




# Exploratory testing for platform video games: strategies and lessons learned


Yohan Duarte   [ Federal University of São Carlos | [yohandp@estudante.ufscar.br](mailto:yohandp@estudante.ufscar.br) ]

Henrique Canella Mandelli   [ Federal University of São Carlos | [hcanella@estudante.ufscar.br](mailto:hcanella@estudante.ufscar.br) ]

Vinicius Durelli  [ Federal University of São João del Rei | [durelli@ufsj.edu.br](mailto:durelli@ufsj.edu.br) ]

Paulo Augusto Nardi  [ Federal University of Technology - Paraná | [paulonardi@utfpr.edu.br](mailto:paulonardi@utfpr.edu.br) ]

Andre Takeshi Endo   [ Federal University of São Carlos | [andreendo@ufscar.br](mailto:andreendo@ufscar.br) ]

 Department of Computing, Federal University of São Carlos, Rod. Washington Luís, km 235 - Jardim Guanabara, São Carlos, SP, 13565-905, Brazil.

Received: 23 February 2024 • Accepted: 01 July 2024 • Published: 05 July 2024

**Abstract:** With the growth of the digital games market, the occurrence of bugs in games has a significant impact and generates dissatisfaction among users. Therefore, conducting tests is necessary to avoid these events and ensure the quality of the distributed product. Among the tests that are performed in games, one that is particularly effective in identifying bugs from the user's perspective is exploratory testing, but this is little covered in game testing literature, not providing new testers with guides or paths to be used for specific occasions. This paper reports an experience of applying exploratory testing strategies in 2D and 3D platform games. We selected seven well-known strategies of exploratory testing and conducted a study that involved the definition of a game testing procedure and proper adaptations for the games under test. By applying the game testing procedure, several bugs with low, medium and high severity were uncovered. The lessons learned and routines carried out in this study can be used by new testers in games of the same category, in an attempt to obtain better results.

**Keywords:** Game Testing, Software Testing, Playtesting, Platform Games, Manual Testing

## 1 Introduction

In recent years, the global digital gaming market has shown a trend of steady growth. This multi-billion dollar industry yielded an estimated revenue of approximately \$180.3 billion in 2021 [Chueca *et al.*, 2024], and it is expected to reach \$363.19 billion by 2027 [Clement, 2023]. Throughout the years, this thriving industry has continually adapted to the changing preferences of gamers by introducing a wide array of game genres (e.g., racing, fighting, and adventure games): each decade is marked by the introduction of entirely new subgenres (e.g., souls-like and walking simulators). Among the plethora of genres available, the platform genre is notably recognized as one of the most familiar to gamers. Platform games are characterized by the primary objective of navigating a character from point A to point B, overcoming various obstacles along the way. A quintessential example of this genre is *Mario*, which has amassed approximately \$32.4 billion in revenue since its inception in 1983 [Dias, 2021]. Currently, games of this genre surpass millions of downloads, with *Fall Guys* being a widely recognized title, reaching 11 million downloads on the computer platform within merely four months of its release [Woods, 2020].

As mentioned, the video game industry has attained remarkable levels of profitability. Nonetheless, to sustain this growth trajectory, it is imperative for the industry to offer games that are free from issues, thereby highlighting the significance of delivering a seamless gaming experience [Xavier *et al.*, 2023]. A notable example highlighting the consequences of game-related issues is evident in the case of *Cyberpunk 2077*, which, despite achieving the

record for the most significant digital launch ever [Bankhurst, 2021], with 10.2 million digital copies sold, faced numerous challenges upon release. These were followed by numerous post-launch challenges including a range of issues related to performance and gameplay mechanics. The dissatisfaction voiced by players, coupled with the game's initial adverse reception, resulted in approximately 9% of consumers requesting refunds for their purchases [Demartini, 2021]. As pointed out by CD Projekt Red CEO and co-founder Marcin Iwiński, many of the issues could have been avoided if testing activities had been carried out properly [Politowski *et al.*, 2021a]. The game industry is becoming more complicated and dynamic, as demonstrated by the cautionary case of *Cyberpunk 2077* [Schultz *et al.*, 2005], which emphasizes the pressing need for the broad adoption of systematic testing procedures. It is worth noting, however, that game testing is different from traditional software testing, both in terms of the number of steps involved [Redavid and Farid, 2011; Politowski *et al.*, 2021a,b] and the priorities it emphasizes during execution. Specifically, in hopes of ensuring that the product attains a level of quality considered satisfactory for meeting consumer expectations, game developers typically prioritize the testing of functionalities. Consequently, more technical facets of testing activities are seen as an afterthought [Kasurinen and Smolander, 2014].

Over the course of testing video games, manual tests<sup>1</sup> are utilized to verify user logic (i.e., behaviors and decision-making patterns that players exhibit while interacting with the game) and their adaptations and responses to commands

<sup>1</sup>Manual tests are tests crafted and performed manually by a human tester without the use of automated scripts.

and environments. Manual testing is widely adopted in game development, since test automation is challenging due to code highly-coupled with the user interface, wide state space to explore, and non-determinism [Murphy-Hill *et al.*, 2014]. A more systematic approach to conducting manual testing sessions is *Exploratory Testing* (ET) (Whittaker; Bach; Itkonen). Specifically, ET is focused on conducting a series of undocumented testing sessions to identify issues. In this approach, ET seeks to leverage the expertise and creativity of testers, thereby enabling a more comprehensive examination of the system under test (SUT) [Hendrickson, 2013]. Additionally, the knowledge acquired in previous ET sessions can be leveraged to enhance the exploration in subsequent sessions. Therefore, ET is a goal-focused, streamlined approach to testing that keeps testers engaged throughout the testing process. Given these benefits, we hypothesized that ET can be employed to uncover problems in video games.

This paper presents an experience report on the use of ET to uncover problems in platform digital games. To this end, five games representing the platform category were chosen, and ET sessions were conducted to create a routine for testers to use while testing games of the same category. We conducted an in-depth review of the principles and concepts of ET to better come up with test cases and perform ET sessions properly. Additionally, all ET sessions were recorded to provide more details and allow for future reference and learning. Another contribution derived from our investigation encompasses a set of strategies that can be followed while conducting ET sessions for 2D and 3D platform digital games. We also report on an analysis of the severity and the number of bugs (i.e., faults) encountered upon applying these strategies to the aforementioned platform games.

In this journal version, we augment our initial conference paper [Duarte *et al.*, 2023] in the following ways: (i) our sample has been expanded by including three additional platform games, thereby extending the original corpus from two to five games; (ii) based on the results obtained from the expanded sample and the feedback received during the conference, we have provided a more detailed account of our lessons learned and introduced specific guidelines tailored to each exploratory testing strategy.

The remainder of this paper is organized as follows. Section 2 gives an overview of key concepts essential for comprehending our study, emphasizing definitions related to platform games. Section 3 summarizes related work on ET. Section 4 outlines the study setup we used to probe into the effectiveness of ET at uncovering problems in platform games. Section 5 describes the analysis of the results, Section 6 discusses threats to validity, and lessons learned are presented in Section 7. Section 8 presents concluding remarks and summarizes future work.

## 2 Background

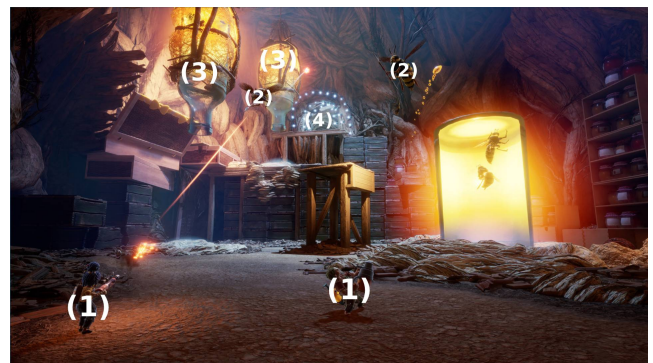
Assuming that games constitute formal, rule-based systems characterized by a diversity of consequences and outcomes that are contingent upon values influenced by player effort [Juil, 2003], it becomes imperative for a game to demonstrate exemplary gameplay to be deemed engaging. Playabil-

ity, as delineated by Al-Azawi *et al.* [2013], encapsulates the challenges encountered by the player (gameplay), the seamless functioning of the game, and the evolution of both character and storyline.

