgios N Yannakakis, and Kostas Karpouzis. 2013. Fusing visual and behavioral cues for modeling user experience in games. *IEEE transactions on cybernetics* 43, 6 (2013), 1519–1531.

[61] Anil Shankar, Honray Lin, Hans-Frederick Brown, and Colson Rice. 2015. Rapid Usability Assessment of an Enterprise Application in an Agile Environment with CogTool. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems.* 719–726.

[62] Samantha Stahlke, Atiya Nova, and Pejman Mirza-Babaei. 2019. Artificial Playfulness: A Tool for Automated Agent-Based Playtesting. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems.* 1–6.

[63] Ron Sun. 2008. Introduction to computational cognitive modeling. *Cambridge handbook of computational psychology* (2008), 3–19.

[64] Marika Tähti and L Arhippainen. 2004. A Proposal of collecting Emotions and Experiences. *Interactive Experiences in HCI* 2 (2004), 195–198.

[65] Chek Tien Tan and Andrew Johnston. 2011. Towards a Non-Disruptive, Practical and Objective Automated Playtesting Process. In *Workshops at the Seventh Artificial Intelligence and Interactive Digital Entertainment Conference.*

[66] Arnold POS Vermeeren, Effie Lai-Chong Law, Virpi Roto, Marianna Obrist, Jettie Hoonhout, and Kaisa Väänänen-Vainio-Mattila. 2010. User experience evaluation methods: current state and development needs. In *Proceedings of the 6th Nordic conference on human-computer interaction: Extending boundaries.* 521–530.

[67] Tanja EJ Vos, Peter M Kruse, Nelly Condori-Fernández, Sebastian Bauersfeld, and Joachim Wegener. 2015. Testar: Tool support for test automation at the user interface level. *International Journal of Information System Modeling and Design (IJISMD)* 6, 3 (2015), 46–83.

[68] James Z Wang, Xin Lu, Poonam Suryanarayan, Reginald B Adams, Jia Li, Michelle Newman, et al. 2018. Automatically computing emotions aroused from images through shape modeling. US Patent 9,904,869.

ANNEX A2

Agents for Automatic User Experience Testing

Pedro M. Fernandes[1*] , **Manuel Lopes**[1] , **Rui Prada**[1]
[1]INESC-ID and Instituto Superior Técnico, Univ. de Lisboa
{pedro.miguel.rocha.fernandes, manuel.lopes, rui.prada}@tecnico.ulisboa.com

Abstract

The automation of functional testing in software has allowed developers to continuously check for negative impacts on functionality throughout the iterative phases of development. On the other hand, User eXperience (UX) has hitherto relied almost exclusively on testing with real users. User testing is a slow endeavour that becomes a bottleneck for development, which in turn makes UX testing a bottleneck as well. To address this problem, we here propose an agent based approach for automatic UX testing. We develop agents with basic problem solving skills and a core affect model, allowing us to assess their artificial affective state as they traverse different levels of a game. Although this research is still at a primordial state, we believe the results here presented make a strong case for the use of agents with the intent of automatic UX testing.

1 Introduction

Hitherto, User eXperience (UX) testing has relied on real users interacting with the system under test [Vermeeren *et al.*, 2010; Rivero and Conte, 2017]. To gather insights regarding the experience of the users, multiple approaches have been used, from questionnaires [Schrepp *et al.*, 2014] to physiological measurements [Callejas-Cuervo *et al.*, 2017; Moon and Lee, 2016]. When applied correctly, such approaches allow us to gain a wealth of knowledge about the caveats of the system under test and take action in order to improve UX.

Testing with users, however, is a slow endeavour. It is not efficient to have a user testing session at each step of the iterative development process. Whereas today is common practice to run automated functionality tests each time a system suffers an alteration, the same cannot be currently done concerning UX.

In this paper, we will propose using agents endowed with an affective model to run automated UX tests. We don't defend such agents could, in the foreseeable future, completely replace testing with real users. User testing might even be used to improve the accuracy of the automated agents, as will later be discussed. What we propose is that agents could be used to maintain a focus on UX throughout the entire development process, running alongside automatic functional tests.

Our research objective is to understand if such testing agents could be a viable approach to partially automate UX testing. With this intent, we have developed an agent endowed with an internal core affect model [Russell, 2003]. This agent was then used to gain insights on the UX of different maps of a game, Lab Recruits (Sec. 3). The results here presented aim to be a proof of concept, showing some of the information one could obtain using such a UX testing agent. We believe these primordial results make a strong claim for the usefulness of such agents. We further propose a number of ways on which the agents could be improved to be accurate and more versatile (Sec. 7).

This paper is organised as follows. In Sec. 2 we give a brief overview of previous research in the area. In Sec. 3 the testing environment is presented. In Sec. 4 we describe the core architecture of the agent. The results are described in Sec. 5 and the conclusions on Sec.6. Finally, in Sec. 7 we discuss the results and propose a number of ways in which the presented UX testing agents could be improved.

2 Related Work

Agents have been previously endowed with emotional dimensions and agents have been used for testing games and assisting with level design. But to the knowledge of the authors, both these things have never been done simultaneously in order to test UX. We believe this represents a research gap.

With regards to emotional agents, two reviews have been done, providing a wealth of information on different methods and approaches for creating agents endowed with artificial emotions [Rumbell *et al.*, 2012; Kowalczuk and Czubenko, 2016]. Most of these emotional agents were developed in order to be able to have more realistic interactions with users. They were therefore created to improve UX, but not to test it.

Playtesting agents have been developed that strive to play games in a similar fashion to real users. Designed to have some of the same limitations that human players have, these agents can have, for example, a field of view and a limited short-term memory. Such agents can then be used to play games and find problems and exploits that would normally require testing with real users to find. Stahlke et al. proposed

[*]Contact Author

a framework to create this type of agents, following heuristics based on human behaviours, like exploration, hazard avoidance and aggression [Stahlke *et al.*, 2019].

Playtesting agents have also been used to aid design. Holmgård et al. [Holmgard *et al.*, 2018] developed agents with a range of different personas, that is, following different behaviours and with different objectives. These agents were then used to aid developers create maps for a 2D game. These agents allowed developers to predict how different players would traverse the map and make design decisions accordingly. The creation of playtesting agents has also been used to test board games [de Mesentier Silva *et al.*, 2017; Guerrero-Romero *et al.*, 2018], card games [García-Sánchez *et al.*, 2018] and frameworks for the development of playtesting agents have been proposed [Borovikov *et al.*, 2019; Pfau *et al.*, 2017].

Outside the realm of games and focusing on the problem of automated UX testing, there is research in automatic Graphical User Interface (GUI) testing [Vos *et al.*, 2015; Memon *et al.*, 2000] and automatic usability testing [Bellamy *et al.*, 2011; Kaber *et al.*, 2011; Katsanos *et al.*, 2013]. Both these areas are of relevance for UX, but we were unable to find any work in them that attempted to do any real-time estimates as an agent interacted with a system. In the area of User Modelling, real-time estimates of the internal state of users is often done [Desmarais and d Baker, 2012]. Some works have even focused on modelling the internal emotional states of users[Martinho *et al.*, 1999; Conati and Maclaren, 2009], but to out knowledge, none has done so with the intent of UX testing.

3 Test Case: Lab Recruits Game

The system under test for this paper will be a game called Lab Recruits[1]. This environment was chosen as it allows to effortlessly design different maps and provides an easy integration with Aplib [Prasetya and Dastani, 2020], a Belief–Desire–Intention (BDI) based agent framework. We can thus design Lab Recruits maps and then deploy on those maps agents created using the Aplib framework.

Lab Recruits is a simple 3D game where the player must interact with objects, for example buttons, in order to achieve a goal, which can be opening a specific door. The only actions the player can do is move or try to interact with objects. Objects with which the player can interact will henceforth be called **interactables**.

For the examples presented in this paper, the following objects were used:

- **Door with Button:** A door that the player can open by interacting with the corresponding button. This object pair can be abstracted as a NPC or interactable which, when interacted with, allows the player to progress towards the objective. This object pair is represented in game by a door and a button connected by a wire.
- **Simple Button:** A button which isn't connected to anything. Even though the player can interact with it, such an interaction is not necessary for completing the player's objective. This object can be abstracted as a NPC or other interactables that are present in games but which are not directly relevant to the completion of the player's objective. This object is represented in game by a sphere that can have different colours and which does not have a wire connecting it to anything.
- **Chair:** A chair which cannot be interacted with. Finding this chair is, in all our examples, the objective of the player. This object can be abstracted as any item that a player must find or a specific location/state that must be reached. This object is represented in game by a black office chair.

[1]https://github.com/iv4xr-project/labrecruits

Further ahead in the paper, 4 Lab Recruits maps will be introduced (Fig. 2, 4, 6 and 8). All of those maps follow the same basic premise: the player spawns in a maze, which she must traverse in order to find a chair. Finding the chair is the ultimate goal of the player and when the chair is found, the simulation ends.

4 Agent Model

In order to be used for the UX testing of our Lab Recruits maps, the agents had to fulfil two main requirements: (a) be able to traverse the maze; and (b) record information relevant for UX assessment as they did so. Our approach to solving (a) is described in Sec. 4.1 and our solution for (b) in Sec. 4.2.

As previously mentioned, the agents were developed using the Aplib framework, being therefore built upon a BDI architecture.

4.1 Search and Traverse Algorithm

As the AI capabilities of the agents are not the main focus of this paper, they will be only briefly described here in order to give the reader an understanding of the agent's behaviour. Our objective was to have an agent that would behave in a similar fashion to a real user.

The agent has a field of view and cannot perceive that which is behind a wall or further than a specific distance. Being endowed with a spatial memory, the agent creates an internal map recording all the locations and objects it has found. This internal map is that which the agent uses for navigation.

The moment the agent spawns in a map, it only knows the location of that which it can directly perceive and knows which object it is trying to find: a chair. It also has the prior knowledge of how to open a door which is connected to a button. As the focus of our simulations was to test UX, it was counter productive to have the agent learn something that the grand majority of real users would already know how to do.

The chair finding algorithm runs as follows:

1. The agent perceives the environment and adds to its internal map all the locations and objects it has found.
2. If the chair was found, the simulation ends.
3. If the chair wasn't found but a button that opens a door was found, the agent will move towards the button and interact with it in order to open the door. If more than one door opening button was found, the agent will randomly choose one to interact with.

4. If neither the chair nor a door opening button were found, the agent moves to the closest information limit of its internal map, striving to find more locations and objects. A location is considered an information limit if the agent doesn't have any information of what is beyond it. The agent considers walls hard limits and will not attempt to explore past them.

These steps are repeated in this order until the chair is found or all off the map is completely explored.

4.2 Affective Model

UX is a broad concept that entails a great number things. The ISO 9241-210:2019 defines UX as the "user's perceptions and responses that result from the use and/or anticipated use of a system, product or service.", further clarifying that "Users' perceptions and responses include the users' emotions, beliefs, preferences, perceptions, comfort, behaviours, and accomplishments that occur before, during and after use." [DIS, 2010].

We decided to begin our research by focusing on the emotional state of the user throughout the interaction with the system. To do so, we needed to endow the agent with an emotional model. Several theoretical models of emotion have been proposed [Russell and Mehrabian, 1977; Ortony *et al.*, 1990; Plutchik, 1980], a number of which have already been used to give agents artificial emotions [Rumbell *et al.*, 2012; Kowalczuk and Czubenko, 2016]. We decided to base our model on the Core Affect theory of emotions [Russell, 2003]. This theory defends that emotions are constructed from an initial affective state through processes like attribution and appraisal. This affective state can be defined using two dimensions, which will henceforth be referred to as Pleasure and Arousal. These dimensions can take positive or negative values.

An affective state, by itself, is not enough to define an emotional state. But according to the Core Affect theory, it is the starting point of emotions, and as such will be the starting point of our research. In Sec.7 we will discuss how this approach could be built upon in order to characterise complete emotional states.

Having the two dimensions that will define our agent's artificial affective state, we now need to define how the agent's interaction with the environment will alter those dimensions. At this stage of the research, our main focus is not yet accuracy, but understanding the feasibility of an agent based approach to automatically test UX. As such, on defining how the agent's interaction with the environment alters its artificial affective state, we decided to use simple rules. There is a lot of room for improvement over this rule based approach and some proposals will be made in Sec.7.

Both the affective dimensions will be in the range $\{-5, 5\}$. When the agent spawns in the environment, both dimensions are neutral, that is, having a value of 0. From that moment on, the affective dimensions are thus calculated:

- **Pleasure:**
  - Whenever the agent is able to accomplish his objective (finding the chair) or intermediate sub-objectives (opening doors), the Pleasure dimension **increases** by 1.
  - If the agent doesn't accomplish any objective or sub-objective for 10 seconds, the Pleasure dimension **decreases** by 0.4.
- **Arousal:**
  - Whenever the agent finds a new interactable (buttons and doors), the Arousal dimension **increases** by 1
  - If the agent doesn't find any new interactables for 10 seconds, the Arousal dimension **decreases** by 0.4.

The values here used are not yet an accurate representation of the affective response of a real user. To be so, they would need to be experimentally tested, which they have not yet been at this point in the research. As it stands, we wish to understand if such a simple affective model could already provide meaningful UX insights when used to test a number of different scenarios.

5 Results

On this section, four different Lab Recruits maps will be tested. For each, we will present the changes to the affective state of the aforedescribed agent (Sec. 4) as the map is traversed. Such changes will be shown both in function of the location of the agent and in function of time.

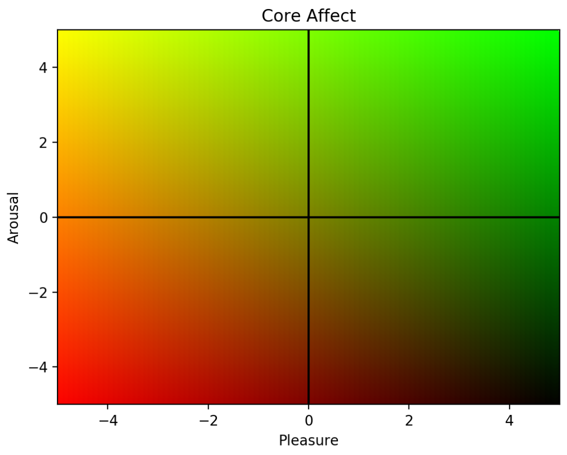


Figure 1: The colour gradation in the two dimensional core affect space, used to represent the artificial affective state of the agent as it traverses the maps.

5.1 Map 1

The first Lab Recruits map we will explore is the simplest one (Fig. 2). The agent must traverse a maze where it finds no interectables in order to reach a door. When the agent interacts with the button that opens that door, it finds the chair, which is its objective.

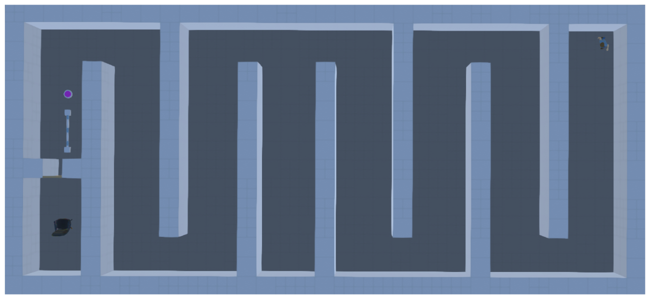


Figure 2: The layout of Map 1. In this map, the agent starts on the top right corner and must cross the maze, finding no interactables on its way, until it reaches a door and a button, which it has to press in order to find the chair.

The evolution of the agent's affective state in function of time and space can be found on Fig. 3b and Fig. 3a, respectively.

Both the Pleasure and Arousal dimensions remain negative throughout the agent's traversal of the map. Both Arousal and Pleasure steadily decrease (Fig. 3b) until the agent finds the door and the button, where both dimensions finally increase (Fig. 3a).

5.2 Map 2

In the second map, the agent must traverse the same maze as it did in Map 1. However, this time the agent will find interactables on its way to its final objective. These interactables, being buttons not connected to anything, are not relevant to the completion of the agent's objective.

The evolution of the agent's affective state in function of time and space can be found on Fig. 5b and Fig. 5a, respectively.

The Pleasure dimension of the agent's affective state steadily decreases as it traverses the maze (Fig. 5b), but this time its Arousal dimension increases each time the agent finds an interactable (Fig. 5a).

5.3 Map 3

In Map 3, the agent once again finds itself in a maze. This time, to reach the chair, the agent will not only have to open the final door but 3 other doors that are located throughout the maze. The agent is unable to reach its objective unless the 4 doors are opened. Besides the doors and their corresponding buttons, the agent will find 2 other interactables in the maze.

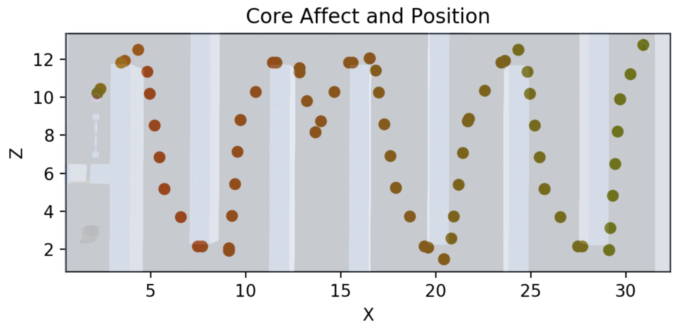
The evolution of the agent's affective state in function of time and space can be found on Fig. 7b and Fig. 7a, respectively.