Another aspect related to player engagement is the quality of the game. Aleem *et al.* [2016] posit that only good games can keep players interested, and the quality of a game can be evaluated through its sound design, graphical fidelity, and the robustness of its compiled code.

As previously noted, games are categorized into genres, and it is not uncommon for them to straddle multiple genres when they amalgamate elements from distinct categories. Novak [2017] posits that the definition of a game genre can incorporate a confluence of elements, including theme, setting, interface, perspective, and gameplay strategies. In addition to these elements, gameplay often influences the classification of a title, as the storyline and challenges it presents play a pivotal role in shaping the player's emotional response.

In platform games, the player controls a character who defeats enemies or avoids obstacles [Minkinen, 2016] to reach their goal, which is to go from point A to point B. Figure 1 shows “It Takes Two”, a cooperative platform game, where some of the aforementioned elements can be observed: (1) the characters controlled by the players, (2) the enemies, (3) the obstacles to be navigated, and (4) the end goal.



**Figure 1.** It Takes Two, an example of 3D platform game. Marks (1) are the characters controlled by the players, (2) the enemies, (3) the obstacles to be navigated, and (4) the end goal. Adapted from Electronic Arts Inc. [2021].

During the early arcade era, there existed a direct financial incentive to challenge players extensively, as the difficulty of games often translated into increased revenue from repeated attempts. Although this practice has become less prevalent in the modern gaming landscape, many video games nowadays still feature extensive and challenging levels. In such instances, bugs that prevent level progression or undermine gameplay can be exceedingly frustrating for players. To address this issue, as reported by Politowski *et al.* [2021a], the predominant strategy for identifying and eliminating bugs is manual gameplay testing (i.e., an end-to-end testing process also known as play-testing). Given that automating play-testing sessions is difficult [Lovreto *et al.*, 2018], we surmise that relying on a more systematic approach to manual testing can be more effective in terms of uncovering problems: specifically, we believe that ET can be seamlessly integrated into the testing process of video games, enabling developers

to establish more systematic and repeatable testing sessions. These sessions can emphasize not only the player-centered aspects of the game but also its technical aspects, thereby enhancing the overall quality and playability of the game. Considering our proposed ET-centric approach to testing platform games, the developer's testing effort is now driven by strategies that are employed during ET sessions. Furthermore, since play-testing demands skill and creativity and thus cannot be broken down in a series of repetitive tasks, the ET workflow in this context also enables the capitalization on human expertise during play-testing. This approach ensures that the unique insights and innovative strategies of testers are fully leveraged to enhance the testing process. Bearing this in consideration, we devised an ET-based approach to conduct play-testing sessions from the player's perspective.

### 3 Related Work

Kaner *et al.* [1999] posit that testing is directly related to software quality. The authors also define software testing as the process of seeking errors and assert that, regardless of the planning, time, people, and resources invested, it is not possible to test a software completely.

When tests are performed without the use of scripts, they are classified as ET, where the tester can test the SUT in any way they want and generate documentation while conducting the tests [Whittaker, 2009]. However, the expected results rely heavily on the creativity and knowledge of the tester, which may lead to inefficiencies in the absence of sufficient experience. As defined by Bach [2003], ET is the process of learning, designing, and executing tests simultaneously, hence ET sessions leverage the knowledge acquired from each session to enhance subsequent ones, thereby establishing a continuous improvement cycle (as shown in Figure 2). The author defines specific situations in which ET should be employed, such as quickly learning about the product, providing rapid feedback, conducting tests with vague instructions, relying on the user manual, investigating and isolating faults or issues.

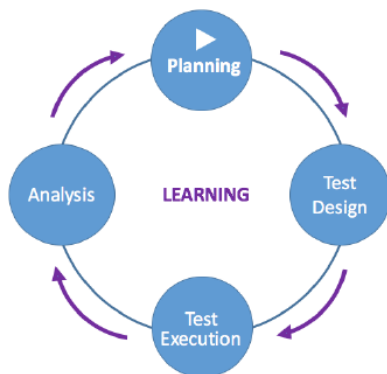


Figure 2. ET cycle [Copche *et al.*, 2021].

According to Iftikhar *et al.* [2015], ET is labor-intensive, lacks scalability, and requires a substantial investment of time and resources. Nevertheless, ET excels when used to validate the user's perspective, which is an important factor in game testing.

The management and control of ET can be achieved through the technique of Session-Based Testing (SBT) [Lynsday and Eeden, 2003]. This technique involves planning, managing, and tracking the progress of tests in sessions lasting up to a maximum of 12 hours, being its composition a charter, time limited sessions, results, and questions, with no requirement to detail the techniques or strategies to be used during the sessions [Itkonen, 2011].

Whittaker [2009] proposes several strategies for efficiently conducting ET sessions. These strategies have inspired numerous authors. In the context of our study, we chose the same seven strategies selected by Micallef *et al.* [2016] in their study:

1. **Tour Bus Strategy:** the tester takes a “tour” of the system, stopping at any feature as desired for a short period of time and then returning to the main route. Similarly to a guided tour in a city, the hallmark of this exploration strategy is stopping at places of interest (i.e., important features).
2. **Exploratory Smoke Testing:** the tester randomly checks if the system's features are functioning properly, without following a pattern or rules.
3. **Crime Spree Tour:** the tester focuses on a specific feature or neighborhood with the intention of breaking it (i.e., uncovering problems). For example, the tester might try to break a number range field, filling it out with letters or large numbers.
4. **Garbage Collectors Tour:** the tester selects an objective, finds the fastest way to accomplish it, performs tests, and moves on to the next objective. Similarly to a garbage collector, traveling between neighborhoods and stopping at each house.
5. **Back Alley Tour** - The tester focuses on exploring the less frequently used features of the system. It is the opposite of a guided tour, where popular places would be visited.
6. **User Interface Exploration:** the tester learns by exploring the user interface, understanding the functionalities of different parts of the interface, and testing their behaviors.
7. **Bad Neighborhood Tour:** the tester examines the “neighboring” features of where a bug was found, aiming to uncover other problems. This strategy is founded upon the analogy drawn from a neighborhood that tourists should avoid due to its proximity to a troubled area.

Manual testing and, in particular ET, has an important role in Game Software Engineering. Kasurinen and Smolander [2014] interviewed 27 professionals from seven game development companies and observed that they chose for exploratory and usability testing during the testing phase, focusing on internal mechanics, balance rules, and user experience. The authors also affirm that these companies have resources to conduct technical tests, but they prioritize the improvement of the mentioned aspects.

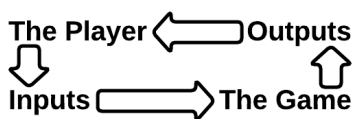
In another study, Politowski *et al.* [2021b] conducted an analysis of 200 post-mortems<sup>2</sup> and listed the top 10 issues in

<sup>2</sup>A post-mortem is a document that presents an analysis of successes,

the game industry, suggesting testing as a solution for those related to fun, complexity, and design vision. The authors assert that in game testing, the tester has the role of evaluating gameplay mechanics, identifying and reproducing issues. They also state that the techniques used for game testing are mainly manual, allowing the tester to learn about the game while performing the tests.

Schultz *et al.* [2005] state that almost all tests performed on games are black-box tests, where the testing is conducted without access or knowledge of the source code. In such cases, tests are executed based on inputs provided by the tester and the corresponding outputs received, creating a loop, as depicted in Figure 3. The authors also define an important structure that should be followed when conducting tests in games:

1. *Pre-production*: the testing planning document begins.
2. *Kickoff*: it addresses critical issues related to the software, serving to improve quality and test execution.
3. *Alpha*: testing is initiated, and all game modules should be tested. However, at this stage, the game is not yet fully finalized, so tests are conducted in a limited way.
4. *Beta*: testers can play without the restrictions imposed in the *Alpha* phase, and the tests conducted prioritize game refinement, addressing bugs that were not identified in the previous stages and new bugs discovered.
5. *Gold*: all test suites are executed again on what is considered the final version of the game, delaying the release in case any bugs are found.
6. *Release*: the released game is tested, and any bugs uncovered at this stage are discussed and, if necessary, slated for fixing.
7. *Post-Release*: the identified bugs are resolved through patches or updates. However, when this occurs, the entire bug list must be reviewed, and the changes should be tested for compatibility with the game and previous updates.