Unlike the previous maps, both the Pleasure and Arousal affective dimensions remain positive throughout the agent's traversal of the map (Fig. 7b). The agent's Arousal increases whenever it sees an interactable and its Pleasure increases each time the agent successfully opens a door (Fig. 7a).

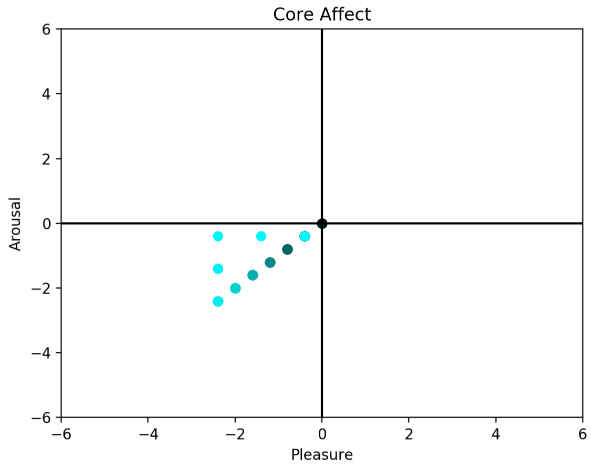
5.4 Map 4

The fourth map we will analyse is considerably different from the previous 3. This time, the agent is not in a maze but in a room with 28 doors. Each door has a button that opens it and behind one of the doors, is the chair. The agent can see from the start all the doors and buttons but cannot see the chair before opening the right door.

In this map, the agent's affective experience is very different depending on how "lucky" the agent is. In the best case scenario, the agent will open the correct door on the first attempt. On the worst case scenario, the agent will open 27 doors before finally opening the correct one. Because of this, we will here explore both the best and worst case scenarios.



(a)



(b)

Figure 3: The evolution of the agent's affective state in function of time (a) and space (b) as it traverses Map 1 (Fig. 2). The colours in (a) can be mapped to the 2-dimensional Core Affect space using Fig. 1. In (b), the dots become increasingly light blue as time passes. This means that the first affective measurement is a totally black dot, whereas the last measurement is a bright blue dot. We can see that both affective dimensions turn gradually more negative over time until the agent finally finds the door and opens it, which leads to an increase in both the affective dimensions. The agent's affective state remains in the negative-arousal and negative-pleasure quadrant throughout the traversal.

Best Case Scenario

In the best case scenario, the first button that the agent activates opens the correct door, leading it to the chair.

The evolution of the agent's affective state for this scenario in function of time and space can be found on Fig. 9b and

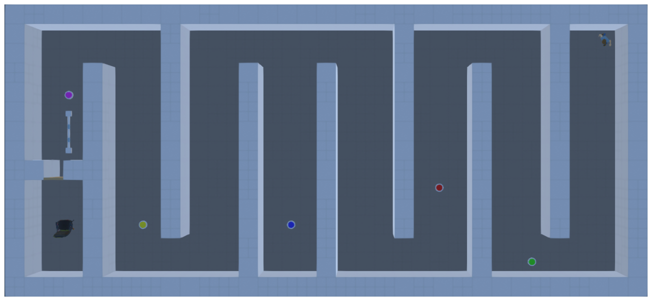


Figure 4: The layout of Map 2. In this map, the agent starts on the top right corner and must cross the maze, finding four interactables on its way, which are not directly relevant to its objective. At the end of the maze, the agent will find a door and a button, which it has to press in order to find the chair.

Fig. 9a, respectively.

In this scenario, the agent's traversal of this map is swift, as the agent must only move from its original position to that of the correct button. Its Arousal dimension doesn't have time to suffer any changes and the Pleasure dimension increases as the agent is able to open the door and find the chair.

Worst Case Scenario

In the worst case scenario, the agent first opens at random 27 doors before finally opening the correct one, behind which it finds the chair.

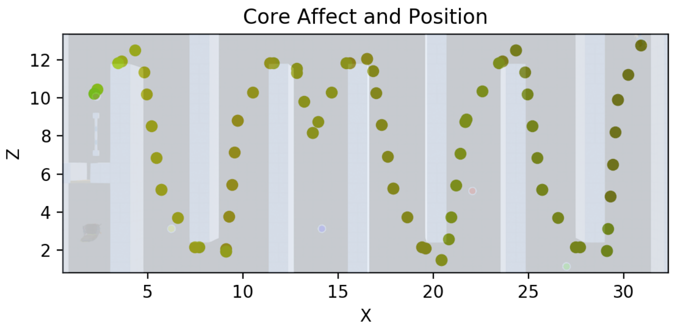
The evolution of the agent's affective state for this scenario in function of time and space can be found on Fig. 10b and Fig. 10a, respectively.

Whenever the agent is able to successfully open a door, the agent's Pleasure dimension increases (Fig.10a). However, since the agent could see all the buttons and doors since the beginning, there is nothing to increase the agent's Arousal, which steadily decreases with time (Fig 10b).

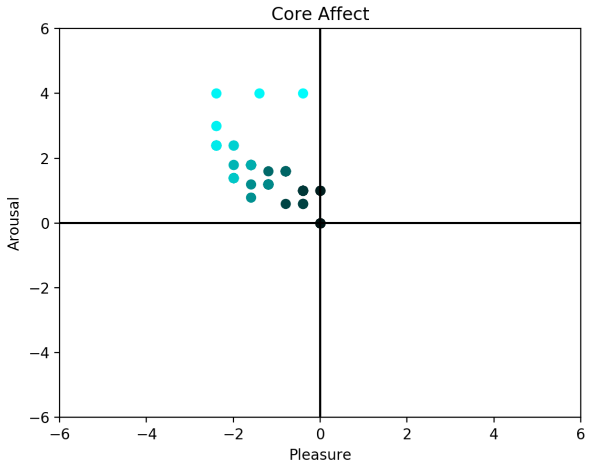
6 Conclusion

Our agent had considerably different affective responses when traversing each of the 4 different maps that we have tested.

On Map 1 (Fig. 2), the absence of any interactables throughout the map traversal led the agent's affective state to be on the negative-arousal and negative-pleasure quadrant during the entirety of the simulation (Fig. 3a and 3b). On Map 2 (Fig. 4), interactables were scattered throughout the maze, being, however, not relevant to the completion of the agent's objective. This made the agent's affective state to remain on the positive-arousal and negative-pleasure quadrant (Fig. 5a and 5b). On Map 3 (Fig. 6), not only the agent finds interactables but also doors that it needs to open in order to accomplish its objective. Both these things make the agent's affective state remain on the positive-arousal and positive-pleasure quadrant (Fig. 7a and 7b). Finally, on Map 4 (Fig. 8), the nature of the scenario made it so that the agent would have very different traversals depending on which doors it chose to open. Because of this we decided to study both the best and the worst case scenario. On the best case scenario, the agent quickly finds the chair, having an increase on the



(a)



(b)

Figure 5: The evolution of the agent's affective state in function of time (a) and space (b) as it traverses Map 2 (Fig. 4). The colours in (a) can be mapped to the 2-dimensional Core Affect space using Fig. 1. In (b), the dots become increasingly light blue as time passes. This means that the first affective measurement is a totally black dot, whereas the last measurement is a bright blue dot. In this map, the arousal dimension increases each time the agent finds a new interactable. The pleasure dimension, however, steadily decreases until the agent finds the door and opens it, finding the chair. The agent's affective state remains in the positive-arousal and negative-pleasure quadrant throughout the traversal.

pleasure dimension of its affective state as a result (Fig. 9a and 9b). On the worst case scenario, the agent continuously opens doors that he had already found and that don't lead to its objective, increasing its pleasure dimension but making the agent's arousal steadily decrease. In this case, the agent's affective state remains in the negative-arousal and positive-pleasure quadrant (Fig. 10a and 10b).

We can thus see how different maps arouse different affective states on the test agent. They also do so in ways that are to some extent predictable. It's not surprising that a map that consists of a monotonous maze will give a player neither arousal nor much pleasure whereas a maze with challenges that the player can solve and a number of objects to interact with will do the opposite.

However, predicting what will happen in maps of greater

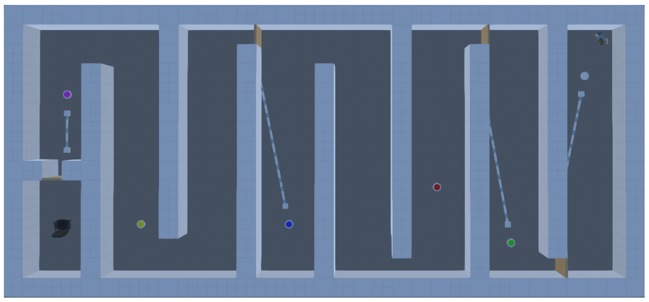


Figure 6: The layout of Map 3. In this map, the agent starts on the top right corner and must cross the maze, having to open 3 doors before reaching the final door, behind which is the chair. As opening these 3 doors is mandatory to reach the final goal, interacting with the 3 relevant buttons becomes a sub-goal. The agent will further find 2 interactables which are not relevant to the main goal before reaching the final door and the button which it has to press to find the chair.

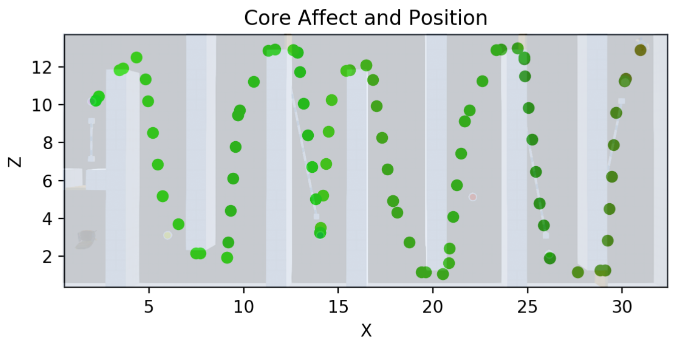
magnitude, which might have several sections with different densities of challenges and objects is not so trivial. The information this approach provides can allow developers to pinpoint exactly where changes need to be made in order to arouse in the players certain affective states. Designers might decide to purposely make a section of the map lower the arousal of a player in order to prepare him to a high arousal inducing location that appears soon afterwards.

The very different affective results gotten from traversing the same map according to different paths (Fig. 9 and 10) highlight the importance of the agent's decisions on its affective experience. Here we have chosen to show, on Map 4, the best and worst possible cases depending on the doors the agent decides to open. For the sake of brevity, we have not shown the expected average result, which would have the agent open half the doors before finding the chair. The agent's affetive state would in that case be in the same quadrant as the worst case scenario, but with a higher value in the arousal dimension.

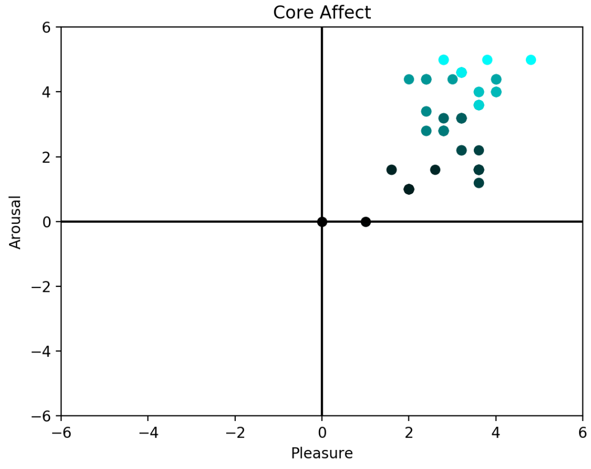
This paper aims to be a proof of concept, showing that UX testing agents are able to provide information that can be highly useful for both testing and designing. The fact such agents can show UX related information at precise moments and locations also allows them to relay information that would be very difficult to obtain with real users unless physiological methods were employed. Enquiring a user about her experience as she plays a game will inherently alter the user experience, but that is not the case with UX testing agents.

7 Discussion

The first questions that might come to the mind of the reader are: ”But how accurate is this information? How can I be sure the information the agents conveys does indeed reflect how a real user would feel and experience the system?”. Those are very good questions, which we are currently working on to be able to answer. With this paper, our goal was to show how one could employ agents in order to test UX and explore the type of information such agents could provide. We believe our results show that agents could help automate UX testing



(a)



(b)

Figure 7: The evolution of the agent's affective state in function of time (a) and space (b) as it traverses Map 3 (Fig. 6). The colours in (a) can be mapped to the 2-dimensional Core Affect space using Fig. 1. In (b), the dots become increasingly light blue as time passes. This means that the first affective measurement is a totally black dot, whereas the last measurement is a bright blue dot. We can see that both affective dimensions increase as the agent traverses the map. Arousal increases whenever the agent finds a new interactable and pleasure increases whenever the agent is able to accomplish a goal or sub-goal (opening doors). The agent's affective state remains in the positive-arousal and positive-pleasure quadrant throughout the traversal.

and make it an integral part of the development process without becoming a bottleneck. In the following paragraphs, we will discuss ways in which this UX testing agents could be improved and made accurate.

To ensure the results do indeed correlate with how users experience the system, user testing could be employed in order to fine tune the model to faithfully represent the affective changes users experience. This testing with real users could be done only once in order to attune the UX testing agents to the system under test, allowing the developed agents to be used automatically and without requiring user testing for the remainder of the system's life-cycle.

This ”tuning” process could be done, for example, using machine learning and user testing based on physiolog-

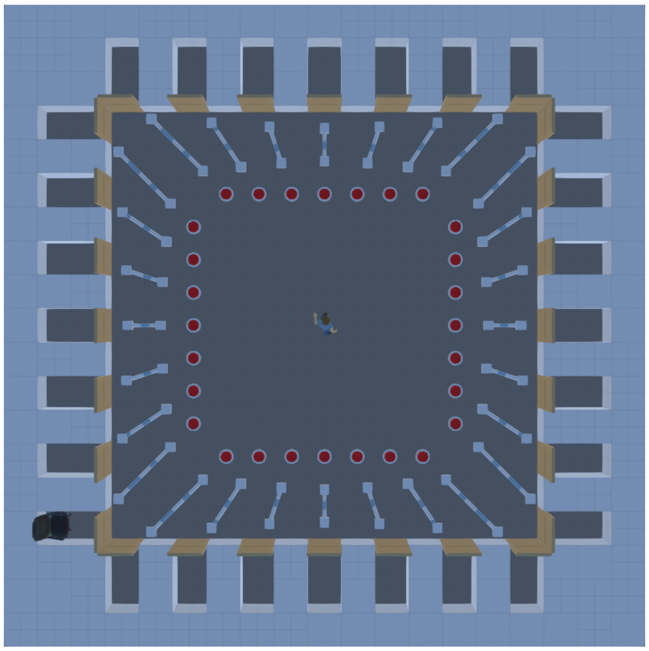
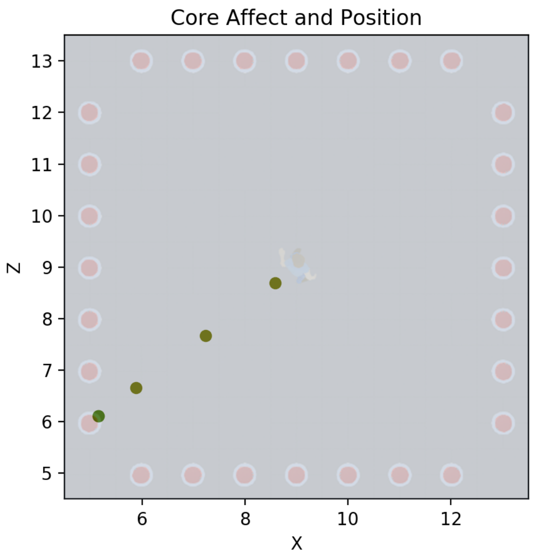


Figure 8: The layout of Map 4. This map is considerably different from the previous three, but the objective remains the same: finding the chair. The agent begins at the centre of the map and must press the correct button from the 28 buttons that surround it. All of the buttons open a corresponding door, but the chair is only behind one of them. The agent has no way of knowing behind which door the chair is.

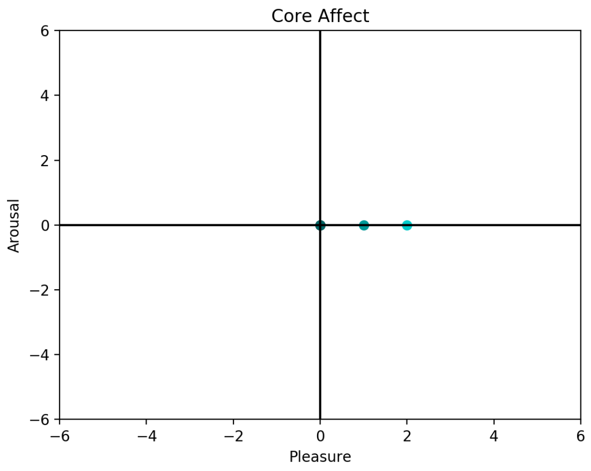
ical methods [Callejas-Cuervo *et al.*, 2017; Moon and Lee, 2016]. Users could be asked to interact with the system under test as both physiological measurements and game events are recorded. The game events, like player location and object interaction, could then be used as input for the machine learning model and an interpretation of the physiological measurements under the emotional model chosen could be the desired output. We could then fine tune the model parameters and the model itself to ensure they accurately represent how a real user experiences the system.

In order to have access to full emotional states instead of affective states, the model could be expanded to have an extra dimension, Dominance, and modified in order to be in accordance with the PAD model of emotions [Russell and Mehrabian, 1977]. This model claims emotions can be characterised in a 3-dimensional state, the dimensions being Pleasure, Arousal and Dominance.

The results of our tests make it seem the greater the number of interactables, the better the affective response. However, this might change as soon as we add a cognitive load component to our model. Cognitive load theory defends there is an upper limit to the number of new objects that a user can keep track of [Sweller, 2011]. This could mean that the presence of too many objects related to accomplishing the user's objective could in fact make the user enter a state of cognitive overload, leading to a negative UX. It is thus a viable expansion to the approach to not only improve the emotional model but also implement cognitive load considerations.



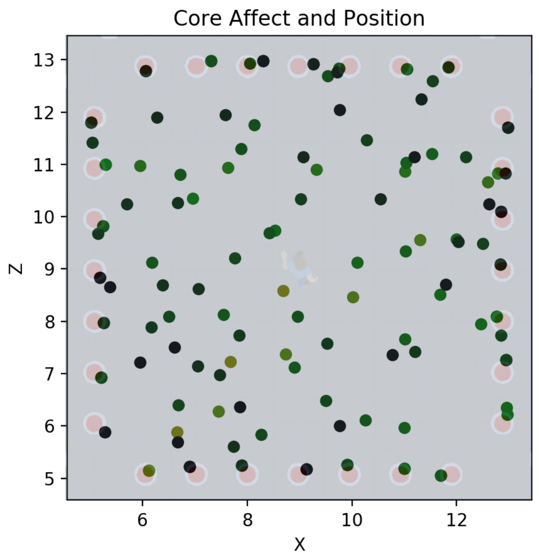
(a)



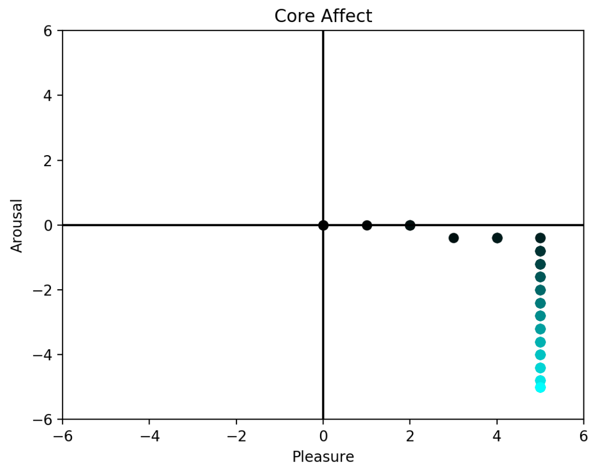
(b)

Figure 9: The evolution of the agent's affective state in function of time (a) and space (b) as it traverses Map 4 (Fig. 8) in the **best case scenario**. The colours in (a) can be mapped to the 2-dimensional Core Affect space using Fig. 1. In (b), the dots become increasingly light blue as time passes. This means that the first affective measurement is a totally black dot, whereas the last measurement is a bright blue dot. This traversal is a swift one, with the agent going directly to the right button and increasing its pleasure dimension. As the agent had found all the buttons since the start, the arousal dimension remains neutral.

The fact that different traversals of the same map lead to different affective results exposes the importance of developing agents that behave in the most "human like" way possible. If the agents make choices that a user would never do, then the UX information the agents convey might prove to be of no importance. It might also be of no interest to try and program agents to exploit systems in order to find ways of interacting which lead to a negative UX, as the agent might, in the exam-



(a)



(b)

Figure 10: The evolution of the agent's affective state in function of time (a) and space (b) as it traverses Map 4 (Fig. 8) in the **worst case scenario**. The colours in (a) can be mapped to the 2-dimensional Core Affect space using Fig. 1. In (b), the dots become increasingly light blue as time passes. This means that the first affective measurement is a totally black dot, whereas the last measurement is a bright blue dot. In this scenario, the pleasure dimension increases each time the agent is able to open a door by interacting with a button. As it never finds any new interactables, the agent's arousal dimension steadily decreases throughout the traversal. As such, the agent's affective state remains in the negative-arousal and positive-pleasure quadrant throughout the traversal.

ple of a game, decide to continuously walk against a wall and proceed to claim the game can be very uneventful. We thus defend that the development of UX testing agents will not only require well tuned models to assess UX, but also agents that behave as similarly to human users as possible.

Finally, we here present a preliminary approach to test a very simple game, but this approach could be modified to test a number of other systems and should not be interpreted as game-specific. Agents with a similar architecture could be used, for example, to test the interface of a store, with the pleasure and arousal dimensions being modelled based on the user finding an item of interest or being able to accomplish a successful purchase or using a coupon. We believe this approach could be used with any system where events that affect the core affect state of an user can be, to a certain degree, identified. This event identification could even be done automatically, using machine learning models and physiological measurements.

References

[Bellamy *et al.*, 2011] Rachel Bellamy, Bonnie John, and Sandra Kogan. Deploying cogtool: integrating quantitative usability assessment into real-world software development. In *Proceedings of the 33rd international conference on software engineering*, pages 691–700, 2011.

[Borovikov *et al.*, 2019] Igor Borovikov, Yunqi Zhao, Ahmad Beirami, Jesse Harder, John Kolen, James Pestrak, Jervis Pinto, Reza Pourabolghasem, Harold Chaput, Mohsen Sardari, et al. Winning isn't everything: Training agents to playtest modern games. In *AAAI Workshop on Reinforcement Learning in Games*, 2019.

[Callejas-Cuervo *et al.*, 2017] Mauro Callejas-Cuervo, Laura Alejandra Martínez-Tejada, and Andrea Catherine Alarcón-Aldana. Emotion recognition techniques using physiological signals and video games-systematic review. *Revista Facultad de Ingeniería*, 26(46):19–28, 2017.

[Conati and Maclaren, 2009] Cristina Conati and Heather Maclaren. Empirically building and evaluating a probabilistic model of user affect. *User Modeling and User-Adapted Interaction*, 19(3):267–303, 2009.

[de Mesentier Silva *et al.*, 2017] Fernando de Mesentier Silva, Scott Lee, Julian Togelius, and Andy Nealen. Ai-based playtesting of contemporary board games. In *Proceedings of the 12th International Conference on the Foundations of Digital Games*, pages 1–10, 2017.

[Desmarais and d Baker, 2012] Michel C Desmarais and Ryan SJ d Baker. A review of recent advances in learner and skill modeling in intelligent learning environments. *User Modeling and User-Adapted Interaction*, 22(1-2):9–38, 2012.

[DIS, 2010] ISO DIS. 9241-210: 2010. ergonomics of human system interaction-part 210: Human-centred design for interactive systems (formerly known as 13407). *International Standardization Organization (ISO). Switzerland*, 2010.

[García-Sánchez *et al.*, 2018] Pablo García-Sánchez, Alberto Tonda, Antonio M Mora, Giovanni Squillero, and Juan Julián Merelo. Automated playtesting in collectible card games using evolutionary algorithms: A case study in hearthstone. *Knowledge-Based Systems*, 153:133–146, 2018.

[Guerrero-Romero *et al.*, 2018] Cristina Guerrero-Romero, Simon M Lucas, and Diego Perez-Liebana. Using a team of general ai algorithms to assist game design and testing. In *2018 IEEE Conference on Computational Intelligence and Games (CIG)*, pages 1–8. IEEE, 2018.

[Holmgard *et al.*, 2018] Christoffer Holmgard, Michael Cerny Green, Antonios Liapis, and Julian Togelius. Automated playtesting with procedural personas with evolved heuristics. *IEEE Transactions on Games*, 2018.

[Kaber *et al.*, 2011] David B Kaber, Rebecca S Green, Sang-Hwan Kim, and Noa Segall. Assessing usability of human–machine interfaces for life science automation using computational cognitive models. *Intl. Journal of Human–Computer Interaction*, 27(6):481–504, 2011.

[Katsanos *et al.*, 2013] Christos Katsanos, Nikos Karousos, Nikolaos Tselios, Michalis Xenos, and Nikolaos Avouris. Klm form analyzer: automated evaluation of web form filling tasks using human performance models. In *IFIP Conference on Human-Computer Interaction*, pages 530–537. Springer, 2013.

[Kowalczuk and Czubenko, 2016] Zdzisław Kowalczuk and Michał Czubenko. Computational approaches to modeling artificial emotion–an overview of the proposed solutions. *Frontiers in Robotics and AI*, 3:21, 2016.

[Martinho *et al.*, 1999] Carlos Martinho, Isabel Machado, and Ana Paiva. A cognitive approach to affective user modeling. In *International Workshop on Affective Interactions*, pages 64–75. Springer, 1999.

[Memon *et al.*, 2000] Atif M Memon, Martha E Pollack, and Mary Lou Soffa. Automated test oracles for guis. *ACM SIGSOFT Software Engineering Notes*, 25(6):30–39, 2000.

[Moon and Lee, 2016] Seong-Eun Moon and Jong-Seok Lee. Implicit analysis of perceptual multimedia experience based on physiological response: a review. *IEEE Transactions on Multimedia*, 19(2):340–353, 2016.

[Ortony *et al.*, 1990] Andrew Ortony, Gerald L Clore, and Allan Collins. *The cognitive structure of emotions*. Cambridge university press, 1990.

[Pfau *et al.*, 2017] Johannes Pfau, Jan David Smeddinck, and Rainer Malaka. Automated game testing with icarus: Intelligent completion of adventure riddles via unsupervised solving. In *Extended Abstracts Publication of the Annual Symposium on Computer-Human Interaction in Play*, pages 153–164, 2017.

[Plutchik, 1980] Robert Plutchik. A general psychoevolutionary theory of emotion. In *Theories of emotion*, pages 3–33. Elsevier, 1980.

[Prasetya and Dastani, 2020] ISWB Prasetya and Mehdi Dastani. Aplib: An agent programming library for testing games. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, pages 1972–1974, 2020.

[Rivero and Conte, 2017] Luis Rivero and Tayana Conte. A systematic mapping study on research contributions on ux evaluation technologies. In *Proceedings of the XVI Brazilian Symposium on Human Factors in Computing Systems*, pages 1–10, 2017.

[Rumbell *et al.*, 2012] Timothy Rumbell, John Barnden, Susan Denham, and Thomas Wennekers. Emotions in autonomous agents: comparative analysis of mechanisms and functions. *Autonomous Agents and Multi-Agent Systems*, 25(1):1–45, 2012.

[Russell and Mehrabian, 1977] James A Russell and Albert Mehrabian. Evidence for a three-factor theory of emotions. *Journal of research in Personality*, 11(3):273–294, 1977.

[Russell, 2003] James A Russell. Core affect and the psychological construction of emotion. *Psychological review*, 110(1):145, 2003.

[Schrepp *et al.*, 2014] Martin Schrepp, Andreas Hinderks, and Jörg Thomaschewski. Applying the user experience questionnaire (ueq) in different evaluation scenarios. In *International Conference of Design, User Experience, and Usability*, pages 383–392. Springer, 2014.

[Stahlke *et al.*, 2019] Samantha Stahlke, Atiya Nova, and Pejman Mirza-Babaei. Artificial playfulness: A tool for automated agent-based playtesting. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–6, 2019.

[Sweller, 2011] John Sweller. Cognitive load theory. In *Psychology of learning and motivation*, volume 55, pages 37–76. Elsevier, 2011.

[Vermeeren *et al.*, 2010] Arnold POS Vermeeren, Effie Lai-Chong Law, Virpi Roto, Marianna Obrist, Jettie Hoonhout, and Kaisa 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*, pages 521–530, 2010.

[Vos *et al.*, 2015] Tanja EJ Vos, Peter M Kruse, Nelly Condori-Fernández, Sebastian Bauersfeld, and Joachim Wegener. Testar: Tool support for test automation at the user interface level. *International Journal of Information System Modeling and Design (IJISMD)*, 6(3):46–83, 2015.

ANNEX A3

Assessing Players' Cognitive Load in Games

Alberto Ramos
Department of Computer Science and Engineering
Instituto Superior Técnico
Lisbon, Portugal
alberto.ramos@tecnico.ulisboa.pt

***Abstract*—Due to the exponential growth of computer technologies, video games are becoming more complex each passing year; with tasks and challenges that, very often, defy the player's cognitive abilities. Handling limitations of the Working Memory and proper Cognitive Load management is crucial when dealing with problem-solving tasks; however, these concepts appear to be highly undervalued, or even unknown, in the gaming industry.**

To address this problem and help game designers to better understand the intrinsic complexity of their games, this work applies the attention-shifting principles of the Time-Based Resource Sharing (TBRS) Memory Model in the game Way Out (a game we have developed from scratch). We formulated the idea of Attention-Grabbing Events and tried to incorporate them into the game, aiming to create a tool-set that estimates the player's Cognitive Load while playing a video game. To validate our hypothesis, we compared the data collected from the game with the questionnaire NASA TLX – a subjective method that assesses the mental workload felt during a task.

Although we were unable to directly estimate the player's Cognitive Load, we believe that this work was a step forward towards achieving that goal. The amount of Attention-Grabbing Events and gameplay time, when compared with the NASA TLX, seem to be a good indicator of Cognitive Load levels; however, the TBRS Cognitive Load formula, in its current form, does not appear to be reliable when directly applied in a general gameplay scenario – at least following the approach we did.

***Index Terms*—Cognitive Load (CL), Working Memory (WM), Time-Based Resource Sharing (TBRS), Video Game, Game Development, NASA TLX.**

I. Introduction

Due to the exponential growth of computer technologies in the last decades, video games are becoming more complex and diversified than ever. From deep and intriguing storytelling to complex game mechanics, it is unquestionable that the video game industry is doing a proper job in keeping up with this growth and creating games that are becoming more realistic and immersive each passing year.

Back in the 70s and 80s, when video gaming was emerging and becoming mainstream, games were much simpler and had straightforward mechanics that a joystick and a few set of buttons could handle. Space Invaders, for example, a fixed shooter created by Tomohiro Nishikado in 1978 that is considered one of the most influential video games of all time, consists of controlling a space cannon horizontally while firing descending alien forces. The enemy spaceships approach the player more rapidly as time passes, making the game harder the longer it's played. The mechanics, however, are quite simple and easy to memorize.

Nowadays the story has diverged immensely – each year, thousands of new video games are released with complex mechanics that take much longer to master and require entire keyboards to be played with. An example of this can be observed in the game Dark Souls, an action role-playing game that was developed by FromSoftware and released in 2011. In this game, the player assumes the role of an undead character that explores the virtual kingdom of Lordran to seek the fate of his kind. A game well known for its hard boss fights that, in order to be beaten, forces the player to learn from past mistakes by memorizing the enemies movements and weaknesses. Demanding mechanics like these require a great amount of attention and cognitive resources and, if not dealt with accordingly, can easily lead to negative emotions such as frustration or anger.

Handling limitations of the Working Memory and proper Cognitive Load management is crucial when dealing with problem solving tasks and is proven to positively influence effective performance and learning [1]. Since the Working Memory has a limited capacity and is believed to only retain information for a small period of time of approximately twenty seconds, it is easily overloaded if more than a few chunks of information need to be simultaneously processed.

These limitations and concepts, which are highly important in neurological and physiological matters, appear to be quite undervalued and ignored in the gaming industry. If Cognitive Load and the overall correct management of Working Memory's resources are taken into consideration by game designers in early phases of game development, highly beneficial results could be obtained. By estimating the amount of Cognitive Load that a players' Working Memory is using while playing a video game, game designers would have, in addition to play testing feedback, an extra source of reliable information that would be an indicator of their game levels complexity. Hence, excessively demanding tasks could be detected and adjusted accordingly earlier, facilitating and cutting costs in the play testing phase and allowing the developers to focus on other aspects of the game.

Assuming it is possible to estimate the duration of time in which the players' attention was fully grabbed during a game, it is theoretically possible to apply the principles of a Memory Model to assess the players' Cognitive Load. Therefore, we hypothesise that if the attention-shifting principles of the Time-Based Resource Sharing (TBRS) Memory Model are incorporated in games, and if the model's formula to assess

Cognitive Load is correctly used, it would be possible to estimate the amount of cognitive resources used by a player's Working Memory, while playing a video game.

This work aims to confirm whether or not our hypothesis is valid, by integrating this model within a game that we have developed from scratch. We will compare the collected game data with a subjective method that also estimates a users' Cognitive Load during an activity – the NASA TLX questionnaire.

II. Background

The distinction between the nowadays called "Short-Term Memory" (STM) and "Long-Term Memory" (LTM) was firstly, somewhat, controversial. It was argued that such division was useless and would unnecessarily complicate the concept of memory. However, evidence that such division would, in fact, make sense, started to emerge around the 60s. A strong argument in favor of a dichotomy in the memory system was noticed by Milner, while studying patients with hippocampal lesions [2], who appeared to became incapable of either store or retrieve information from the LTM but could still process and register immediate input for short periods of time. This inspired R. C. Atkinson and R. M. Shiffrin to deepen the studies of the memory and the dichotomy of the LTM and this new "Short-Term Store", leading them to conceive the first Memory Model [3].

The Working Memory (WM), initially named Short-Term Store and, nowadays, often called Short-Term Memory, is now commonly known as a cognitive system crucial for reasoning and decision-making that can hold information for a short period of time. Additionally, contrary to the LTM, the WM has a limited capacity and a certain amount of resources available to properly work.

In the context of our work, **Cognitive Load (CL) refers to the amount of resources used by our WM to properly function** (*i.e.* to solve problems, learn novel information, react to stimulus, etc.). These resources are limited and need to be properly managed to avoid "cognitive overloading" [4].

A. Memory Models

While studying and analysing different Memory Models proposed over the years (e.g. Multi-Store Memory Model [3], Working Memory Model (1974 [5] and 2000 [6]), to better understand the core components that allow our Working Memory to properly function, we came across one that particularly grabbed our attention – the Time-Based Resource Sharing (TBRS) Memory Model, initially proposed by Barrouillet and Camos in 2004 [7].