**Figure 3.** Player's input and game output loop.  
Adapted from Schultz *et al.* [2005].

While manual testing is still the *de facto* approach for gameplay testing, automated solutions have been proposed in the literature. Iftikhar *et al.* [2015] propose an automated testing strategy for games where a state machine is used to generate and execute tests automatically. The authors acknowledge that there is an investment of time and resources in developing these test models. However, they assert that once ready, the generation and execution of automated tests save the effort required by manual testing.

Artificial Intelligence (AI) methods have also been applied in test automation. For instance, Sriram [2019] developed an artificial intelligence to perform automated testing in 2D

platform games. Among the limitations encountered by the author, it is stated that the created agent does not follow paths similar to those of a human tester, presenting the lack of a human perspective in the testing process. The author also mentions the limitation of the number of mechanics to be tested by the agent, as it was unable to keep up with the variety of mechanics presented by 2D platform games.

So far, we did not find any study related to manual and exploratory testing of video games in the Journal on Interactive Systems.

## 4 Study Setup

ET has been widely used in practice and has a vast literature about its adoption in traditional software, yet little is known about its applicability to digital games. To shed some light on this subject, we undertook a study to evaluate the applicability of ET in the context of digital games. Thus, we set out to answer the following research question (RQ):

- *How applicable are the ET strategies to digital games?*

To address this RQ, we investigated two key aspects: (i) how a subset of established strategies for ET can be applied, and (ii) whether or not they are capable of helping game testers to uncover bugs. Additionally, we share the lessons learned that have the potential to enhance the future integration of ET in digital games.

### 4.1 Games Under Test

We chose to explore platform games first to establish a foundational understanding of ET strategies within an important game genre. We conducted our study with both 2D and 3D platform games. Despite belonging to the same genre, 2D and 3D games can differ greatly, prompting us to investigate whether any adaptations are necessary. It is important to note that considering other genres would require reevaluating and adjusting the strategies used in this study. Generalizing the test procedure to other genres is a subject for future studies.

We selected five games for this study: *Little Spy* – v1.0.6 (*Game 1*), released in 2021; *Diver Down* – v1.2 (*Game 2*), released in 2018; *Tiny Crate* – v11.2022 (*Game 3*), released in 2021; *Portal* – v12.2014 (*Game 4*), released in 2007; and its sequel *Portal 2* – v01.2022 (*Game 5*), released in 2011. They are briefly described next.

Game 1, Game 2, and Game 3 belong to the 2D platform genre and are available in the `itch.io`<sup>3</sup> platform. These indie games are available as open source projects at GitHub and have been developed using the Godot open source engine [Godot, 2024].

In Game 1, players control a spy air-dropped into enemy territory to collect as many intelligence items as possible; then, they should return to the helicopter within a specified time frame [Winpress, 2024]. Figure 4 depicts a level from Game 1 where the player (1) navigates platforms, collects items (2), avoids enemies (3), and reaches the helicopter (4).

failures and lessons learned in a project.

<sup>3</sup><https://itch.io>



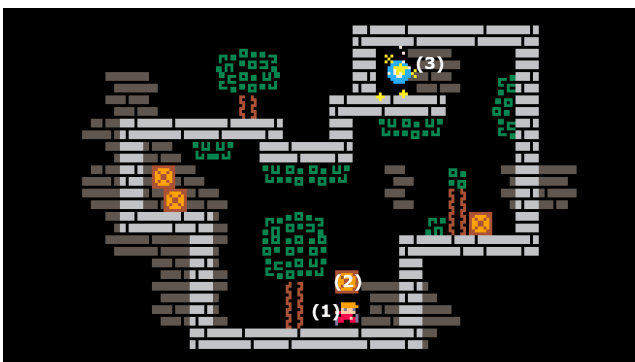
**Figure 4.** Level example of Game 1. Mark (1) is the player, (2) an item, (3) an enemy, and (4) the helicopter.

Game 2 allows players to dive onto solids and use this ability to progress through stages, while avoiding lights [Escada Games, 2024]. Figure 5 illustrates a level of Game 2, where the player (1) can dive onto the solid surface in front of him (2), needs to avoid the lights (3), and proceeds to the door (4).



**Figure 5.** Level example of Game 2. Mark (1) is the player, (2) a solid surface, (3) the lights, and (4) the exit door.

In Game 3, players must lift and toss crates to create platforms, enabling them to reach higher ground [Harmony Honey, 2024]. Figure 6 shows a Game 3 level where the player (1) is lifting a crate (2), those crate can be moved around and help him to reach the end of the level (3).

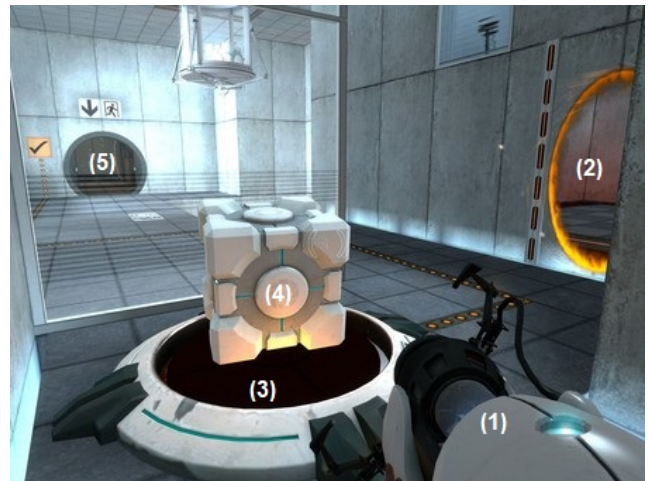


**Figure 6.** Level example of Game 3. Mark (1) is the player, (2) a crate, and (3) the end of the level.

Game 4 and Game 5, developed by the video game developer and publisher Valve Corporation, fit into the 3D platform genre and incorporate elements from diverse genres including action, adventure, and puzzle-solving [Valve Corporation, 2023b,a]. In these games, the player is a test subject who must progress through levels known as test chambers,

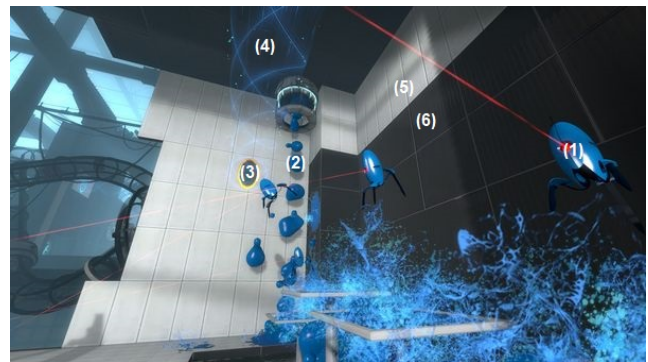
either by pressing buttons or avoiding contact with enemies. To accomplish this, the player is provided with a portal gun capable of shooting two interconnected portals. Upon entering one portal, the player or any object emerges from the other portal.

There are graphical and gameplay differences between Game 4 and Game 5, primarily due to the time frame between their releases. Figure 7 depicts a level from Game 4, highlighting some key elements: (1) the portal gun, (2) a portal created by the player, (3) the button that needs to be pressed to complete the level, (4) the object used by the player to press the button and move on, and (5) the door indicating the level exit.



**Figure 7.** Level example of Game 4. Mark (1) is the portal gun, (2) a portal created by the player, (3) a button, (4) an object used to press the button, and (5) a door opened by the button. Adapted from Valve Corporation [2023b].