This Memory Model explains how the WM functions, based on four main assumptions:
The **first**, is that both the processing and maintenance of information requires and share the same resource, which is attention.
The **second** assumption is that as soon as attention is switched away, the activation of the memory traces suffers from a time-related decay. Additionally, the refreshment of these decaying memories traces, requires their retrieval from memory by attentional focusing.
The **third** assumption is that any processing that captures attention, disrupts maintenance by preventing the refreshment of memory traces; therefore, WM functioning is limited by a central bottleneck.
Which leads to the **fourth** and final assumption: since attention can only be devoted to one process at the time, maintenance and processing cannot occur concurrently, meaning that, to maintain information in WM (to avoid forgetting) it is required that the individual regularly switches attention from processing. This means that the central bottleneck allows only one central process at time, **making the sharing of attention time-based**.

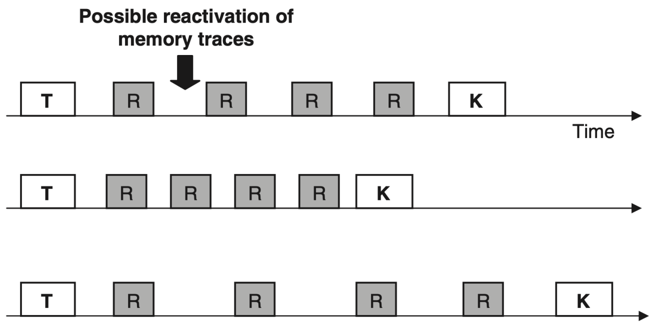


Fig. 1: Time Based Resource Sharing - "Reading digit span task" exercise.

To validate their hypothesis, they came up with a simple task where **participants were asked to maintain letters in memory while simultaneously performing a secondary task that involved reading a series of digits that were presented, one at the time, on a screen** (Figure 1).

The idea is that if time pressure is applied to a task as simple as this, it can easily become much more demanding. Thus if the digits from the secondary task are presented at a fast pace, maintaining the letters in WM becomes much harder since there is less time to reactivate memory traces – leading to a higher CL. However, if the digits are presented at a slow or comfortable pace, there is time to reactivate memory traces – leading to a low or moderate CL.

In the case of this model, **Cognitive Load refers to the total amount of time during which attention was fully captured** and can be formulated as:

$$CL = \frac{\sum_{i=1}^{N} a_i}{T} \tag{1}$$

a_i reflects the latency in which the i_{th} event fully captured attention.
T refers to the total duration of the task or activity.

If the total number of processes N is known, the formula can be simplified by using average processing times:

$$CL = \frac{\overline{a}N}{T} \tag{2}$$

To illustrate the concept of CL, *i.e.* the balance between the competing actions that are processing and maintenance, and grabbing the example from Figure 1; suppose that a participant has to say 10 letters out loud, each takes 200ms to be said and the time available is 4 seconds. The resulting CL of this example would be 10 x 200 / 4000 or 0.5. However, if the time available doubled, the resulting CL would be cut in half (10 x 200 / 8000 or 0.25).

B. Methods to assess Cognitive Load

When it comes to measuring CL, the main challenge is knowing if the methods used are valid, reliable and practical. Conventionally, there are two main approaches to assess the WM's capacity: **Objective** and **Subjective** [8].

The **Objective approach** mainly relies on behavioral data collected from the users while performing a task. Whilst commonly more reliable, this approach may affect a users' focus from the task itself, since it often requires the usage external and intrusive machinery. Direct objective measures include examples of dual-task methodologies, eye-tracking or task-invoked pupillary response and brain-activity measures.

The **Subjective approach** is probably the most common and, as the name implies, requires the subject to do some sort of self-report after completing a task. Usually these subjective self-reports require the subject to rate the perceived mental effort or task difficulty in a numerical scale and there are several different types of reports focusing on different problems. One of the great advantages of using self-reports is its simplicity, since it solely requires the appropriate set of questions for the activity that's being implemented on. Additionally, being subjective means that there is no need of using external equipment collecting behavioral data during an activity, making this a non-intrusive approach.

There are two most commonly used techniques for subjectively assessing mental workload [9], the **NASA TLX** and the **SWAT**. They both divide the workload in multiple subscales and are proven to provide quite similar results [9]. However, for the context of our work, since it assesses a wider variety of mental workload components involved in the experience, we ended up opting to use the NASA TLX technique.

C. NASA TLX

The NASA TLX (NASA Task Load Index), developed in 1981 by Sandra G. Hart of the NASA Ames Research Center [10] is one of the most known subjective techniques to assess CL. It has been used in various domains such as healthcare, aviation, and others of similar technical complexity. It is a subjective workload assessment technique that relies on a multidimensional construct to derive an overall workload score based on a weighted average of ratings on six subscales: Frustration, Effort, Temporal Demand, Physical Demand, Mental Demand and Performance. These sub-scales of the workload are based on the assumption that some combination of these dimensions are likely to represent the "workload" experienced by most people performing most tasks [11]. Three of the subscales focus on the demands imposed on the subject (mental, temporal and physical demand), whereas the other three explore the interaction of the subject with the task (effort, performance and frustration levels) .

NASA TLX consists of two parts: **weights** and **ratings**. Generally, the first requirement is for the participant to evaluate the contribution of each subscale – its weight – of the workload during the task (the weights themselves also provide diagnostic information as the nature of the workload imposed by the task).

To do so, there are 15 possible pairwise comparisons of the six subscales of workload. Each pair (for instance, Temporal Demand vs Mental Demand) is presented at the time and the subject has to chose the member of each pair that contributed more to the workload of the task performed (in our case, the game). At the end of every pairwise comparison, we count the number of times that each subscale is selected instead of the others. It can range from 0 (never selected in a pairwise comparison) to 5 (selected in every pairwise comparison) – this is the resulting weight assigned for that specific subscale.

The second requirement is to obtain individual numerical ratings for each subscale – which reflect the magnitude of that factor in the task. Thus, the respondents are asked to rate each subscale individually from 0 to 10 or 0 to 100 (least to most taxing).

The adjusted ratings for each of the six subscales of the workload is computed by multiplying their respective weight with their raw rating (Equation 3). For example, if the weight and rating of Temporal Demand was 4 and 50 respectively, its Adjusted Rating would be *4 x 50 = 200.*

$$AdjustedRating = Weight * RawRating \tag{3}$$

Using the NASA TLX, the overall workload of a task, *i.e.* its resulting CL, is the result of the sum of the Adjusted Ratings divided by 15 (which is the total amount of pairwise comparisons) (Equation 4).

$$Workload_{NASATLX} = \frac{\sum AdjustedRatings}{15} \tag{4}$$

III. Implementation

To test our hypothesis – whether or not is possible to estimate the player's Cognitive Load based on the attention-shifting principles of the TBRS – we decided to create a game from scratch; since it didn't impose restrictions in our creativity and gave us the necessary flexibility to create a satisfying game environment in which it made sense to fully test our hypothesis.

The chosen name for the game was – Way Out. The player plays as a golem who just woke up in a mysterious laboratory and is trying to figure out the purpose of his existence. To do so, he has to solve puzzles and challenges in a dungeon-like environment to both progress through the map and find clues about himself.

The hidden plot is that a human scientist has become the first to achieve full conscience transmutation. The puzzles the golem has to solve were created by the golem himself in his human form, and are a simple way to determine if the scientist's cognitive and reasoning skills have remained intact in his new body.

However, due to the nature of this work, the small demo that we have created and tested mainly explores the puzzles and challenges of the game and not the plot itself.

With the goal of analyzing possible CL variations, a total of four versions of the game were developed. Each version's puzzles had particular tweaks and changes to analyse this eventual discrepancy. These key particularities of the game will be explored in-depth in the following sections.

A. Attention-Grabbing Events

According to TBRS memory model, CL is the result of the total attention time of a task divided by the total time of that task (Formula 1).

With the goal of adapting the TBRS attention-shifting principles and its CL formula to game development, we decided the following: even though they are most likely very distinct from game to game, through any game the player has to execute certain actions or events to progress, which usually take a certain chunk of time to be performed. Whenever one of these events occurs, its duration (*i.e.* the time since the event begins until it ends) could be translated into a period of time in which the player's attention was supposedly shifted towards processing information. We call these – Attention-Grabbing Events (AGE).

Having the total gameplay and AGEs duration, it is theoretically possible to recreate the TBRS CL calculation. However, since all games are different and we are trying to generalize our model to cover any game type, **we highly emphasize that the game designers are the ones who should ponder and choose the AGEs, taking into account the type of game being developed.**

This being said, we will now explain which events were considered attention-grabbers in our game:

- **Object Interactions:** The time spent interacting with interactive objects (Figures 2(a) and 2(b)).

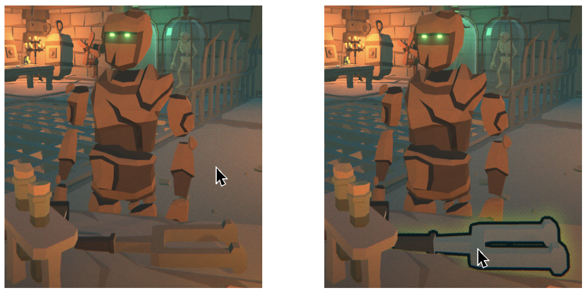

((a)) Mouse outside an interactive object ((b)) Mouse inside an interactive object

Fig. 2: Way Out: Object Interactions

- **Interface Interactions:** The time spent with the Inventory, Instructions, Notebook and Sphere placeholder interface opened (e.g. Figure 3).
- **Notifications:** Time in which notifications were shown on screen (e.g. Figure 4).

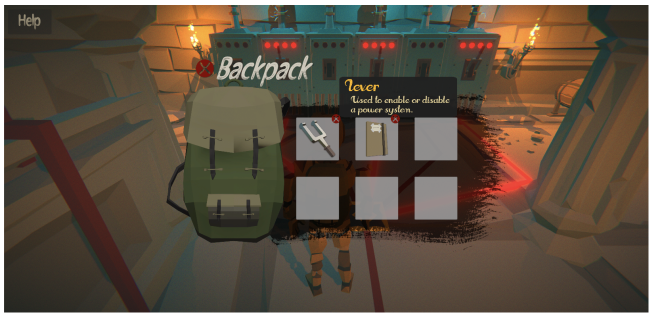


Fig. 3: Way Out: Interface Example (Inventory)

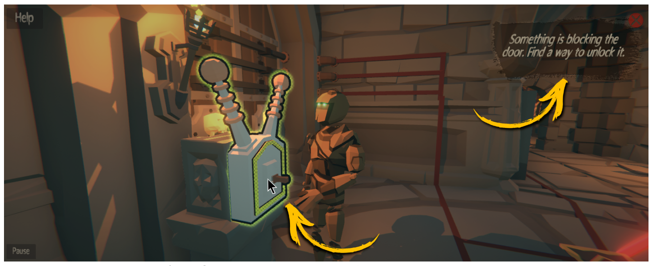


Fig. 4: Way Out: Notification Example

However, the overlap of events would interfere with the formula, since it would mean that the player's attention would be shifted towards processing multiple events at once. An example of this happening can also be observed in Figure 4, where the player is interacting with an object whilst a notification is simultaneously being displayed. Pressing the button triggered the notification, but the player kept hovering the interactive object.

Therefore, in order to mitigate these temporal overlays, we found useful to create an hierarchy for our game's AGEs (Figure 5).

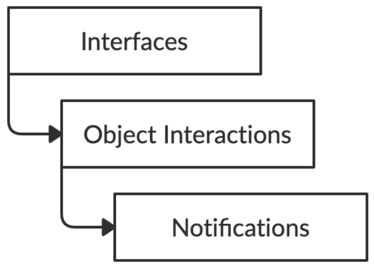


Fig. 5: Attention-grabbing Events hierarchy

We intentionally designed the interfaces to occupy a large chunk of the screen, so we assume that whenever the player is actively interacting with an UI element (e.g. the is Inventory open), object interactions and notifications become disabled, mitigating a possible overlap of attention.

The same principle applies to object interactions – if the player starts interacting with an object while a notification is being displayed on the screen, it is assumed that the player's attention is being shifted towards the interaction, not taking into consideration notification's display time in the equation.

B. Game Versions

According to the TBRS memory model, a task is more cognitive demanding when it requires a larger amount of attention shifting. In the case of our game, as mentioned in the previous section (III-A), we consider object interactions, notification and interface display times our main AGEs. Therefore – theoretically – for the same gameplay duration, the greater the number of AGEs, the greater the value of the CL will be.

That being said, and since the intrinsic difficulty of a task is proven to be correlated with higher levels of CL [9], we started by creating two versions of our game, **"Easy and "Hard"**, with the **goal of analyzing if the data collected from the Easy versions of the game would indicate lower levels of CL than the Hard ones**.

Both versions contain the exact same type of puzzles, challenges and possible interactions. However, the puzzles from the Easy version were intentionally twisted to require a lesser number of interactions for its resolutions, which would overall result in a less amount of AGEs.

Furthermore, we will also wanted to validate our model focusing on **time**. Once again, TBRS defends that the CL is the result of the attention time dedicated to a task divided by the time of that task [7]. However if the player is, for example, trying to solve a problem and has to move through the map without interacting with any objects, the gameplay time is counting but the attention time is not which, according to the formula (1), would result in a lower CL. And this is precisely the point that we want to verify. **If, for the same puzzles, in order to solve them the player is forced to move around the map, would this increase, maintain or decrease the player's CL?**

TABLE I: Game Versions.

	Normal Movement	Additional Movement
Easy	A1	B1
Hard	A2	B2

Thus, as seen in Table I, we ended up creating four versions of the game.

Two that solely explore the contrast between the intrinsic difficulty of the game – **A1** and **A2** – where all the items required for the resolution of the puzzles are all relatively close to each other, not forcing the player to move through the map in order to solve them. Note that **"Normal Movement"** means there is no extra movement, *i.e.* all the items required for the resolution of the puzzle are relatively close to each other.

The other two – **B1** and **B2** – beyond exploring the intrinsic difficulty of the puzzles, also explore the repercussions that the additional movement induced on the player has on the CL results. More specifically, the effects that additional gameplay time has on the player's CL. In these versions, the items required for the resolution of the puzzles are scattered around the map, forcing the player to move more through the map in order to solve them – hence the **"Additional Movement"** in Table I. This will theoretically increase the overall gameplay time and, consequently, using the TBRS CL formula, decrease the CL.

C. Game Puzzles

The puzzles implemented[1] were designed to verify the effects that the variations in AGEs and movement had on the players' CL. For that purpose, we have developed two main puzzles that slightly vary between the four versions of our game.

To test the discrepancies between the players' attention through the versions (A1 vs A2 and B1 vs B2), both puzzles require more or less AGEs for their resolution. To test the difference in the players movement through the versions (A1 vs B1 and A2 vs B2), *i.e.* more or less gameplay time, we changed the items disposition between the versions of the puzzles.

For instance, the first puzzle of our game – The Lever Puzzle – requires the player to move 6 levers to unlock a door. Initially, of the 6 levers, only 3 are correctly positioned and ready to move. For all four versions of the game, the player has to first find the 3 missing levers and place them on the machines that still require one.

The difference between the Easy and Hard versions of the game are the effects that the movement of each lever has on the other machines.

In the Easy versions (A1 and B1), each lever only affects its machine. For instance, Lever #3 only affects the state of Machine #3, turning it on (lever up) or off (lever down). Therefore, the easy versions' solution is fairly simple – the player solely has to find and place the missing levers correctly and turn on the machines (by moving each lever up).

On the other hand, in the Hard versions of the game, each lever movement can affect the state of multiple machines. For example, Lever 5 affects the state of Machines #4, #5 and #6, by either turning them on or off depending on their current state. This leads to a theoretical higher number of interactions – and AGEs – since the solution is not as straight forward as the opposite versions.

When it comes to the players' movement, the A and B versions of the game differ from the position of the machines – represented as white dots in Figure 6.

In both the A versions (Figure 6 (a)), all the machines are close to each other, allowing the player to clearly see the effect that each lever interaction has on the puzzle. While in the B versions (Figure 6 (b)), the machines are scattered around two rooms. Hence, to analyse the effect that each lever interaction has on the puzzle, the player has to move around.

D. Data Gathering and Cognitive Load calculation

To follow the principles of the TBRS memory model and in order to use its formula [7], we need to collect relevant gameplay data that estimates the players' attention time during the game. Therefore and, as mentioned on a previous section (Section III-A), beyond the **Total Gameplay Time**, we will mainly collect data related with the duration of the multiple AGEs that occur throughout the game (listed in Section III-A).

[1]A full walk-through of all the game puzzles and versions is available at: https://www.youtube.com/watch?v=95j85Add1Rg&ab_channel=AlbertoRamos

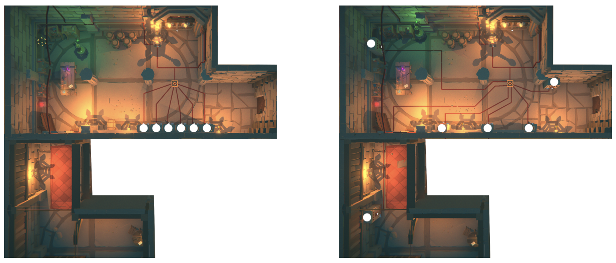

((**a**)) Versions A ((**b**)) Versions B

Fig. 6: Way Out: Lever Puzzle (Versions A and B)

The total sum of AGEs will return the **Total Attention Time** (Equation 5) during the game. The **Total Attention Time** will after be used as the dividend in the adapted TBRS CL formula; whereas the **Total Gameplay Time** will be the divisor (Equation 6).

$$TotalAttentionTime = \sum_{i=1}^{N} AGE \tag{5}$$

$$CL = \frac{TotalAttentionTime}{TotalGameplayTime} \tag{6}$$

Additionally, in order to support any possible unexpected values, we also collected the number of times each type of AGE happened (e.g number of times the Inventory was opened).

When the player completes the game, all the data listed above will be stored on a online Google Sheets document for further analysis.

IV. Procedure and Results

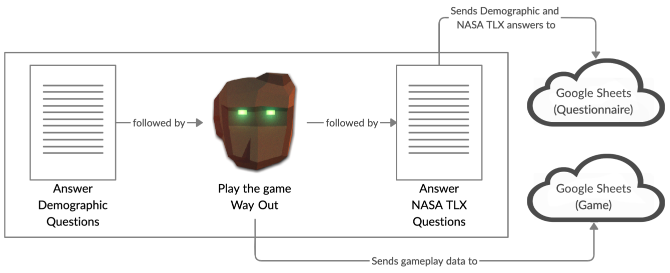

Sends Demographic and NASA TLX answers to
Answer Demographic Questions
followed by
Play the game Way Out
followed by
Answer NASA TLX Questions
Google Sheets (Questionnaire)
Google Sheets (Game)
Sends gameplay data to

Fig. 7: Procedure to acquire data

In short, the structure of the followed procedure is summarized in Figure 7.

With the goal of seeking basic information about the respondents and understand where they fit in the general population, the *first* part of the procedure consists of asking the participants the following demographic questions: “Age”, “Gender”, “Mother Tongue”, “How often do you play video games?”, “Do you enjoy point and click puzzle games?”.

Once collected, this data allows us, if needed, to divide the population of respondents in various groups, which will be useful in the overall analysis.

In the midst of the questionnaire, after answering the demographic questions, the participants are asked to play the game Way Out, which is the *second* part of the procedure – extensively explained in the previous section (Section III). After playing and finalising the game, a random code name is generated and provided to the player, so it can be pasted the questionnaire, linking the game data with the questionnaire answers.

The *third* and final part of the procedure consists of asking the participants questions related with their workload during the game, in order to validate our hypothesis. To do so, we need to compare the game data that may affect the CL with an existing valid and trustworthy method that accurately measures the workload of a task.

In a general sense we are examining the “workload” experienced by the player during the gameplay. Cognitive Load and Mental Workload are often used as synonyms and the relationship between workload factors and CL types was analysed in depth by Galy, Cariou and Mélan (2011) [9].

Therefore, after playing the game, the participants were asked to answer a few questions related with their overall workload during the game.

For that purpose, we will use the **NASA TLX** questionnaire [12] which was explained in detail in a previous section (Section II).

A. Pilot

Before broadening the experience to a larger sample of participants, we opted to first test it with a small sample – aiming to correct eventual game bugs and to better understand whether the questionnaire was adequate. During this phase, we specifically asked the participants to be extra critical and transparent, since our goal was precisely to adjust any eventual flaws with the experience.

Apart from a few game bugs pointed out, a consistent feedback received during this phase was that the pairwise comparisons, at the end of the questionnaire, were somewhat confusing. Some even went as far as saying that the comparisons looked all very similar and that “in the end, they selected almost randomly”. Discarding the pairwise comparisons is another way of using the questionnaire and often called – RAW TLX.

However, since the RAW TLX is a “trimmed” version of the NASA TLX without the pairwise comparisons, we ended up providing the full version of the questionnaire in the actual experiment; with the premise that the first thing to analyse was the possible discrepancies between the two versions (NASA TLX versus RAW TLX) – and whether or not it was justified to use the shorter version of the questionnaire, when analysing and comparing the collected data.

B. Sample

In total, we had a convenience sample of 54 participants responding to the questionnaire and playing the game. It is important to emphasise that all tests were done remotely. Hence, the experiment was advertised in multiple social platforms – namely Discord, Facebook and Instagram.

To analyse the obtained results, we used the software **SPSS Statistics (V26)** from IBM; where all the NASA TLX calculations were made and the charts, graphs and tables presented in this section were generated.

From the 54 participants, 45 (83.33%) identified themselves as males whilst 9 (16.67%) identified as females. The majority of our respondents (90.74%) speaks Portuguese as their native language while the other 10% speak others (such as English, Norwegian, German and Swedish).

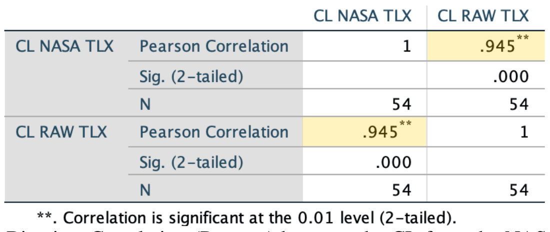
When asked how frequently they play video games, half (27 – exactly 50%) responded that they "made some time in their schedule to play video games", 17 respondents answered that they "play occasionally" and 10 "do not play video games often".

Lastly, when asked about how familiar they were with this game genre, 29 (53.70%) of our respondents answered that they "enjoyed and have played/watched others play multiple times", 20 (37.04%) were "not familiar or did not have a formed opinion" and only 5 (9.26%) "did not appreciate these types of games".

C. The questionnaire of choice: NASA TLX

To clarify a question brought up during the pilot tests phase (Section IV-A), we started our analysis by comparing the questionnaire results with and without the pairwise comparisons, *i.e.* by comparing the NASA TLX with the RAW TLX – to choose which version of the questionnaire would be more suitable to validate our hypothesis.

We found that there was a high positive correlation between the CL reported by the two versions of the questionnaire, with a nearly perfect Pearson correlation of 0.945 (as seen in Figure 8).



		CL NASA TLX	CL RAW TLX
CL NASA TLX	Pearson Correlation	1	.945**
	Sig. (2-tailed)		.000
	N	54	54
CL RAW TLX	Pearson Correlation	.945**	1
	Sig. (2-tailed)	.000	
	N	54	54

**. Correlation is significant at the 0.01 level (2-tailed).

Fig. 8: Bivariate Correlation (Pearson) between the CL from the NASA TLX and RAW TLX.

Since it is typically more common to use the full version of the questionnaire, and due the high positive correlation observed with its trimmed version, we opted to solely validate the gameplay data with the full version of the questionnaire – the NASA TLX.

D. Hyphothesis

According to our model, we hypothesise that the CL values will be higher in the Hard versions – A2 and B2 – where, theoretically, more AGEs occur. Additionally, we also want to observe the repercussions in CL when, for the same type of puzzles, the items required for their resolution are scattered around the map (A1 and A2 versus B1 and B2), forcing the player to move more and, consequently, increasing the overall gameplay time. More specifically, we were interested in finding out whether or not the hypothetical increase of gameplay time would affect the CL. Would it increase it, because the players were more consciously focused in shifting attention towards maintenance – to avoid forgetting the relevant and required items? Or would it decrease because the players have more time to process and maintain the information required for the resolution of the puzzles in WM?

To clarify our hypothesis, we started by analysing **gameplay time** (Figure 9) where, interestingly enough, we noticed that both the A versions took, in average, slightly longer to complete than the B versions. We ran a *Kruskal-Wallis* test that showed that there is at least one pair of significantly different groups (H(3) = 18.74 ; $p \leq .001$). The pairwise comparisons with a *Bonferroni* correction showed that the harder versions (A2 and B2) took significantly longer to complete than the easier versions ($p \leq .05$) – comparing A1 with A2 and B1 with B2.

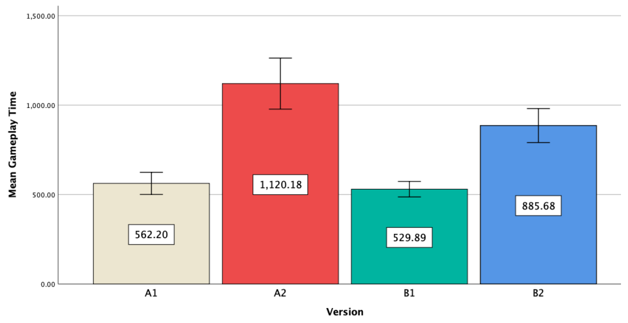
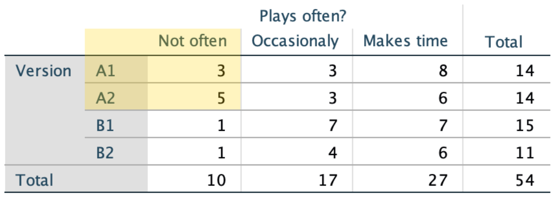


Fig. 9: Simple Bar Mean of Gameplay Time by the game versions; Error Bars refer to the Standard Error of the Mean (SEM).

Due to the game versions implementation, these results were unexpected – since we tried to implement the game in a way that the B versions would result, in average, in a higher gameplay time than the A versions. Therefore, we analysed two main things: The first was whether or not the A versions had a higher number of participants that did "not play often" than the B versions. As seen in Figure 10, we found that to be indeed true.



		Plays often?			
		Not often	Occasionaly	Makes time	Total
Version	A1	3	3	8	14
	A2	5	3	6	14
	B1	1	7	7	15
	B2	1	4	6	11
Total		10	17	27	54

Fig. 10: Table showing the distribution of the "Gameplay Frequency" groups across all four versions of the game.

The second, was to analyse if there was, in fact, a significant difference in gameplay time between the different "gameplay frequency" groups (*i.e.* "Does not play often", "Plays occasionally", "Makes time to play"). We ran a *Kruskal-Wallis* test that confirmed that there was, indeed, a significant difference; H(2) =10.320, p = .006. The pairwise comparisons with a *Bonferroni* correction showed that there is a significant difference in gameplay time between the groups "Occasionaly - Not Often" with p = .004, and "Makes time - Not Often" with p = .003.

To confirm whether these results were due to the groups distribution, we decided to analyse them without the 10 respondents that answered "Not Often" – considering them, in this specific analysis (Figure 11), as outliers.

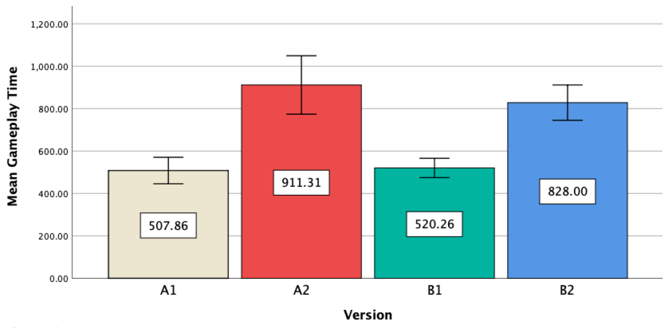


Fig. 11: Simple Bar Mean of Gameplay Time by the game versions – without the group "Not Often"; Error Bars refer to the Standard Error of the Mean (SEM).

As seen in Figure 11, discarding the respondents that answered "Not Often", the average gameplay time of the A versions decreased much more when compared with the B versions – A1 went from 562.20s to 507.86s and A2 from 1120.18s to 911.31s, while version B1 went from 529.89s to 520.26s and B2 from 885.68s to 828.00s. However, although closer, the average gameplay time between the A and B version was still very similar, which was not intended when designing the game.

This led us to conclude that our manipulation of the "Additional Movement" (B) versions was unsuccessful. Meaning that we were unable to answer one of the questions we initially had: "For two puzzles with a similar intrinsic difficulty, how would the variations in gameplay time affect the player's CL?"

Following the gameplay time, we analysed the other factor that, according to the TBRS, also influences the CL of a task – the **attention time**. Again, very briefly, for each player, the attention time results from the sum of the duration of every AGE during the gameplay. We ran a *Kruskal-Wallis* test to analyse the distribution of attention time across the different game versions (H(3) = 23.12; $p \leq .001$). The pairwise comparisons with a *Bonferroni* correction showed the same pattern found in gameplay time: A1 demanded significantly less attention time than A2 (p = .002) and B1, less attention than B2 (p = .011). As expected, the versions of the game with a higher difficulty (A2 and B2) had also, on average, a higher attention time (Figure 12).

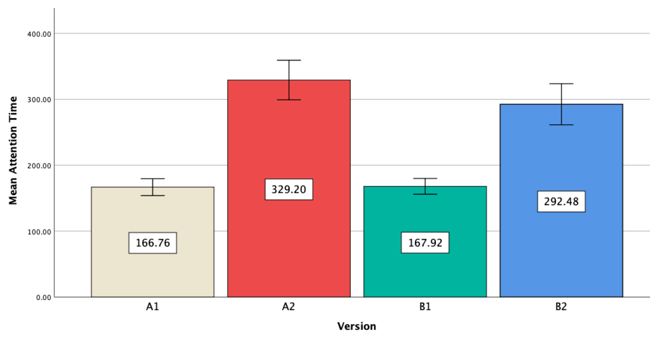


Fig. 12: Simple Bar Mean of Attention Time by the game versions; Error Bars refer to the Standard Error of the Mean (SEM).

Onto the actual **CL values** reported from the game (Figure 13), we can conclude they were very similar in every version (around 32%). We ran a *Kruskal-Wallis* test to analyse the distribution of the calculated CL (using the TBRS formula) across the different game versions (H(3) = .842; p = .839), and found that there were no statistically significant differences between the medians. These results, however, do not reflect the differences noticed in terms of gameplay and attention time; meaning that, perhaps, the adapted TBRS CL formula, in its current form, is not sensitive enough to detect the variations across the versions.

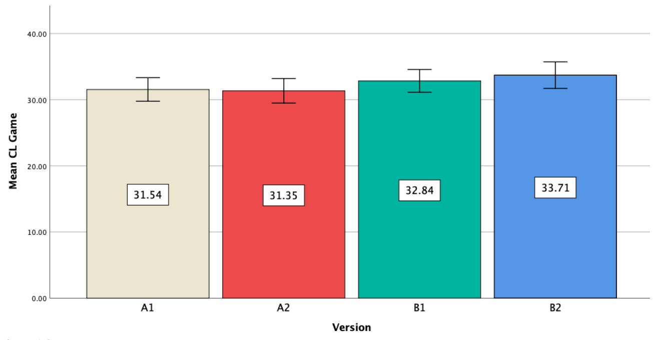
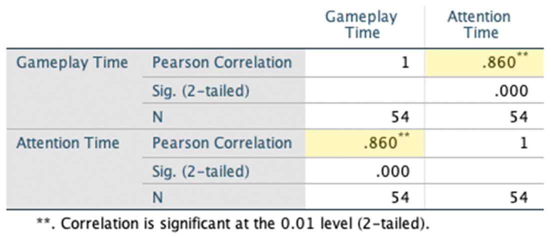


Fig. 13: Simple Bar Mean of the players CL percentages by the game versions (using the TBRS CL formula); Error Bars refer to the Standard Error of the Mean (SEM).

Observing both the **gameplay time** (Figure 9) and **attention time** (Figure 12) bar charts, a noticeable pattern can be seen – a higher gameplay time appears to result in a higher average of attention time. To clarify this, we made a Pearson correlation between these two variables (Figure 14) and we ended up observing a high positive correlation of 0.860. This means that, whenever the gameplay time increases, there is a high chance that the attention time will also follow that path.



		Gameplay Time	Attention Time
Gameplay Time	Pearson Correlation	1	.860**
	Sig. (2-tailed)		.000
	N	54	54
Attention Time	Pearson Correlation	.860**	1
	Sig. (2-tailed)	.000	
	N	54	54

**. Correlation is significant at the 0.01 level (2-tailed).

Fig. 14: Bivariate Correlation (Pearson) between the gameplay time and attention time

If both the dividend and divisor have a high positive correlation (*i.e.* in the equation, when one increases/decreases the other also follows that path) – the resulting CL will always be similar regardless of the times spent in the game – which justifies the results obtained in Figure 13. A possible way to mitigate this problem would be by significantly restricting the gameplay time and, for instance, by asking the player to complete as many tasks as possible in the time limit. However, since our goal was to generalize our hypothesis to any game type, we opted not to add a time restriction in our implementation.

Onto the **NASA TLX scores**, there is a noticeable CL variation across the game versions (Figure 15), leading us to observe two main things:

- According to the NASA TLX, as predicted, the players that played the more challenging versions of the game (A2 and B2), reported higher values of CL during the game (comparing A1 and B1 with A2 and B2).
- The "Additional Movement" (B) versions appear to have induced a slightly higher percentage of CL when compared with their respective "Normal Movement" (A) versions. This inclines us to assume that the distance between crucial items for the game appears to, in some way, affect the CL (comparing A1 with B1 and A2 with B2). Nevertheless, as discussed previously, the "Additional Movement" (B) versions were not successfully manipulated – preventing us from concluding anything concrete related to this topic.