Figure 8 shows a level from Game 5, where other elements are presented: (1) an enemy that shoots when the player is spotted, (2) a new liquid mechanic that causes different effects on surfaces and objects it touches, (3) one of the portals placed by the player, (4) a new mechanic of a “gravity tube” that moves the player or objects in the indicated direction, (5) a surface that allows the player to place a portal, and (6) a surface that does accept portals.



**Figure 8.** Level example of Game 5. Mark (1) is an enemy, (2) a new liquid mechanic, (3) a portal placed by the player, (4) a new mechanic of a “gravity tube”, (5) a surface that allows the player to place a portal, and (6) a surface that does accept portals. Adapted from Valve Corporation [2023a].

We selected these games for several reasons. In settling for Games 1, 2 and 3, we aimed to select 2D platform games

crafted by indie developers or studios, specifically those available as open source projects. These games are characterized by their non-trivial nature and showcase a diverse set of mechanics. The level progression increases in difficulty as the player advances, contributing to their complexity. Our search for games meeting these criteria lead us to itch.io. We targeted indie games because our study about ET strategies would particularly help them. Finally, opting for open source games enhances the potential for future studies, as we have more control and access to the underlying code.

For Games 4 and 5, our aim was to include complex 3D games, well-known in their genre and in the game market, and developed by a reputable company. We considered complex games the non-trivial ones, where there are many game elements and mechanics, lengthy phases, well-designed large maps, interactions with enemies or non-player characters (NPCs), soundtrack, and meticulously polished graphics. Testing this kind of games poses some challenges, and we anticipated they would yield valuable insights and lessons. They are also mature games, released some years ago, and have received various updates.

We envisaged that positive results for the selected games would motivate further studies about applying ET to digital games. Conversely, negative results would show major limitations of ET strategies, given the widespread availability and popularity of 2D/3D platform games among players.

## 4.2 Game Testing Procedure

This section outlines the procedure elaborated to apply ET to the five selected games.

**Single Session Gameplay:** Initially, the game under test was played through in a single test session without time constraints or predefined strategies. To accommodate the potential for an extended duration, we allowed interruptions so that the game was saved and resumed after a break. This session allowed for the tester to grasp the mechanics of the game, easing the preparation for subsequent test sessions and providing unrestricted access to all game levels. Despite the absence of predetermined timeframes and strategies in this session, the tester actively sought out and documented bugs as they were encountered. This test session bears some resemblance to ad hoc tests performed by play-testers. This particular test session was termed *Single Session Gameplay*.

**Application of ET strategies:** Due to time constraints and potential adaptations required, we decided to employ a subset of ET strategies proposed by Whittaker [2009]. In a prior empirical study on ET by Micallef *et al.* [2016], seven of the Whittaker’s strategies were selected and analyzed; we found these seven strategies suitable for game testing as well. Consequently, we used these strategies used in our study: Tour Bus Strategy, Exploratory Smoke Testing, Crime Spree Tour, Garbage Collectors Tour, Back Alley Tour, User Interface Exploration, and Bad Neighborhood Tour (see Section 3). We then examined the strategies and how they may be applied to the game, with a focus on assessing the quantity and duration of test sessions for each strategy. Additionally, we

undertook adaptations in the strategies for the game’s context.

In Tour Bus Strategy, the “main route” refers to completing the game level, while objects, map elements, or alternative options serve as “features”. In Back Alley Tour, alternative routes for level completion were categorized as “rarely used elements”, along with specific features like console or less displayed options to players. Strategies Exploratory Smoke Testing, Crime Spree Tour, and Garbage Collectors Tour were employed following their definitions without any adaptations.

For these five strategies, namely Tour Bus Strategy, Exploratory Smoke Testing, Back Alley Tour, Crime Spree Tour, and Garbage Collectors Tour, we defined an uninterrupted test session lasting  $x$  minutes, focusing on exploring a single game level. The timespan of  $x$  minutes was determined based on the experience in the Single Session Gameplay. Subsequently, we randomly selected six levels of the game and applied each strategy to these levels within the specified timespan. Due to the limited number and brevity of levels in Game 1, we considered all its levels as a single one. Table 1 summarizes the settings defined for the five games.

**Table 1.** Settings for the five ET strategies (i.e., Tour Bus, Exploratory Smoke, Back Alley, Crime Spree and Garbage Collectors.).

Game	Session Time	Selected Levels
Game 1	5min	All 6 levels in one session
Game 2	3 min	Level 1, Level 4, Level 8, Level 14, Level 18, Level 19
Game 3	7 min	Level 3, Level 12, Level 18, Level 23, Level 24, Level 27
Game 4	10 min	Chambers 4-7, Chambers 10-12, Chamber 13, Chamber 14, Chamber 17, Chamber 19
Game 5	10 min	Chapter 2, Chapter 3, Chapter 4, Chapter 6, Chapter 7, Chapter 8

For the User Interface Exploration strategy, we chose to focus on Graphical User Interface (GUI) elements akin to those found in conventional software, such as menus and input fields. This strategy, however, was not applied to Games 1 and 2 due to the absence of such GUI elements in these games. In the case of Game 3, we defined a 10-minute test session, whereas 15-minute test sessions were utilized for Games 4 and 5.

The last strategy applied was Bad Neighborhood Tour. Each bug detected so far in the game was examined, and the tester analyzed whether there is a “bad neighborhood” around it to be explored or not. To do so, the tester assessed the surrounding area and deliberated on potential issues related to the bug. For each bug exhibiting a potential “bad neighborhood”, the tester conducted a test session, allowing sufficient time for exploration without a time limit.

**Operational details:** During the test sessions, the testers were instructed to avoid performing actions that could lead to previously-detected bugs; this guideline aimed to save session time to test other features of the game. The test sessions for Games 1-3 were performed in a computer AMD Ryzen 7 5800x 3.8GHz 16GB RAM AMD Radeon RX 6600 8GB

with Windows 11, while Games 4 and 5 were executed in a computer AMD Ryzen 5 1600 3.2GHz 16GB RAM Nvidia GTX 1070 6GB with Windows 11. We opted for two authors to participate as testers so that we could gauge deeper insights about the intrinsic steps of using ET in games. It is worth noting that, prior to this study, neither of them had played any of the five games under test. Both authors have experience with software engineering and testing, and are well-versed in playing games. All test sessions were captured via screen recording using OBS Studio [OBS Project, 2022], encompassing video and audio of the testers. We used the recordings for further analyses, including test session evidence, checking potential issues, and confirming bugs.

When a bug was discovered during the tests, the tester logged it in a spreadsheet and correlated it with the session recording. Each bug is assigned an ID, and in the case of recurring bugs, the ID from the first occurrence was also used in subsequent instances. All bugs were classified based on their severity, determined by the extent to which they bother or disrupt the player's experience. If the bug is minor and would be likely overlooked by the player, we classified it as *low* severity. If the bug pertains to rendering issues or malfunctioning of game elements, we classified it as *medium* severity. In instances where a bug significantly disrupts the player immersion, violating the game's rules, leading them to get stuck or hindering the progression in the game, we classified it as *high* severity.

## 5 Analysis of Results

This section analyzes the results obtained from the 152 test sessions (totaling more than 25 hours), in which the ET strategies were applied to the five games selected. Overall, 111 bugs were reported, being 105 unique bugs (some bugs were found twice). The main findings and the answer to the RQ are highlighted as framed rectangles throughout the section.

Table 2 gives an overview of the main results. For each ET strategy, it shows per game the number of test sessions, the time invested, and the number of bugs reported. Game 1 had 8 test sessions totalling around 50 minutes, and 10 bugs were reported. Game 2 had 33 sessions which took almost two hours, and 14 bugs were identified (12 bugs are unique). Game 3 had 34 sessions which took approximately four hours and a half, and 10 bugs were reported (8 bugs are unique). Game 4 had 40 test sessions totalling almost eight hours, and 42 bugs were reported (41 bugs are unique). Game 5 had 37 test sessions which took around ten hours and a half, and 35 bugs were reported (34 bugs are unique).