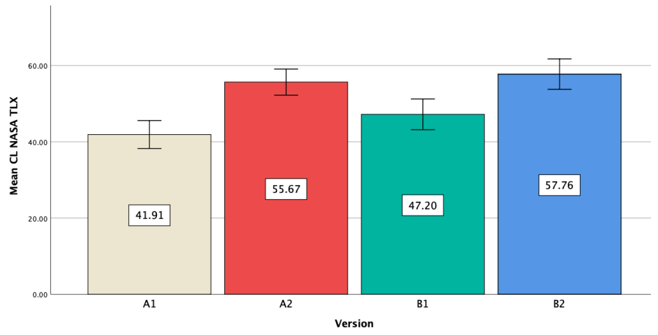
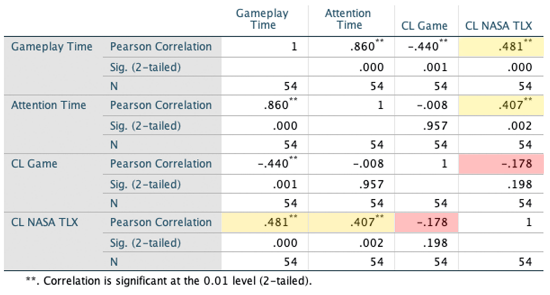


Fig. 15: Simple Bar Mean of the NASA TLX's CL by the game versions; Error Bars refer to the Standard Error of the Mean (SEM).

Comparing the NASA TLX scores with the CL obtained from the game, we notice that there is no correlation (Figure 16 highlighted with red). This can be justified by the same reason why the average CL reported from the game rounded the 32% for every version (Figure 13).

However, we also wanted to observe if there was a correlation between the NASA TLX scores and both the individual dimensions that, according to the TBRS memory model, affect the CL – gameplay time and attention time. Even though not perfect, as seen in Figure 16 (highlighted in yellow), there is a positive Pearson correlation between the NASA TLX scores with both the game times. This makes sense because the same pattern has been observed across the previous results: **The Hard versions (A2 and B2) resulted in significant longer game times (both total gameplay and attention) and higher NASA TLX scores; while the opposite was observed in the Easy versions (A1 and B2)**.



		Gameplay Time	Attention Time	CL Game	CL NASA TLX
Gameplay Time	Pearson Correlation	1	.860**	-.440**	.481**
	Sig. (2-tailed)		.000	.001	.000
	N	54	54	54	54
Attention Time	Pearson Correlation	.860**	1	-.008	.407**
	Sig. (2-tailed)	.000		.957	.002
	N	54	54	54	54
CL Game	Pearson Correlation	-.440**	-.008	1	-.178
	Sig. (2-tailed)	.001	.957		.198
	N	54	54	54	54
CL NASA TLX	Pearson Correlation	.481**	.407**	-.178	1
	Sig. (2-tailed)	.000	.002	.198	
	N	54	54	54	54

**. Correlation is significant at the 0.01 level (2-tailed).

Fig. 16: Bivariate Correlation (Pearson) between the gameplay time, attention time, CL from the game and CL from the NASA TLX.

V. Discussion

It is unquestionable that the video game industry is doing a proper job in keeping up with the exponential technological growth. Each passing year, thousands of games are launched with complex mechanics and challenges that, if not dealt with properly, can easily defy the limitations of the players WM. This work hypothesised that it was possible to assess the players CL based on their gameplay behaviours – and figuring out a way to accomplish it was our motivation.

The approach we took consisted of applying the attention-shifting principles of the TBRS Memory Model in the game Way Out (a game we have developed from scratch). Based on the model's principles, we formulated the idea of Attention-Grabbing Events (AGE) – which are periods of time during the gameplay in which the player's attention is most likely being grabbed. In Way Out, we considered the following events as attention-grabbers: object interactions, actively interacting with the game's UIs and display notification times. Having the total gameplay time and player's attention time, it would be possible to apply a formula similar with the one from TBRS to estimate the player's CL.

We implemented four versions of the game to manipulate two variables, each with two levels (a 2x2 factorial design): we manipulated the number of AGEs to analyse the repercussions that more or less AGEs had on the player's CL (versions A1 and A2); and we also manipulated how much players had to move around the map, aiming to see the effects that a longer gameplay time had on their CL (versions B1 and B2).

To validate our results, we opted to use the NASA TLX Questionnaire – a subjective approach that assesses the mental workload experienced during a task. The experiment was advertised across multiple social media platforms, and we ended up with a convenience sample of 54 participants. It consisted of answering a few demographic questions; followed by playing the game Way Out; and ended with the NASA TLX questionnaire.

The main variables we wanted to analyse across all game versions were: the total gameplay and attention times, the CL experienced by the players during the game (using the TBRS formula) and the resulting CL from NASA TLX (the players NASA TLX scores). While analysing the gameplay data – namely the gameplay time – we ended up with some unexpected results. The "Additional Movement" (B) versions took, in average, less time to complete than the "Normal Movement" (A) versions. Leading us to conclude that our manipulation of the B version was unsuccessful; and preventing us from answering a question we initially had: "For two puzzles with a similar intrinsic difficulty, how would the variations in gameplay time affect the player's CL?"

On the contrary, the game data indicated that our manipulation of the intrinsic difficulty of the puzzles was successful – the players that played the harder versions (A2 and B2) spent more time interacting with objects and playing the game, when compared with the players that played the easier versions (A1 and B1).

Using the TBRS CL formula to calculate the CL experienced by the players during the game, we noticed that it was nearly the same across all the game versions (around 32%). However, we also noticed that there was a high positive correlation between the gameplay and attention times; and, since the formula we used to calculate the CL results from the division of these two variables – the similar percentages of CL can be justified by this positive correlation. Nevertheless, we concluded that the TBRS CL formula, at least in its current form, is not sensitive enough to directly measure the player's CL in a gameplay scenario.

Finally, we analysed the NASA TLX scores, aiming to compare them with the game data. We noticed that the players that played the harder versions (A2 and B2) scored higher percentages of CL when compared with the ones that played the easier versions (A1 and B1). This led us to conclude that, although the TBRS formula does not appear to be sensitive enough to directly assess the player's CL, there was a positive correlation between the game times (both total gameplay and attention time) and the NASA TLX scores, meaning that – more AGEs and gameplay time resulted in higher scores of CL using the NASA TLX.

This was the first study that tried to assess the player's CL, in an automatic non-intrusive way, while playing a video game. Even though we were unable to directly estimate the player's CL, we believe that our work was a step forward towards achieving that goal. Based on the TBRS attention-shifting principles, the amount of AGEs and gameplay time, when compared with the NASA TLX scores, seem to be a good indicator of CL levels; however, the TBRS CL formula, in its current form, does not appear to be reliable when directly applied in a general gameplay scenario – at least following the approach we did.

VI. Limitations and Future Work

In order to strengthen our conclusion, a larger sample of players should be gathered – ideally with the same amount of participants for each different version and with similar gaming experience.

Directly following our work, it would be interesting to verify whether intrinsic time pressure in a similar game, using the TBRS adapted CL formula, would return more reliable results. In other words, would the direct division of the total attention time by the total restricted gameplay time, return similar CL values to the ones reported in a valid questionnaire (for instance, NASA TLX).

In addition, it would also be interesting to answer one of the questions that we initially had, but were unable to answer due to the unsuccessful manipulation of the "Additional Movement" (B) versions: How would the items disposition affect the player's CL? More specifically, how would the CL vary if the player had to memorize something crucial for the gameplay, but no AGEs happen for an extended period of time? For instance, the player retains a code sequence in WM that is written in a room, but that information is only useful after the player follows a long trail.

Even though our initial goal was to support game designers (especially during the testing phase) – by providing them with a toolset that measured the CL percentage experience by the players, while playing a video game – this work could be expanded in a broader set of fields. For instance when designing and implementing autonomous agents; where human-like behaviours, based on the available cognitive resources, could be improved by using the principles of the TBRS and attention-shifting in an approach similar to ours. In this scenario, game designers would also be the ones defining the AGEs, taking in consideration the environment in which the agents were situated.

References

[1] J. Sweller, J. J. G. Van Merrienboer, and F. Paas, "Cognitive architecture and instructional design," *Educational Psychology Review*, September 1998.

[2] W. Scoville and B. Milner, "Loss of recent memory after bilateral hippocampal lesions," *Journal of neurology, neurosurgery, and psychiatry*, February 1957.

[3] R. C. Atkinson and R. M. Shiffrin, "Human memory: A proposed system and its control processes." *In K. W. Spence and J. T. Spence (Eds.), The Psychology of learning and motivation: Advances in research and theory*, 1968.

[4] F. Paas, A. Renkl, and J. Sweller, "Cognitive load theory and instructional design: Recent developments," *Educational Psychologist*, June 2010.

[5] A. Baddeley and G. Hitch, *Working memory*. Academic Press, 1974.

[6] A. Baddeley, "The episodic buffer: A new component of working memory?" *Trends in cognitive sciences*, December 2000.

[7] P. Barrouillet and V. Camos, "The time-based resource-sharing model of working memory," *The Cognitive Neuroscience of Working Memory*, June 2007.

[8] J. Sweller, P. Ayres, and S. Kalyuga, *Cognitive Load Theory*, ser. Explorations in the Learning Sciences, Instructional Systems and Performance Technologies. Springer New York, 2011.

[9] E. Galy, M. Cariou, and C. Mélan, "What is the relationship between mental workload factors and cognitive load types?" *International journal of psychophysiology : official journal of the International Organization of Psychophysiology*, October 2011.

[10] S. Hart and L. Staveland, *Human Mental Workload*. Elsevier Science, 1988.

[11] S. Hart, "Nasa-task load index (nasa-tlx); 20 years later," in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, October 2006.

[12] A. Cao, K. Chintamani, A. Pandya, and R. Ellis, "Nasa tlx: Software for assessing subjective mental workload," *Behavior research methods*, March 2009.

ANNEX A4

Collaboration analysis in multi-player based simulations

Bruno Carreira
bruno.carreira@tecnico.ulisboa.pt

Instituto Superior Técnico, Lisboa, Portugal

December 2020

Abstract

This work consists on an approach aimed at helping developers of interactive software to test their scenarios, more specifically collaboration inducing maps. Our approach is based on using two different automated agents behavioural traces (policy), one specifying the scenarios Design Goal and the other an example of a non-Design Goal. After training said agents and by comparing the agents optimal behaviour when solving each scenario to the two policies we can determine if the scenarios allow to differentiate between the Design Goal and the non-Design Goal and order the scenarios from easiest to differentiate to hardest. Our approach was tested in two different environments, in a custom built simulator and in the iv4XR project game and AIGym Lab Recruits.

Keywords: Multi-player based simulation; scenario testing; collaboration detection; automated agents behavioural traces.

1. Introduction

We want to develop a method that helps designers of interactive software to test their scenarios and see if they can distinguish different behaviours.

When we think about problem solving, any type of problem, if we only receive the final answer, can we be totally certain that the problem was clearly understood? That the idealized steps/methods/behaviours for that problem were followed?

One example, in line with our work, is the use of serious games for team training simulation. When a company is focused on using training simulations to improve team collaboration, they need to be mindful of questions such as:

- Can they conclude that there was teamwork by simply observing if the team reaches the objective?
- What if the team did reach the objective but the individual members didn't work as a team and actually just acted by themselves?
- Did the scenario allowed them to have a choice between collaborating and not collaborating, or did the scenario simply forced them to work together? If so, can they really say that they learned to collaborate?

Answering these questions can be somewhat of a difficult task, especially during the developing phase of scenarios where there are constant changes in the maps, objectives, among others.

Emerging from the iv4XR project[1], our main goal is to use Reinforcement Learning (RL) policies to evaluate designed scenarios on their capability of allowing and distinguishing collaborative and non-collaborative behaviours. These policies will be extracted from automated agents developed for this work and corroborated from real-life users.

We will consider different scenarios in two different examples:

- Squary-Shappy - Self-developed game/simulator for this work.
- Lab Recruits - AIGym and game developed for the iv4XR project.

2. Related Work

The approach taken for our work derives from areas such as Automated Playtesting with Agents and Collaborative Behaviour, specifically definition of collaborative and non-collaborative behaviour. In this section we present a condensed description of work previously made related to the approach we here present, and that also served as inspiration.

2.1. Automated playtesting with Agents

Aiming at decreasing the time taken and the manual overhead that playtesting causes has led to the

[1]www.iv4xr-project.eu/

use of automatic playtesting. The main objective behind automated playtesting consists in the use of virtual and autonomous agents that will test the software gathering data only previously obtained by real-life users. Correlated to our work, automated playtesting with agents is commonly associated to the games industry where automatically testing scenarios/maps is one of the main focuses since it could be used to help the content creation process in game development.

Silva et al. [2] demonstrated how the use of different intelligent agents allowed them to evaluate game rules and map variants in the game *Ticket To Ride*. This work consisted on the use of several game related heuristic-based automated agents to simulate common users strategies and analyse several map variations of the game. Their approach allowed them to identify several key observations specific to the game such as how different strategies matchup in the different game/map variations, discover specific states not covered by the game rules, undesirable cities (paths) taken as well as the most probable routes to win in each map.

Holmgard et al. [6] present a method that consists on using player modeling to create automated agents as game-personas enabling them to automatically playtest and evaluate content in the game *MiniDungeons 2*, more specifically the maps/levels. After several experiences, described by the authors, they observed that use of automated agents as personas to analyse game levels in games with similar complexity to the *MiniDungeons 2* proved to be helpful when evaluating the game levels, more significantly, identifying which maps were better suited for each type of players. Similar to [6], Mugrai et al. [8] presented their work on the use of different human-like agents and strategies, meaning game personas, to playtest various levels of the game *Match-3*. Supported by a user study, their experiences proved that the use of different player perspectives and strategies allowed them to properly new evaluate level designs of the game, specifically in measuring what can be perceived as the difficulty of levels.

The research presented in this Subsection provides an indication that analysis of games rules and scenarios/maps can be done using specifications of design goals (strategies, game objectives, among others) in automated agents, which is inline with this works objective of using automated agents as a way of evaluating game-like scenarios by identifying collaborative and non-collaborative behaviours (design goals).

2.2. Collaborative Behaviour

To understand how to differentiate and recreate collaborative and non-collaborative behaviours for our automated agents, we must understand how collaboration works and is developed in both human-human and agent-agent interactions.

2.2.1 Human-Human collaboration

Most of, if not all, living entities have used collaboration in several aspects of their lives since it improves individual production [5] and humans have been collaborating since the early stages of our existence until the modern day, whether it's hunting animals in a prehistoric age or raising a modern building [9], working as a team towards a shared goal has been a common behaviour for our species.

Although every situation is unique and every group interacts in a different manner, there are several collaboration aspects that are considered universal such as, clear and open communication [10], consensus about goals and methods for completing tasks [4], clear definitions on each contributors role [11], placing group goals above individual satisfactions [1], among others. The successful implementation of these elements through communication and coordination is the key for teams/-groups to work effectively in a collaborative manner. Stephen et al. [3] researching high performing teams suggested that most experience teams develop a shared mental model in order effectively coordinate and predict each others movements while also improving their communication. Shared mental models (SMM) are referred to as knowledge structures for group members to similarly understand the team objectives, individual needs and responsibilities. This allows for teams to equally visualize the problem at hand and anticipate future actions and states. No matter the complexity, or simplicity of a task, the use of shared mental models has been corroborated as a collaborative indicator in human-human collaboration [7].

2.2.2 Agent-agent collaboration

Creating collaborative agents can be a problem. How can we have artificial agents working as team? Putting group goals above their own? How can we create agents that really work as a team and not just happened to have the same individuals goals?

Different answers and methods have be researched and used, but one of the most common techniques to teach agents on how to collaborate is to create a system similar to what high-performing human teams develop, a shared mental model. The SMM method has been widely researched

([14], [15]) and similarly to humans, a mental model means an internal representation of a situation and for agents provides a cognitive structure that connects and extends the notion of individual goals and needs to a team context [12].

Although the use of an SMM is proven to be an effective method to teach agents how to collaborate, using a centralized method provides us with a similar, more simpler approach and fundamentally collaborative agents. A centralized approach aims at providing a complete scheme of the general current state [13] (intrinsic communication) to all agents as well guaranteeing that by executing coordinated joint actions, the best action for both agents (collaboration) is always chosen.

The work presented in this Subsection served as an inspiration on how to create our automated agents and have them inherently act in both our Desired (collaborative) and Undesired (non-collaborative) behaviours.

3. Methodology

In this section, we will describe the approach used to create, detect and analyse behaviour in each scenario and a detailed explanation of the two environments used to test our solution. For both environments we will detail the map designing process, the different architectures of the automatic agents, the algorithms implemented to train the agents as well the optimization methods for said training. We will also present a description of the real-life users playtesting experiment made with the Lab Recruits game.

3.1. Behaviour Comparison Approach

In a software testing method the main premise is to check if the software reaches its main purpose, the Design Goal (DG). For a game developer the DG can be to create an engaging experience, for a developer of an intelligent tutoring system a DG could be to ensure that the students acquire the knowledge components and for a developer of a serious game for team training simulation, the DG may be that the users work as a team. Although this is strongly related to regular software testing, the main difference is that in our case the Design Goal can not be specified only in terms of logic properties of the software but also need to take into account human factors.

3.1.1 How to define the Design Goals?

Design Goals can be defined in many different ways such as examples of the intended behaviour, a reward/cost function that induces the behavior, a set of rules and many others. Here are some examples:

- Behavioural traces - The DG can be directly specified via designer provided demonstrations of the desired behavior as well as different behaviours that are to be tested. For instance if the DG is to train users on how to work as a team, the designer might provide traces of a group of people working as a team and traces of people not working as a team.
- Consequential descriptions - Instead of focusing directly on the behaviour, the DG can be a specification of the consequences the behaviour must have. Those consequences can be external or internal to the user. For instance, if the DG is to provide an engaging simulation experience, the designer might provide a description of the envisioned emotional state of the user.
- Goal descriptions - Another alternative way is to provide the goal of the behaviour and not the behaviour itself. In a reinforcement learning perspective, while the behavioural descriptions provide the policy, the goal description will provide the reward.