Initially, the Single Session Gameplay (row 3 in Table 2) helped to find 12 bugs (2 in Game 1, 2 in Game 2, 1 in Game 3, 2 in Game 4, and 5 in Game 5), over an approximate period of 7 hours and 35 minutes. As seen in Table 2, the duration of each session progressively increases from Game 1 (0:09:17) to Game 5 (4:36:47). Notably, the Game 5 session took more than twice the time invested in Game 4, due to Game 5's greater length and complexity. This part helped to comprehend the game mechanics and unlock access to all game levels.

Concerning Tour Bus Strategy, Exploratory Smoke Test-

ing, Crime Spree Tour, Garbage Collectors Tour, and Back Alley Tour (shown in rows 4-8 of Table 2), each one of them was applied, in variable time sessions, to six game levels, except for Game 1 in which one level was considered (see Table 1). In Game 1, each of these strategies was applied in one session with around five minutes. For Games 2-5, each strategy was performed in six test sessions, and the total elapsed time was from approximately 18 minutes for Game 2 to around one hour for Games 4 and 5, which can be seen in Table 2. The five strategies were applied directly or with minor adaptations (see subsection 4.2) and collectively helped to uncover bugs across all games: 6 bugs in Game 1, 10 bugs in Game 2, 7 bugs in Game 3, 28 bugs in Game 4, and 25 bugs in Game 5.

*Finding 1:* Five out of the seven ET strategies could be applied to test the games with minor adaptations.

The User Interface Exploration strategy was applied in three games (row 9 of Table 2): a 10-minute session for Game 3 and two 15-minute sessions for Games 4 and 5. Specifically, these test sessions targeted UI elements that are close to traditional software. We found two bugs in Game 3, eight bugs in Game 4, and four bugs in Game 5.

*Finding 2:* For games that have static menus, strategy User Interface Exploration was set up within only one test session and helped to reveal bugs related to traditional UI elements.

Finally, all bugs found so far (8 bugs in Game 1, 12 bugs in Game 2, 10 bugs in Game 3, 38 bugs in Game 4, and 34 bugs in Game 5) were evaluated for potential "bad neighborhood". For Game 1, two sessions with no time limit were conducted to explore the neighborhood of two bugs. This same configuration was replicated for Games 2 and 3. For Game 4, eight sessions without a time limit were carried out to investigate the surroundings of eight bugs, and for Game 5, five sessions (involving four bugs) were conducted. Applying the Bad Neighborhood Tour strategy (row 10 of Table 2) in Game 1 for 14 minutes supported the discovery of two new bugs; two bugs in 7 minutes for Game 2; for a total of almost 43 minutes helped to reveal four new bugs in Game 4; and in around half an hour, one new bug in Game 5. The strategy did not help to uncover new bugs in Game 3.

*Finding 3:* Strategy 'Bad Neighborhood Tour' was run at the end using the bugs previously uncovered; it required an assessment step to reason about the existence of a potential bad neighborhood. It helped to reveal new bugs in four out of five games and clarify doubts related to the previously found bugs.

Overall, Exploratory Smoke Testing resulted in the highest number of bugs for Games 1, 3 and 4, while Tour Bus Strategy helped to find the highest number of bugs for Game 2, and Garbage Collectors Tour for Game 5. In Game 1, Single

**Table 2.** Overview of the main results.

Strategy	Game 1			Game 2			Game 3			Game 4			Game 5		
	Sessions	Time	Bugs	Sessions	Time	Bugs	Sessions	Time	Bugs	Sessions	Time	Bugs	Sessions	Time	Bugs
Single Session Gameplay	1	0:09:17	2	1	0:16:42	2	1	0:42:01	1	1	1:50:33	2	1	4:36:47	5
Tour Bus Strategy	1	0:06:37	1	6	0:18:19	6	6	0:42:00	2	6	1:02:08	2	6	1:02:04	5
Exploratory Smoke Testing	1	0:05:10	2	6	0:18:45	2	6	0:42:57	3	6	1:01:24	10	6	1:00:38	4
Crime Spree Tour	1	0:04:57	1	6	0:18:28	0	6	0:42:50	0	6	1:02:09	7	6	1:02:59	2
Garbage Collectors Tour	1	0:04:45	0	6	0:18:08	0	6	0:42:45	1	6	0:59:29	5	6	1:01:07	9
Back Alley Tour	1	0:04:59	2	6	0:17:31	2	6	0:42:00	1	6	0:59:44	4	6	1:03:34	5
User Interface Exploration	-	-	-	-	-	-	1	0:10:06	2	1	0:15:02	8	1	0:14:08	4
Bad Neighborhood Tour	2	0:14:16	2	2	0:07:10	2	2	0:07:45	0	8	0:42:50	4	5	0:27:42	1
<b>Total</b>	<b>8</b>	<b>0:50:01</b>	<b>10</b>	<b>33</b>	<b>1:55:03</b>	<b>14</b>	<b>34</b>	<b>4:32:24</b>	<b>10</b>	<b>40</b>	<b>7:53:19</b>	<b>42</b>	<b>37</b>	<b>10:28:59</b>	<b>35</b>

Session Gameplay, Back Alley Tour, and Bad Neighborhood Tour are also among the top strategies. Notice that this analysis does not take into account the severity of bugs, or the time spent in the sessions (pace); we revisit the results from these perspectives next.

During the test sessions, bugs with low, medium and high severity were found. In Game 4, a bug with low severity is, e.g., when the player's death is not properly notified. An example of medium severity bug is the lack of a return-to-main-menu option in the game. An example of high severity bug is when interacting with the required object to complete a task can make it to be stuck, preventing the player from progressing further in the level.

Table 3 presents the number of bugs per severity detected, for each strategy. Proportionally, there are more bugs with medium severity: 1 bug for Game 1, 7 bugs for Game 2, 3 bugs for Game 3, 24 bugs for Game 4, and 19 bugs for Game 5. Next are the bugs with high severity (Game 1: 6, Game 2: 5, Game 3: 3, Game 4: 10, Game 5: 14), and the fewest bugs detected were the ones classified with low severity (Game 1: 3, Game 2: 2, Game 3: 4, Game 4: 8, Game 5: 2).

Concerning Game 1, Back Alley Tour found the highest number of high severity bugs, and Single Session Gameplay helped to uncover the only bug with medium severity. As for Game 2, Bad Neighborhood Tour and Tour Bus Strategy uncovered more high severity bugs, Tour Bus also revealed more medium severity bugs, and Back Alley Tour helped to discover the only two bugs with low severity. In Game 3, the three high severity bugs were uncovered by three different strategies: Tour Bus Strategy, Exploratory Smoke Testing, and Back Alley Tour; two out of three medium severity bugs were identified using User Interface Exploration, and Exploratory Smoke Testing resulted in the highest number of low severity bugs. In Game 4, the strategies that resulted in the highest number of bugs of low, medium, and high severity were, User Interface Exploration, Exploratory Smoke Testing, and Garbage Collectors Tour, respectively. For Game 5, only Crime Spree Tour and Garbage Collectors Tour helped to uncover low severity bugs, while Single Session Gameplay, Tour Bus Strategy, and Garbage Collectors Tour resulted in the highest number of medium severity bugs; Garbage Collectors Tour helped to uncover more bugs with high severity.

*Finding 4:* All strategies helped to uncover high severity bugs in at least two games, being Tour Bus Strategy, Exploratory Smoke Testing, and Back Alley Tour capa-

ble of uncovering bugs in four of the five games. Regarding medium severity, Single Session Gameplay uncovered bugs in all of the games, while User Interface Exploration found bugs in the three games considered. For low severity, Exploratory Smoke Testing helped to identify bugs in three games.

Among the applied ET strategies, Tour Bus Strategy, Garbage Collectors Tour, and Back Alley Tour are the top strategy for more high severity bugs in two games.

From another perspective, we analyzed the performance of each strategy by relating the number of bugs and the time invested. Table 4 presents, per strategy and per game, the ratio of bugs found per minute; it also separates the ratio per bug severity. In general (column Total), Single Session Gameplay underperformed in Games 3, 4 and 5, detecting around 0.02 bug per minute; Garbage Collector Tour had the worst results in Games 1 and 2. On the other hand, User Interface Exploration had the best performance when applicable, with 0.198 bugs/minute in Game 3, 0.532 in Game 4, and 0.282 in Game 5; Back Alley Tour was the best in Game 1 (0.400 bugs/minute), and Tour Bus Strategy in Game 2 (0.328).

*Finding 5:* When applicable, User Interface Exploration helped to find more bugs using less session time, even for medium and high severity bugs.

Notice that User Interface Exploration focuses on traditional UI elements and, as a consequence, targets a specific class of bugs. So, we now take a look at the other strategies that explore the gameplay. In Game 1, Exploratory Smoke Testing and Back Alley Tour had the two best results overall (column Total), but only Back Alley had by far the best performance in high severity bugs. In Game 2, Tour Bus Strategy had the best overall result, but Bad Neighborhood Tour had the best performance in high severity bugs. In Game 3, Exploratory Smoke Testing had the best overall best results, and no strategy stood out for high severity bugs. In Game 4, Exploratory Smoke Testing and Crime Spree Tour had the overall best results (column Total), but were outperformed by Garbage Collectors Tour in high severity bugs. In Game 5, Garbage Collectors Tour and Tour Bus Strategy had the overall best results (column Total); Garbage Collectors Tour also had the best performance in high severity bugs.



Table 3. Bug severity by strategy.

Strategy	Game 1			Game 2			Game 3			Game 4			Game 5		
	Low	Medium	High	Low	Medium	High	Low	Medium	High	Low	Medium	High	Low	Medium	High
Single Session Gameplay	0	1	1	0	2	0	0	1	0	1	1	0	0	4	1
Tour Bus Strategy	1	0	0	0	4	2	1	0	1	0	2	0	4	1	1
Exploratory Smoke Testing	1	0	1	0	1	1	2	0	1	1	9	0	2	2	2
Crime Spree Tour	0	0	1	0	0	0	0	0	0	1	4	2	1	0	1
Garbage Collectors Tour	0	0	0	0	0	0	1	0	0	0	2	3	1	4	4
Back Alley Tour	0	0	2	2	0	0	0	0	1	1	2	1	0	3	2
User Interface Exploration	-	-	-	-	-	-	0	2	0	4	2	2	0	2	2
Bad Neighborhood Tour	1	0	1	0	0	2	0	0	0	0	4	0	0	0	1
<b>Total</b>	<b>3</b>	<b>1</b>	<b>6</b>	<b>2</b>	<b>7</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>3</b>	<b>8</b>	<b>24</b>	<b>10</b>	<b>2</b>	<b>19</b>	<b>14</b>

Table 4. Ratio of the number of detected bugs per minute.

Strategy	Game 1				Game 2				Game 3				Game 4				Game 5			
	Low	Medium	High	Total	Low	Medium	High	Total	Low	Medium	High	Total	Low	Medium	High	Total	Low	Medium	High	Total
Single Session Gameplay	0.000	0.108	0.108	0.216	0.000	0.120	0.000	0.120	0.000	0.024	0.000	0.024	0.009	0.009	0.000	0.018	0.000	0.014	0.004	0.018
Tour Bus Strategy	0.151	0.000	0.000	0.151	0.000	0.219	0.109	0.328	0.024	0.000	0.024	0.048	0.000	0.000	0.032	0.032	0.000	0.064	0.016	0.080
Exploratory Smoke Testing	0.193	0.000	0.193	0.386	0.000	0.053	0.053	0.106	0.047	0.000	0.023	0.070	0.016	0.147	0.000	0.163	0.000	0.033	0.033	0.066
Crime Spree Tour	0.000	0.000	0.202	0.202	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.016	0.065	0.032	0.113	0.016	0.000	0.016	0.032
Garbage Collectors Tour	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.023	0.000	0.000	0.023	0.000	0.034	0.050	0.084	0.017	0.065	0.065	0.147
Back Alley Tour	0.000	0.000	0.400	0.400	0.114	0.000	0.000	0.114	0.000	0.000	0.024	0.024	0.017	0.033	0.017	0.067	0.000	0.047	0.032	0.079
User Interface Exploration	-	-	-	-	-	-	-	-	0.000	0.198	0.000	0.198	0.266	0.133	0.133	0.532	0.000	0.141	0.141	0.282
Bad Neighborhood Tour	0.070	0.000	0.070	0.140	0.000	0.000	0.280	0.280	0.000	0.000	0.000	0.000	0.000	0.093	0.000	0.093	0.000	0.000	0.036	0.036
<b>Total</b>				<b>0.200</b>				<b>0.122</b>					<b>0.037</b>			<b>0.089</b>				<b>0.056</b>

*Finding 6:* For strategies that explore the gameplay, Back Alley Tour helped to uncover more bugs per minute in Game 1, Tour Bus in Game 2, Exploratory Smoke Testing in Games 3 and 4, and Garbage Collectors Tour in Game 5. For high severity bugs, Tour Bus Strategy, Garbage Collectors Tour, and Back Alley Tour had the best performance in two games.

Although some ET strategies seem to shine in specific perspectives or games under test, our impressions are that all strategies contributed in some way for a more holistic play-testing. For instance, Single Session Gameplay had the longest test sessions and helped to find a low number of bugs per minute. Nevertheless, it had a pivotal role to structure the testing procedure and support the acquisition of knowledge required to apply other strategies. Another example is User Interface Exploration that has the best ratio of bugs per minute but targets a specific kind of bugs. So, other strategies are important to explore the gameplay.

*Answer to RQ:* ET strategies can be applied to digital games with proper planning, knowledgeable methodology decisions, and minor adaptations. Overall, all strategies helped the tester to uncover several bugs in three indie and open source 2D platform games and two industrial 3D platform games.

## 6 Threats to Validity

There exist several 2D and 3D platform games with different mechanics, making it impossible to encompass all these mechanics by testing only a few games. Therefore, we selected five games: three are open source indie 2D games, while two are closed source 3D games. The last two games are well known titles from the same company, and were selected due to their market recognition and being part of the same franchise. We believe that the way we organize the sessions and

strategies can be applied to other games in the same category, though further studies with different games are required to provide more evidence about ET in this domain.

To achieve a better understanding and insights about applying ET to digital games, two authors performed the role of tester, potentially introducing bias. This experience will help to set up the next steps of the research, which involves customization of ET strategies and other empirical studies. Among them, it is of interest to evaluate how these ET strategies benefit testers with different backgrounds and levels of experience, as well as to scale up the study to include more participants.

Our results suggest that ET strategies can be successfully applied, helping testers to uncover several bugs. As our study dealt with a limited context, the results herein presented cannot be generalized. In the future, it is possible to replicate this study with a larger number of testers and different games, which may require adjustments to session duration and the number of sessions conducted. As this implies in a great variation of the context and scope, we surmise that different results may be obtained.

## 7 Lessons Learned

Frequent motion sickness was noted during the tests of Game 4, necessitating breaks between sessions for the tester's recovery. Upon researching forums and communities, these symptoms seemed to be widespread among players of Games 4 and 5, but more prevalent in Game 4. Suggestions from these forums indicated that modifying certain settings could help alleviate the issue, such as disabling the "Motion Blur" and "Vertical Synchronization" options, and adjusting the "Field of View" setting. This experience underscores that the test sessions may support the identification of other issues related to the game, besides bugs. In a pre-release stage, early feedback on potential motion sickness could alert the developers to take some preventive measures.

While our study focused on already tested, and released games, the testing procedure is anticipated to provide the best return on investment in pre-release stages. On the other hand,

if the proposed procedure helped to uncover several bugs in such a scenario, it is expected to deliver promising outcomes for games that have not undergone extensive testing.

Figure 9 shows a diagram that summarizes the steps we performed; its goal is to assist game testers that want to apply the testing procedure herein described. The diagram presents the following steps: (1) the tester performs a Single Session Gameplay, in which he completes the game with the objective of understanding the game mechanics and reason about appropriate ways to organize the test sessions; (2) any bugs encountered during this session should be recorded; (3) test sessions are defined considering the ET strategies' particularities; (4) the test sessions are carried out; (5) any bugs revealed are once again recorded; (6) the bugs found are assessed with respect to the applicability of the Bad Neighborhood Strategy; (7) new test sessions are defined for applicable bugs; (8) the sessions are conducted; and (9) the bugs uncovered are once again recorded.