In this work, our Design Goal is to have agents/real-life users playing in a collaborative manner. For this we want to check if it's possible to distinguish collaborative and non-collaborative behaviours in different scenarios by comparing Behavioural Traces.

3.1.2 How to test a scenario?

We will now understand how we can test if a given scenario achieves a desired Design Goal.

Considering that some DG are very abstract there are numerous situations where we cannot guarantee if a DG is achieved, but, in most cases, we can determinate if a DG is not achieved. A clear example is if our DG is to understand if a user can learn basic arithmetic's. If we present a user with $(2\times2+2)$ and he responds 6, although is the correct answer, we are still not sure if the user understood that $\times$ has priority in relation to $+$.

Meaning that for any of the different ways to specify the Design Goals we must get a set of behaviours that achieve the desired goals π_g and another set that does not verify the design goals $\pi_{\bar{g}}$.

Figure 1 explains this difference. On the left side we have a system that given only a desired behaviour is able to decide if given scenarios fulfilled the DG or not, but, like we've seen, this doesn't always work. Whereas on the right side we consider that by having the definition of the desired behavior and the undesired behavior allows the system to then decide if not only the DG is fulfilled but also if the non DG ($\neg DG$) is not.

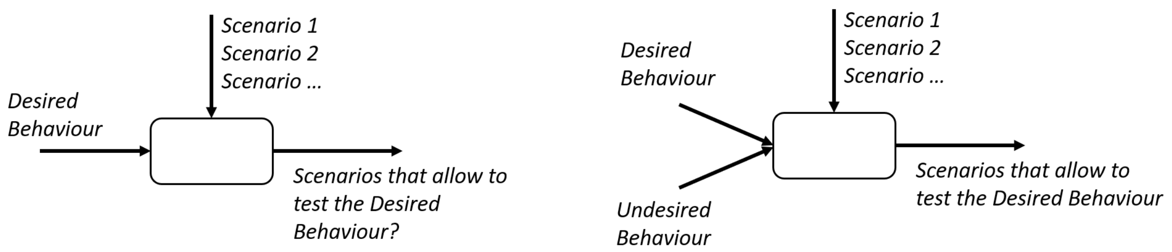


Figure 1: How to test a scenario

Given this difference, our approach is based on testing each scenario on the DG and on a specification of $\neg DG$[2] to determine if we can distinguish them and order the scenarios on their ability to differentiate both behaviours.

3.2. Squary-Shappy

In order to perform initial tests to our approach, we started by creating our own simulation gym, a Python based program called Squary-Shappy that, due to its simplicity, served as the initial environment to create automatic agents and implement our approach. The Squary-Shappy scenarios simulate a 2D object collecting game where agents roam around the map trying to eat as much food as they find. The food is positioned around the map and does not regenerate, meaning the food is limited by the number initially deployed. The maps are also all designed in a closed room format, meaning the agents cannot move out of the established bounds.

3.2.1 Map Designing

Our initial work in map designing consisted of creating 1-Dimensional maps to test our agents. Since 1-Dimensional maps are somewhat restrictive we quickly moved on to 2-Dimensional maps. The maps in the Squary-Shappy environment were designed in individual *.txt* files which allowed us to quickly create new scenarios and alter existing ones, since that, at their core, they were a simple deterministic matrix using a basic symbol format:

- '.' (dot) - Represents empty locations where the agents could move to.
- '1' - Represents walls, objects that agents can not pass through, visually identified by a black square.
- '2' - Represents the objects (food) that agents would aim to collect, providing them with a positive reward, visually identified by a green square.
- '3' - Represents the location of the first agent, visually identified by a blue square.
- '4' - Represents the location of the second agent, visually identified by a red square.

$$\begin{matrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1\\
1 & . & . & . & . & . & . & . & 2 & 1\\
1 & . & . & . & . & 3 & . & . & . & 1\\
1 & . & . & . & . & . & . & . & . & 1\\
1 & . & . & . & . & . & . & . & . & 1\\
1 & . & . & . & . & 4 & . & . & . & 1\\
1 & . & . & . & . & . & . & . & . & 1\\
1 & 2 & . & . & . & . & . & 1 & 2 & 1\\
1 & . & . & . & . & . & . & . & . & 1\\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1
\end{matrix} \tag{1}$$

Figure 2: Squary - Shappy: 2D map matrix example

This method of designing the maps as matrices also brought advantages when it came to train the agents, since any change/action/movement performed in the map could be made by simply changing values in matrix, a process that has very low processing cost.

In order to create a visual aspect of the simulation we used the Pygame [3] module which includes a number of easy to use libraries for computer graphics and game designing in Python.

The maps created for this work in this environment were the following:

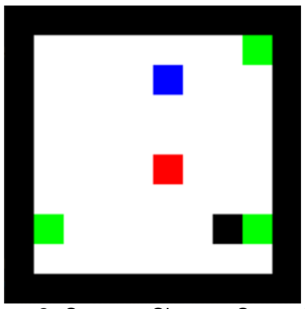


Figure 3: Squary - Shappy: Scenario 1

[2]Although there are clearly many different amounts of possible $\neg DG$, for specific problems, such as ours, the main effects can be easily identified.

[3]https://www.pygame.org/

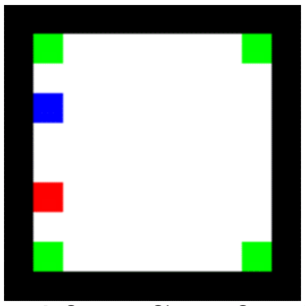


Figure 4: Squary - Shappy: Scenario 2

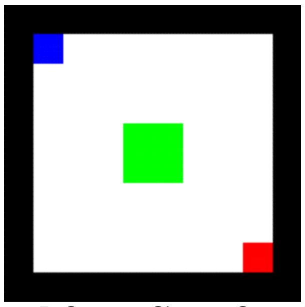


Figure 5: Squary - Shappy: Scenario 3

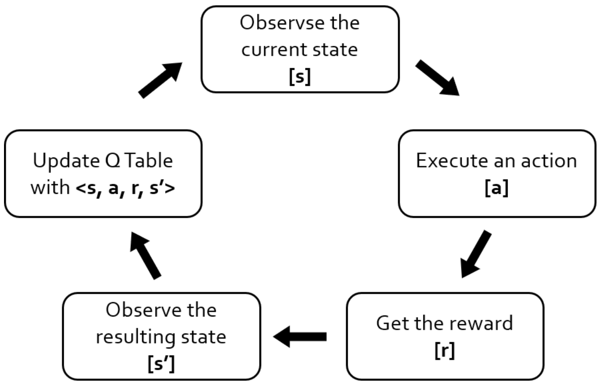
3.2.2 Agents

To automatically create the previously mentioned desired and undesired behaviours we started by developing two different agent architectures and training them using RL.

To define our Desired Behaviour, we implemented as an agent architecture a centralized method. The Centralized agent type means that all agents are a part of a shared mind that takes into account the effort made by every single agent, rewards the agents as a unit, not individuals and fundamentally makes them function as an optimal collaborative group/team. We choose to use the centralized architecture since our goal is not to teach agents to collaborate, but to create inherently collaborative and coordinated agents. On the other hand, our Undesired Behaviour implies that our agents work in a completely individual manner, focusing on their own personal gain. This behaviour was achieved implementing the Individual agents architecture. For these type of agents, each have an individual ”mind” and make decisions based on their own personal gain, meaning they prioritize actions that provide themselves with the highest individual reward and don’t take into consideration other agents.

In order to train the agents we chose to use a simple Reinforcement Learning approach called Markov Decision Process (MDP) with Q-Learning.

An MDP is predicated on the Markov Property ”The future is independent of the past given the present” which implies that in a RL problem, the next state *N+1* only depends on the current state *N*. The MDP with Q-Learning algorithm cycle is easily explained in Figure 6.


Observse the current state [s]
Execute an action [a]
Get the reward [r]
Observe the resulting state [s’]
Update Q Table with <s, a, r, s’>
Figure 6: MDP with Q-Learning

This image shows that during the training, agents observe their current state and chose an action to execute. They will then receive a reward (positive or negative) and observe the state of the environment after performing the selected action. This information will then be used to update the Q-Table, the agents ”brain”, by means of the Q-Learning formula:

$$Q(s,a) = Q(s,a) + \alpha \times [r + \gamma \times \max(Q(s',a')) - Q(s,a)]$$

Where Q(s,a) represents the Q-value of the performed action in the current state, α represents the learning rate, *r* represents the reward, γ the discount factor of future rewards and max(Q(s’, a’)) the maximum Q-value for any action of the next state.

The scenario was formulated into a RL problem by defining the MDP parameters as such:

- States: Like we previously mentioned, in these maps the objective is eat as much food as possible. Since the agents are only be able to move one step at a time and the food position is static, we decided to describe the state of the environment using the positions of both the agents (all agents in the Centralized type and own position in the Individual) as well as all the food objects. Every time a food object was picked, that position was removed from the state.
  - Individual: {PosAgent, PosFood1, PosFood2, PosFood3,...}
    - Example: {<2, 2>, <1, 1>, <4, 8>, <6, 1>, ...}
  - Centralized: {PosAgent1, PosAgent2, PosFood1, PosFood2, PosFood3, ...}
    - Example: {<2, 2>, <6, 3>, <1, 1>, <4, 8>, <6, 1>, ...}

- Actions: In the Squary-Shappy gym, the agents don't possess any type of pathfinding method and since the simulation made in a simple 2D matrix, the agents can only perform the low-level actions. For the Centralized type agents, their actions were formulated in pairs.
  - Individual:
    - NOTHING = Stay in the same place.
    - UP = Take one step up.
    - DOWN = Take one step down.
    - LEFT = Take one step left.
    - RIGHT = Take one step right.
  - Centralized:
    - [NOTHING, NOTHING] = Both agents stay in the same place.
    - [NOTHING, UP] = First agent stays in the same place and the second one moves one step up.
    - ...
    - [RIGHT, RIGHT] = Both agents take one step right.
- Rewards: For the rewards we took a simple approach. Every time an agent picked up a food object they were positively rewarded. If the agent didn't move they didn't receive any reward, positive or negative. For every step an agent took, they would get a small punishment, negative reward. The reward system for the Centralized agents functioned as a collective, meaning all agents were either rewarded or punished the combined amount, whereas for the Individual type agents, they were attributed their own rewards.

```
if(Agent collects Food)
    reward = +100
else if (Agent doesn't move)
    reward = 0
else
    reward = -1
```

Since the scenarios were considerably small and easy to solve, both types of agents, Centralized and Individual, performed 100.000 training episodes which would end when the agents either completed the objective, collecting all of the food, or performed the maximum number of centralized actions, 64.

The agents exploration-exploitation factor for decision making was based on a value $\in [0, 1]$. The probability of making random decisions started as 1 and decreased by a ratio of $\frac{1}{totalEpisodes}$, meaning that in the first episode, the agents had completely random decision making (maximum exploration) and during the course of the training those decisions started to be made less and less randomly, until the final episode where the agents had a random decision probability of ≈ 0, which means maximum exploitation.

3.3. Lab Recruits

Upon creating the Squary-Shappy gym and implementing our approach in it, we moved on to a more high-end environment, the Lab Recruits game designed by the University of Utrecht for the iv4XR project. In this game, the players objective is to click the target green button(s). For this to happen, the characters have to roam around the maps and click red buttons to open doors and access new rooms. Contrary to the Squary-Shappy food objects, these buttons do not disappear and could be clicked unlimited times, turning them ON/OFF to open/close doors.

For the Lab Recruits game, the iv4XR framework ran all its tests and simulations in a Unity environment. Every move/action/change done in the map had to be passed to the Unity application through a socket, applied using the Unity game physics engine and relayed back to the Java application through the same socket.

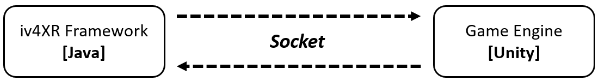


Figure 7: iv4XR Framework to Unity architecture

Although this method provides more accurate data regarding to map positioning, physics, among others, the amount of time needed to train agents in real-time would be too much. For this reason, and since for this work and in our scenarios the Unity physics could be considered meaningless, we decided to create a model of the map that could be simulated in a 2D matrix, similar to the Squary-Shappy environment, to help speed up the agents training process and still provide the wanted policies.

3.3.1 Map Designing

In the Lab Recruits gym, the map designing followed a very similar approach to the one we took in the Squary-Shappy scenarios, meaning that we used a clear symbol format, previously defined by the University of Utrecht, that could easily build/alter scenarios in a .csv file. Although the game itself (Unity executable) possessed physics, the maps were designed as a 2D matrix using discrete units which also lead for the agents to train in that manner, meaning that 1 step in the agents training meant several physical steps to go through the same distance in the game.

These were the three maps created for the Lab Recruits experiments:

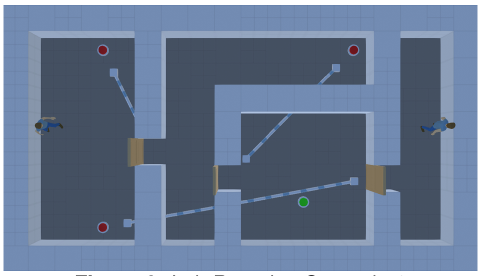


Figure 8: Lab Recruits: Scenario 1

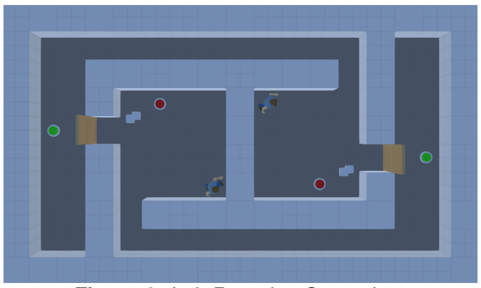


Figure 9: Lab Recruits: Scenario 2

3.3.2 Agents

For this environment, we used the same approach for the agents. We again developed Centralized agents who are a part of a shared mental mind and Individual agents that have their own individual decision process and only take into account their own state and benefit.

The development and training of the agents was done inside the Java iv4XR framework instead of creating a python module [4] since an MDP is a fairly easy algorithm to implement in Java when compared to pairing a Python module to the iv4XR Java framework.

In order to train the agents, we once again used an MDP with Q-Learning since it provided simplicity in the implementation and it had also demonstrated good initial results in the Squary-Shappy scenarios.

Like previously explained, instead of training the agents using the game itself, we developed a model to simulate the environment of the Lab Recruits. For the agents training we defined the following MDP parameters:

- States: In the Lab Recruits games, the agents interact with buttons to open and close doors until reaching the target button. Since the state of the doors (open or closed) are the only visual feedback that real life players have, we choose to use this as part of the agents MDP state. The doors internal state can be described as:

[4] Python is a more used commonly language for Machine Learning due to the large number of libraries, documentation and its simple coding language syntax.

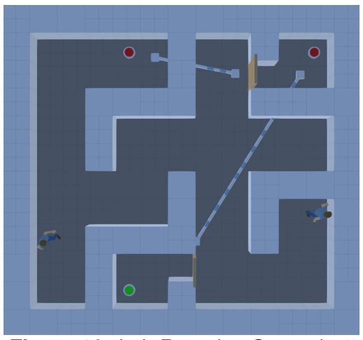


Figure 10: Lab Recruits: Scenario 3

  - 0 = Door is closed.
  - 1 = Door is open.

  Appending this to the agents position resulted in the following policy states:

  - Centralized: {PosAgent1, PosAgent2, IntStateDoor1, IntStateDoor2, ...}
    - Example: {<3, 0, 8>, <10, 0, 3>, 0, 1, ...}
  - Individual: {PosAgent, IntStateDoor1, IntStateDoor2, ...}

- Actions: Although the iv4Xr had a built-in pathfinding system with high-level actions we decided to use the basic low-level actions and added one specific action related to the Lab Recruits game, pressing on a game button. The agents could perform the following actions.

  - Individual:
    - NOTHING = Stay in the same place.
    - UP = Take one step up.
    - DOWN = Take one step down.
    - LEFT = Take one step left.
    - RIGHT = Take one step right.
    - PRESS = If the character is on top of a button, press it.

  For the Centralized type agents to act coordinatively, they're actions were again in pairs with all possible permutations.

  - Centralized:
    - [NOTHING, NOTHING] = Both agents stay in the same place.
    - [NOTHING, UP] = One agent stay in the same place and the other moves one step up.
    - ...
    - [PRESS, PRESS] = Both agents try to press buttons.