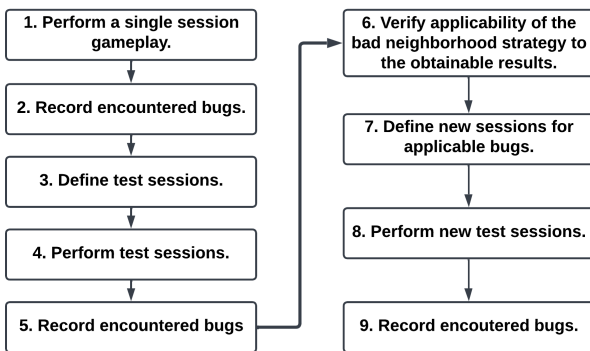


Figure 9. Steps for conducting the ET testing procedure.

We suggest recording and narrating the test sessions to document when bugs are revealed and to clarify any cases that are unclear at first sight. For example, a reported bug may be an inconsistent game behavior that was intentionally introduced (an easter egg). Notice that our testing procedure considered the characteristics of the game, as well as the resources available. So, before commencing the tests, it is important to plan properly, make informed decisions, and perform some adaptations.

As for the tester profile, having knowledge of other games and bugs could help the tester in determining what to search for, expect and define during each test session. In this study, the two testers are well-versed in playing games and have software engineering and testing experience; we believe that this profile helped in time spent learning the game and also in test preparation and execution. Some games do not have an easy way to navigate between levels, which can require the tester to (re)play the game every time he changes strategy. In practice, the game developer should provide means for testers to have free access to levels, set up items, and even remove some game rule; e.g., remove or increase the time limit of a level so that the tester can explore it more freely.

The application of each ET strategy yielded valuable insights that we deemed important for game testers when reproducing it. These insights are described in the following subsections.

**Single Session Gameplay.** In this part, the tester's pri-

mary focus is to obtain a general understanding about the game mechanics, controls, character abilities, scoring system, enemies, and other features. Tutorials and help menus may also aid in the process. To better allocate time and take advantage of other strategies, the exploration cannot divert from the main path as progressing to the game's end. While finding bugs is beneficial, the tester should avoid extensively exploring specific locations, as this task can be undertaken using other strategies. Finally, we recommend the tester takes breaks during extended gaming sessions.

**User Interface Exploration.** For the tested games, we opted for a single test session applying this strategy as the menus and traditional UIs are mostly static. Nevertheless, it may differ for other games, as the number of menus and other UI elements (like inputs, buttons, comboboxes, etc) varies. It is also important to consider if the game state has some influence on the UIs available to the players. Recommended features for this testing strategy also include graphical settings, keybinds, sounds, consoles, heads-up displays (HUD), and status bars.

**Garbage Collectors Tour.** This strategy may draw some inspiration from speedrunning, a gaming practice where the player aims to achieve a specific goal in the game as quickly as possible<sup>4</sup>. The goals extend beyond merely beating a level or completing the entire game. For this strategy, the tester should focus on her objective disregarding other parts of the game, in a rapid way. By reasoning about means of being faster, the exploration may guide the discovery of unintended behavior or bugs.

**Tour Bus Strategy.** This strategy shares some similarities with completionist<sup>5</sup>, a gaming practice where the player intends to complete the entire game while exploring it comprehensively by, for example, obtaining all achievements, or doing all quests. Such a perspective may help to guide the tour to stop not only in the main places of interest, but also the secondary ones.

**Back Alley Tour.** In this strategy, the tester should focus on less frequently utilized features from the player's standpoint. Some examples are hidden elements, developer options, and consoles. The completionist mindset may also offer some insights, as the pursuit of game completion could reveal concealed behavior to be tested.

**Crime Spree Tour.** This strategy aims to explore negative scenarios around specific game features. The tester needs to reason about potentially error-prone features and target them to break the normal behavior of the game. For instance, the tester could try to apply mechanics out of their established context or bounds, or push the limits of certain features.

**Exploratory Smoke Testing.** This strategy should not follow any guideline besides randomly explore features of the game. In particular, we used it to answer "what if" questions about game features; those questions came mostly from observations in previous test sessions.

**Bad Neighborhood Tour.** The neighborhood of a bug may be the home of other bugs, this is an opportunity for the game testers. For instance, if a bug regarding a specific window size of a menu option is found, other menu options

<sup>4</sup><https://www.speedrun.com/support/learn/what-is-speedrunning>

<sup>5</sup><https://ktswblog.net/2022/10/21/being-a-completionist-gamer>

could be explored to verify whether this bug could affect the game in any other way. The concept of neighborhood is also extended: elements of a bug can manifest or “teleport” to different levels or contexts within the game. The tester should explore if a faulty behavior can be partially replicated in different states of the game. It is important, however, to evaluate the bugs beforehand and assess if there is a logical neighborhood to explore. Occasionally, this exploration has already been conducted in previous test sessions.

We have some insights about the ordering of ET strategies in games. By maintaining the Single Session Gameplay as the initial step, the tester could consider to apply User Interface Exploration either before or after implementing the other five strategies. Then, Bad Neighborhood Tour is employed after all other sessions have been completed.

Concerning the five strategies, we suggest beginning with the Tour Bus, where the tester briefly explores various features without delving deeply into any one. Subsequently, Crime Spree Tour would be adopted to explore specific features more deeply. Following this, Garbage Collectors Tour is employed as the tester would be well-versed enough in the game to achieve objectives as quickly as possible. At this point, the tester is capable of recognizing the least used features, making it the proper time to apply Back Alley Tour. Lastly, considering that the tester is likely well-acquainted with the game and tried different strategies, testing randomly without following any pattern could yield good results (i.e., Exploratory Smoke Testing).

## 8 Concluding Remarks

Testing in games is crucial to ensure game quality and ET, a well-known approach for traditional software, is a promising approach to improving gameplay testing. However, there is a lack of studies or guides addressing how ET can be applied to games. Therefore, this paper contributes to shed some light on this topic by conducting a study that applied seven ET strategies in three indie 2D platform games and two mature and well-known 3D platform games. We reported our approach to setup the test sessions and to adapt the strategies for the games’ domain. At the end, several bugs were revealed with different levels of severity. Overall, the results were positive, providing valuable insights for game testers and motivating further investigations in the future.

The results pave the way for several future works. Further empirical studies would help us to understand if similar results can be achieved with a broader range of testers, how we can better structure the game testing procedure, which other ET strategies may be applied, and the main challenges faced by testers when applying ET. To make these findings actionable for practitioners, we believe that strategies for exploratory game testing should be cataloged and made openly available. Additionally, tool support would aid in managing the testing process, including the definition of charters, selection of strategies, and the conduction of test sessions.

Building upon the results, we plan to experiment with ET strategies within an indie game studio. Since these studios usually have scarce resources for testing, they rely heavily on manual play-testing, and may need to recruit inexperienced

testers or volunteers. We believe that our research would particularly benefit the professionals involved in this context. Providing training and guidelines about ET strategies could enhance the effectiveness of manual video game testing and lead to better outcomes.

## Declarations

### Funding

Yohan Duarte is supported by grant #2022/13469-6, São Paulo Research Foundation (FAPESP) and Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brazil (CAPES) - Finance Code 88887.801592/2023-00. Andre T. Endo is partially supported by grant #2023/00577-8, São Paulo Research Foundation (FAPESP).

### Authors’ Contributions

YD: Conceptualization, Methodology, Resources, Investigation, Data curation, Visualization, Validation, Writing - originaldraft, Writing - review & editing. HCM: Investigation, Data curation, Visualization, Writing - review & editing. VD: Conceptualization, Methodology, Visualization, Writing - originaldraft, Writing - review & editing. PAN: Conceptualization, Methodology, Visualization, Writing - originaldraft, Writing - review & editing. ATE: Conceptualization, Methodology, Validation, Visualization, Project administration, Writing - originaldraft, Writing - review & editing.