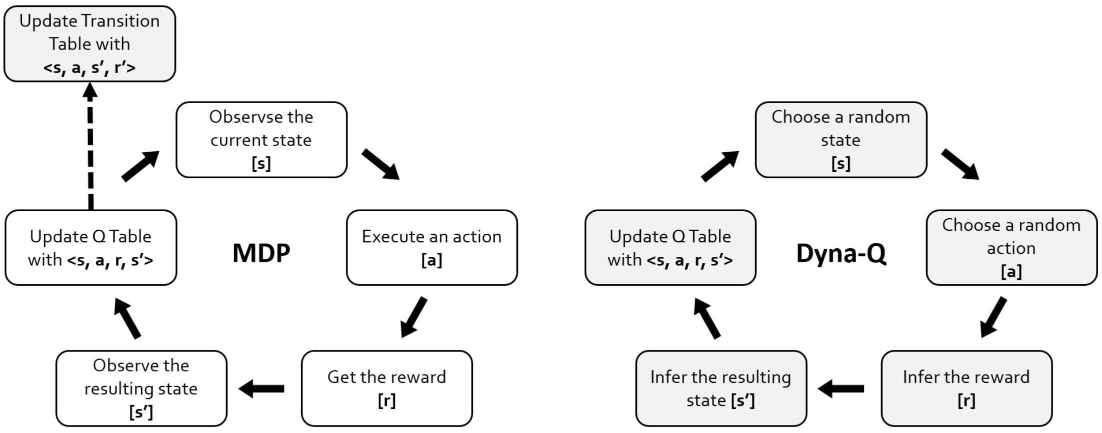

Update Transition Table with <s, a, s', r'>
Observse the current state [s]
Execute an action [a]
Get the reward [r]
Observe the resulting state [s']
Update Q Table with <s, a, r, s'>
MDP
Choose a random state [s]
Choose a random action [a]
Infer the reward [r]
Infer the resulting state [s']
Update Q Table with <s, a, r, s'>
Dyna-Q

Figure 11: MDP + Dyna-Q

- Rewards: Agents will get positively rewarded for reaching the target (turning ON a target button), no reward for doing nothing and a negative reward for all other actions. The only thing we had to make sure was, since the buttons could be clicked unlimited times, the agents would only get a positive reward the first time a target button was turned ON.

```
if(Agent turns ON TargetButton and
  isFirstTime)
     reward = +100
else if (Agent does nothing)
     reward = 0
else
     reward = -1
```

Contrary to the agents training made in Squary-Shappy, since the Lab Recruits game presented more complex scenarios with different sizes, we did not defined a defined maximum number of training episodes, but instead we evaluated the agents performance (number of steps until reaching the objective) and ended the training once that value converged.

3.3.3 Optimizations

One difficulty that we encountered in the Lab Recruits game was the time taken to train the agents, even using a 2D matrix model representation of the game. This could attributed to the size of the maps and the large number of combinations between all states and actions.

In order to try speed up the process, we implemented an optimization method for the agents training process, the Dyna-Q.

Dyna-Q

The Dyna-Q algorithm, shown in Figure 11, aims at collecting the agents past experiences and using them to further update the policy without the need for the agents to physically perform an action in the simulation world. Upon each cycle (after each Q table update) the agents save their experience by updating a Transition table. The Transition table contains the information that if an agent is in a state S and performs action A, then the resulting state will be S' and will receive reward R'. Upon each training episode, the agents enter a state of "hallucination" [5] where they use the information inside the table to virtually replay several random experiences and update the Q table with the resulting values.

3.3.4 User Playtesting

Although the purpose of our work is to help developers test scenarios that need to promote collaborative behaviours without the need of real-life users, in order to understand if the agents policies could be deemed as "human-like behaviour", we decided to have a playtesting experience with users to corroborate, or not, the agents behaviour and our results.

An online form was created containing all the information regarding the experience and a link to download the game. To try and create the same environment as our automated agents, users, in pairs, were asked to play each of the scenarios presented in Subsection 3.3.1 three times (since agents had several training episodes), while trying to reach the objectives as fast as possible and also avoiding having any type of communication between themselves. After completing all the scenarios, users received a randomly generated ID in-game to insert on the form and continue to the questions section.

In the questionnaire and in order to minimize possible opinion persuasion between users, they were asked to respond the same questions but in individual sections. The questions selected were

[5] Hallucination - Commonly used to describe a Dyna-Q cycle since no actions are done in the real world, only the information in the agents memory (Transition table).

created to help understand the internal decision process from the users to verify our systems accuracy. The questions, repeated for each scenario, were:

- When playing Scenario X, did you feel that you played as a team, played in an individually manner or neither? - To help us determine if our system was identifying the correct behaviour when comparing the users to our agents, or if our agents were indeed simulating human-like behaviour.
- Did you had any communication while playing Scenario X? If so, what was it about? - To verify if the main condition of no communication was followed.
- When playing Scenario X, describe your approach. Were you just trying to finish it as fast as possible? Were you exploring the map? - To understand if the users behavioural was made with the same objective as the agents, to finish as fast as possible.

In the end we also asked each user to order the Scenarios by how they promoted collaborative behaviour, 1 being the Scenario that promoted the most and 3 the Scenario that promoted the least. Although this is a high relative question, we thought it would be interesting to then compare to the results we got from the agents.

This playtesting experience was made entirely online meaning we could not properly guarantee the exact same playing parameters for the agents and users regarding communication, understanding of the game controls and the scenarios objective, among others.

3.4. How to distinguish behaviours?

Like we previously explained, our main objective is to help developers of collaborative inducing software to test their scenarios by comparing behavioural traces of a DG and a $\neg DG$ to understand if the scenarios allow to distinguish collaborative and non-collaborative behaviour. In order to achieve that, we created a way of comparing any behavioural trace with the Centralized agents policy (DG) and the Individual agents policies ($\neg DG$).

Many forms of comparison could be made. Imagining that we have a testing behavioural trace, a possible way was to simply count at every game-state if the action made was the best action possible according to each policy, Centralized and Individual, and in the end we could identify if a behavioural trace followed the DG or the $\neg DG$. But what if the actions made were not the best possible actions at each game-state, but the second best? Or the third best? We had to define a voting system to make sure every action, best or not, was accounted for.

The voting system was made using the following process. For each game-state and for both policies, each action was ranked from best to worst.

- Best action = 1.
- Worst action = size of policy's Action space.

After making said ranking, for every game-state in a provided testing behavioural trace, a vote of similarity with the Individual policy will be made with the following equation:

$$V_{(s,a_1,a_2)} = e^{-\frac{r(s,a_1)}{size(A)} \times \frac{r(s,a_2)}{size(A)}} \tag{2}$$

Where *s* represents the current state, a_1 the action made by character1, a_2 the action made by character2, *r(s, a)* represents the corresponding action ranking in the current state, *size(A)* the size of the individual Action space and V_{s,a_1,a_2} represents the vote value for the set of actions in the current state.[6]

Regarding the vote of similarity for the Centralized Policy, an equivalent equation is used:

$$V_{(s,a)} = e^{-\frac{r(s,a)}{size(A)}} \tag{4}$$

Where again *s* represents the current state, *a* represents the centralized action (action pair) of both characters, *r(s, a)* the corresponding action ranking in the current state and *size(A)* the size of the centralized Action space.

After summing each voting values, for both policies, during a provided game run, we normalize them in range [0,1] and we get a general similarity value to both policies, an indication if the provided game behavioural trace is more similar to the DG or the $\neg DG$ and with which difference.

4. Results

In this chapter, we will explained the results gathered from our experiments in both environments, the Squary-Shappy simulation and the Lab Recruits game.

4.1. Automated agents

We began our experiments using our own custom simulator, Squary - Shappy. Like we previously explained, in this environment the objective is for two agents to roam around the map and eat the all the food objects.

[6] An example of this is if in a particular state character1 performed the best action and character2 performed the second-best action according to the Individual Policy, the value received for that game-state, with an action space with size 4, would be:

$$V_{(s,a1,a2)} = e^{-\frac{1}{4} \times \frac{2}{4}} \approx 0.88 \tag{3}$$

In order to understand if we could differentiate the DG and the $\neg DG$, we used both agents types optimal behaviour (always perform the best actions) when solving each of the three scenarios presented in Subsection 3.2.1 and, using the voting system, compared those game runs to both policies, Centralized and Individual. These were the results gathered:

Scenario 1	Centralized agents optimal behaviour	Individual agents optimal behaviour
Centralized Policy	1	0,72
Individual Policy	0,91	1

Figure 12: Results: Squary-Shappy Scenario 1

Scenario 2	Centralized agents optimal behaviour	Individual agents optimal behaviour
Centralized Policy	1	0,76
Individual Policy	0,95	1

Figure 13: Results: Squary-Shappy Scenario 2

Scenario 3	Centralized agents optimal behaviour	Individual agents optimal behaviour
Centralized Policy	1	0,62
Individual Policy	0,78	1

Figure 14: Results: Squary-Shappy Scenario 3

Moving to the Lab Recruits game where agents, and now also real-life users, had to click on a sequence of buttons to achieve each maps final objective, we ran each agent type optimal behaviour on the three Lab Recruits scenarios referred in Subsection 3.3.1, compared them to both policies and we got the following results:

Scenario 1	Centralized agents optimal behaviour	Individual agents optimal behaviour
Centralized Policy	1	0,59
Individual Policy	0,92	1

Figure 15: Results: Lab Recruits Scenario 1

Scenario 2	Centralized agents optimal behaviour	Individual agents optimal behaviour
Centralized Policy	1	0,83
Individual Policy	0,91	1

Figure 16: Results: Lab Recruits Scenario 2

Scenario 3	Centralized agents optimal behaviour	Individual agents optimal behaviour
Centralized Policy	1	0,47
Individual Policy	0,93	1

Figure 17: Results: Lab Recruits Scenario 3

By analysing these comparison results, from both environments, we can extrapolate several observations.

Like expected, each agent type had a maximum similarity value (1) to their policies, meaning that the Centralized agents optimal behaviour performed the best actions at every game-state according to the Centralized policy and the same occurred for the Individual agents optimal behaviour and the Individual Policy.

Since our objective at its core is to evaluate scenarios,we can observe that by averaging the difference between the similarity values in each scenario, we can obtain an order on how the scenarios allow to distinguish the DG and $\neg DG$, from best (highest value) to worst (lowest value):

For the Squary-Shappy environment, we have:

$$Scenario1_{SS} = \frac{|1-0.91|+|0.72-1|}{2} = 0.185 \tag{5}$$

$$Scenario2_{SS} = \frac{|1-0.95|+|0.76-1|}{2} = 0.145 \tag{6}$$

$$Scenario3_{SS} = \frac{|1-0.78|+|0.62-1|}{2} = 0.3 \tag{7}$$

Meaning that, Scenario 3 is the map where we can more easily differentiate if the agents optimal behaviour followed the Centralized Policy or the Individual Policy, followed by Scenario 1 and Scenario 2.

By applying the same method to the Lab Recruits scenarios we have:

$$Scenario1_{LabR} = \frac{|1-0.92|+|0.59-1|}{2} = 0.245 \tag{8}$$

$$Scenario2_{LabR} = \frac{|1-0.91|+|0.83-1|}{2} = 0.13 \tag{9}$$

$$Scenario3_{LabR} = \frac{|1-0.93|+|0.47-1|}{2} = 0.3 \tag{10}$$

And by coincidence, we realize that the best Lab Recruits scenario to distinguish both behaviours is also Scenario 3, followed by Scenario 1 and ultimately Scenario 2.

4.1.1 Lab Recruits - Users playtesting

Like we previously mentioned, in order to corroborate if the results gathered from our automated agents and our approach could be used to simulate human-like behaviour, we performed a playtesting where we asked users to play each of the Lab Recruits scenarios three times and to individually answer a questionnaire regarding their behaviour whilst playing. For comparison purposes, the results here presented are only related to the third try on each scenario. This way we feel that we somewhat approximate the playing environment of the agents (which had training episodes before reaching the optimal behaviour) to the users (considering their first two tries as their ”training”). We receive 13 playtesting experiences and questionnaires responses, totaling 26 individual participants. Here we present the results.

Regarding the last question of ordering the Scenarios by how they promoted collaborative behaviour, Scenario 3 was considered the one that most promoted, followed by Scenario 1 and ultimately Scenario 2.

This order, chosen by the users and equal to the one calculated by our approach, is also backed up by the users answers on whether they played as a team (collaboratively) or as individuals (non-collaboratively) in each of the scenarios, since in Scenario 3 we had the highest percentage of users that responded that they played in a collaborative behaviour with 77% and the remaining users, 23% responding they played in a non-collaborative manner. In Scenario 1, 65% of the users said that they played in a collaborative manner, while the other 35% responded that they played with a non-collaborative behaviour. Finally in Scenario 2 we had the lowest difference of answers, since 54% of the users answered that they played in a collaborative manner and the remaining 46% of the users responded that they played with a non-collaborative behaviour.

In order to understand if our agents could be considered proper representatives of collaborative and non-collaborative human-like behaviour in the scenarios, we decided to compare each of the behavioural traces from the users playtesting experiences to the Centralized and Individual policies using the same voting formula used to compare the agents optimal behaviours to the same policies. Unfortunately we were not able to properly label the users behaviours, since our voting system always attributed the Individual policy with a higher vote of similarity. This result can attributed to a combination of several possible reasons.

One of the possible reasons is the differences between the agents and the users playing environment. Although we tried to approximate the users playing environment to the agents, some variables could not be fully guaranteed. Even giving users three tries at each scenario and only comparing the third try, we cannot be sure that users truly understood the scenarios objective. Although indicating users not to communicate, since the experiment was made online and with users playing on the same computer, next to each other, the no communication rule can not be guaranteed, plus some users responded in the questionnaire saying that they did communicate regarding which person should open a door, who should do what, among other planing topics.

One other possibility is the voting system not making a valid comparison. Although the voting system did identified the agents optimal behaviour to the proper policy, the same did not happened when comparing the users to the policies, this indicates the voting system could need to be changed/improved.

Another possible reason is the Centralized agents not correctly simulating human-like behaviour. Although the use of a centralized policy guaranties that the agents behave with total coordination, having the combined effort/reward of the team in ”mind” and acting in the fastest and most optimal way of collaboratively solving each scenario, it is impossible to guarantee that this policy has incorporated all the collaborative behaviour possibilities humans may have in each scenario.

5. Conclusion and Future Work

Aiming at evaluating designed scenarios on their capability of allowing and distinguishing collaborative and non-collaborative behaviours, we developed two types of automated agents to represent the Design Goal, centralized agents, and the non-Design Goal, individual agents, in different scenarios from the Squary-Shappy simulator and the Lab Recruits game. Upon training said agents we used a voting system to compare each agents type optimal behaviour when solving the scenarios to both policies. This comparison allowed us to observe in each scenario if we could differentiate the two behaviours and also order the scenarios by their ability of distinguishing collaborative and non-collaborative behaviours. For the Lab Recruits game we also had a playtesting experience with real-life users where, although we could not properly label each users behavioural trace to a collaborative or non-collaborative behaviour, we did managed to corroborate our scenarios order using the questionnaire answered by the users.

The results gathered from this work gives us an indication that this approach of using a definition of a Design Goal and a non-Design Goal as behavioural traces from automated agents allows to analyse the scenarios ability to allow and distinguish collaborative and non-collaborative behaviours, which can provide developers with initial data regarding their scenarios. We also believe that this approach could be extended to analyse other types of behaviour.

There are several possibilities for future work. We believe that although analysing the actions made by RL agents shows promising results, we acknowledge that defining the Design Goals and the non-Design Goals using Inverse Reinforcement Learning to more accurately represent human-like behaviour could achieve more accurate results. Another possibility is the combination of this approach to the analysis of agents and users internal state. This could allow developers to not only determine if their scenarios are being developed the intended way but also have a more specific indication of which areas and components of the scenarios are affecting the agents and users decision making.

References

[1] D. Bagshow, M. Lepp, and C. Zorn. International research collaboration : Building teams and managing conflicts. *Conflict Resolution Quarterly*, 24:433–446, 2007.

[2] F. De Mesentier Silva, S. Lee, J. Togelius, and A. Nealen. Ai-based playtesting of contemporary board games. pages 1–10, 08 2017.

[3] S. J. Fiore S.M. Process mapping and shared cognition: Teamwork and the development of shared problem models. In F. S. Salas E., editor, *Team Cognition: Understanding the Factors that Drive Process and Performance*, pages 133 – 152. American Psychological, 2004.

[4] C. Harris, A. Barnier, and J. Sutton. Consensus collaboration enhances group and individual recall accuracy. *Quarterly journal of experimental psychology*, 65:94–179, 08 2012.

[5] E. Harskamp and N. Ding. Structured collaboration versus individual learning in solving physics problems. *International Journal of Science Education - INT J SCI EDUC*, 28:1669–1688, 11 2006.

[6] C. Holmgård, M. Green, A. Liapis, and J. Togelius. Automated playtesting with procedural personas with evolved heuristics. *IEEE Transactions on Games*, 02 2018.

[7] J. Mathieu, T. Heffner, G. Goodwin, E. Salas, and J. Cannon-Bowers. The influence of shared mental models on team process and performance. *The Journal of applied psychology*, 85 2:273–83, 2000.

[8] L. Mugrai, F. de Mesentier Silva, C. Holmgård, and J. Togelius. Automated playtesting of matching tile games. *CoRR*, abs/1907.06570, 2019.

[9] S. Rahman, I. Endut, N. Faisol, and S. Paydar. The importance of collaboration in construction industry from contractors' perspectives. *Procedia - Social and Behavioral Sciences*, 129, 05 2014.

[10] L. Rose. Interprofessional collaboration in the icu: how to define?*. *Nursing in Critical Care*, 16(1):5–10, 2011.

[11] J.-W. Strijbos, R. Martens, W. Jochems, and N. Broers. The effect of functional roles on group efficiency. *Small Group Research*, 35:195–229, 08 2004.

[12] P. Van den Bossche, W. Gijselaers, M. R.Segers, G. Woltjer, and P. Kirschner. Team learning: Building shared mental models. *Instructional Science*, 39:283–301, 05 2011.

[13] M. Weerdt and B. Clement. Introduction to planning in multiagent systems. *Multiagent and Grid Systems*, 5:345–355, 12 2009.

[14] J. Yen, X. Fan, S. Sun, T. Hanratty, and J. Dumer. Agents with shared mental models for enhancing team decision makings. *Decision Support Systems*, 41(3):634 – 653, 2006. Intelligence and security informatics.

[15] J. Yen, X. Fan, S. Sun, R. Wang, C. Chen, and K. Kamali. Implementing shared mental models for collaborative teamwork.