### Competing interests

The authors declare that they do not have any competing interests.

### Availability of data and materials

All raw data collected, along with recordings that may be utilized for future replications, can be accessed on the following page: <https://github.com/yohanduartep/et-sessions>

## References

- Al-Azawi, R., Ayesh, A., and Obaidy, M. A. (2013). Generic evaluation framework for games development methodology. *2013 Third International Conference on Communications and Information Technology (ICCIT)*, pages 55–60.
- Aleem, S., Capretz, L. F., and Ahmed, F. (2016). Critical success factors to improve the game development process from a developer’s perspective. *Journal of Computer Science and Technology*, 31:925–950.
- Bach, J. (2003). Exploratory testing explained. <https://satisfice.us/articles/et-article.pdf>.
- Bankhurst, A. (2021). Cyberpunk 2077 teve o maior lançamento digital da história, mostra levantamento. <https://br.ign.com/cyberpunk-2077/86692/news/cyberpunk-2077-teve-o-maior-lancamento-digital-da-historia>. Access on 02 July 2024.
- Chueca, J., Verón, J., Font, J., Pérez, F., and Cetina, C. (2024). The consolidation of game software engineering: A systematic literature review of software

- engineering for industry-scale computer games. *Information and Software Technology*, 165:107330. DOI: <https://doi.org/10.1016/j.infsof.2023.107330>.
- Clement, J. (2023). Digital video game market revenue worldwide from 2017 to 2027. <https://www.statista.com/statistics/1344686/global-digital-gaming-revenue/>. Access on 02 July 2024.
- Copche, R., Souza, M., Villanes, I. K., Durelli, V. H. S., Eler, M., Dias-Neto, A. C., and Endo, A. T. (2021). Exploratory testing of apps with opportunity maps. In *SBQS '21: XX Brazilian Symposium on Software Quality, Virtual Event, Brazil, November 8 - 11, 2021*, page 6. ACM. DOI: <https://doi.org/10.1145/3493244.3493248>.
- Demartini, F. (2021). Reembolsos de cyberpunk 2077 representaram 9% do faturamento da cd projekt red. <https://canaltech.com.br/games/reembolsos-de-cyberpunk-2077-representaram-9-do-faturamento-da-cd-projekt-red-183371/>. Access on 02 July 2024.
- Dias, G. (2021). Mario, pokémon: As franquias mais lucrativas da indústria. <https://www.theenemy.com.br/retro/mario-pokemon-franquias-lucrativas-rankeado>. Access on 02 July 2024.
- Duarte, Y., Durelli, V. H. S., Nardi, P. A., and Endo, A. T. (2023). Exploratory testing strategies for video games: an experience report. In *Proceedings of the 22nd Brazilian Symposium on Games and Digital Entertainment, SBGames 2023, Rio Grande (RS), Brazil, November 6-9, 2023*, pages 46–55. ACM. DOI: <https://doi.org/10.1145/3631085.3631227>.
- Electronic Arts Inc. (2021). It takes two media. <https://www.ea.com/games/it-takes-two/media>. Access on 02 July 2024.
- Escada Games (2024). Diver Down by Escada Games. <https://escada-games.itch.io/diver-down>. Access on 02 July 2024.
- Godot (2024). Godot Engine - Free and open source 2D and 3D game engine. <https://godotengine.org/>. Access on 02 July 2024.
- Harmony Honey (2024). Tiny Crate by Harmony Honey. <https://harmonyhoney.itch.io/tinycrate>. Access on 02 July 2024.
- Hendrickson, E. (2013). *Explore it!: reduce risk and increase confidence with exploratory testing*. Pragmatic Bookshelf.
- Iftikhar, S., Iqbal, M. Z., Khan, M. U., and Mahmood, W. (2015). An automated model based testing approach for platform games. *MODELS '15*, page 426–435. IEEE Press.
- Itkonen, J. (2011). *Empirical studies on exploratory software testing*. PhD thesis. <http://urn.fi/URN:ISBN:978-952-60-4339-5>.
- Juul, J. (2003). The game, the player, the world: looking for a heart of gameness. In *Digital Games Research Conference 2003, 4-6 November 2003, University of Utrecht, The Netherlands*.
- Kaner, C., Falk, J. L., and Nguyen, H. Q. (1999). *Testing Computer Software, Second Edition*. John Wiley & Sons, Inc., 2nd edition.
- Kasurinen, J. and Smolander, K. (2014). What do game developers test in their products? In *2014 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, ESEM '14, Torino, Italy, September 18-19, 2014*, pages 1:1–1:10. ACM. DOI: <https://doi.org/10.1145/2652524.2652525>.
- Lovreto, G., Endo, A. T., Nardi, P., and Durelli, V. H. S. (2018). Automated tests for mobile games: An experience report. In *17th Brazilian Symposium on Computer Games and Digital Entertainment, SBGames 2018, Foz do Iguaçu, Brazil, October 29 - November 1, 2018*, pages 48–56. IEEE Computer Society. DOI: <https://doi.org/10.1109/SBGAMES.2018.00015>.
- Lyndsay, J. and Eeden, N. V. (2003). Adventures in session-based testing. *Workroom Productions Ltd. May*, 27.
- Micallef, M., Porter, C., and Borg, A. (2016). Do exploratory testers need formal training? an investigation using hci techniques. In *2016 IEEE Ninth International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, pages 305–314. DOI: <https://doi.org/10.1109/ICSTW.2016.31>.
- Minkinen, T. (2016). Basics of platform games. <https://api.semanticscholar.org/CorpusID:198316187>.
- Murphy-Hill, E., Zimmermann, T., and Nagappan, N. (2014). Cowboys, ankle sprains, and keepers of quality: How is video game development different from software development? In *Proceedings of the 36th International Conference on Software Engineering, ICSE 2014*, page 1–11, New York, NY, USA. Association for Computing Machinery. DOI: <https://doi.org/10.1145/2568225.2568226>.
- Novak, J. (2017). *Desenvolvimento de Games*. Cengage Learning, 2nd edition.
- OBS Project (2022). Wiki obs studio. <https://obsproject.com/wiki/>. Access on 02 July 2024.
- Politowski, C., Petrillo, F., and Guéhéneuc, Y.-G. (2021a). A survey of video game testing. In *2021 IEEE/ACM International Conference on Automation of Software Test (AST)*, pages 90–99.
- Politowski, C., Petrillo, F., Ullmann, G. C., and Guéhéneuc, Y.-G. (2021b). Game industry problems: an extensive analysis on the gray literature. *Inf. Softw. Technol.*, 134:106538.
- Redavid, C. and Farid, A. (2011). An overview of game testing techniques. <https://api.semanticscholar.org/CorpusID:5653083>.
- Schultz, C. P., Bryant, R., and Langdell, T. (2005). *Game Testing All in One*. Thomson/Course Technology.
- Sriram, V. (2019). Automated playtesting of platformer games using reinforcement learning. Masters thesis. Northeastern University, Boston USA.
- Valve Corporation (2023a). Portal 2 on steam. <https://store.steampowered.com/app/620/Portal2/>. Access on 02 July 2024.
- Valve Corporation (2023b). Portal on steam. <https://store.steampowered.com/app/400/Portal/>. Access on 02 July 2024.
- Whittaker, J. A. (2009). *Exploratory Software Testing: Tips, Tricks, Tours, and Techniques to Guide Test Design*. Addison-Wesley Professional, 1st edition.

- Winpress, M. (2024). Little Spy by Martin Winpress. <https://wimpress.itch.io/little-spy>. Access on 02 July 2024.
- Woods, A. (2020). Fall guys has sold over 11 million copies on pc and is now the most downloaded ps plus game. <https://www.gamesradar.com/fall-guys-has-sold-over-11-million-copies-on-pc-and-is-now-the-most-downloaded-ps-plus-game/>.
- Xavier, B., Viana, D., and Santos, R. (2023). A dive into the state of the practice of the brazilian game software ecosystem. *IEEE Transactions on Games*, pages 1–10. DOI: <https://doi.org/10.1109/TG.2023.3242217>.