Environment. Technology. Resources. Rezekne, Latvia
Proceedings of the 12th International Scientific and Practical Conference. Volume II, 170-174

Analysis of Artificial Intelligence Applications for Automated Testing of Video Games

Imants Zarembo
Faculty of Engineering
Rezekne Academy of Technologies
Rezekne, Latvia
imants.zarembo@rta.lv

***Abstract*—Game testing is a software testing process for quality control in video games. Game environments, sometimes called levels or maps, are complex and interactive systems. These environments can include level geometry, interactive entities, player and non-player controllable characters etc. Depending on the number and complexity of levels, testing them by hand may take a considerable effort. This is especially true for video games with procedurally generated levels that are automatically created using a specifically designed algorithm. A single change in a procedural generation algorithm can alter all of the video game levels, and they will have to be retested to ensure they are still completable or meet any other requirements of the game. This task may be suitable for automation, in particular using Artificial Intelligence (AI). The goal of this paper is to explore the most promising and up-to-date research on AI applications for video game testing to serve as a reference for anyone starting in the field.**

Keywords—Artificial Intelligence, Software Testing, Test Automation, Video Game Testing.

I. Introduction

Video game industry has seen a major expansion in recent years with the number of games being produced rapidly increasing and global games market value rising year over year and reaching $134.9 billion in 2018 [1]. Video game development complexity has grown over the years as well, starting from early generations consisting of simplistic or no graphics at all and restricted to a limited number of commands entered through a keyboard, to modern games with realistic graphics and highly interactive scenarios. This increase in complexity has led to an increase in effort required to ensure quality. Testing is an essential quality assurance activity in software engineering. Software testing is a process of evaluation of the functionality of a software application with an intent to find out whether the developed software meets the specified requirements and to identify defects. In comparison with general software development, video game quality assurance must take into consideration several additional aspects, such as [2][3]:

- Fun factor testing;
- Balance testing;
- Game level/World testing;
- AI testing;
- Multiplayer/Network testing;
- Audio testing;
- Physics testing etc.

Due to increasing demand from game development companies, many video games use procedural generation techniques to generate content [4] to ensure quality and quantity of the content, thus increasing replay value. An example of such procedural generation is game levels which can be automatically created using specifically designed algorithms, which means that player can have new game levels every time he starts the game. Such game levels consist of level geometry, interactive entities, player and non-player characters etc. Testing procedurally generated levels by hand may take a considerable effort and may be a suitable task for automation, in particular using AI for playtesting.

Test automation is a widely used technique of employing special software to control the execution of tests and the comparison of actual outcomes with predicted outcomes. Although automated testing still has its challenges [5] it is widely used in the software development industry for quality assurance. In comparison automated testing in video game development is a less common practice. One reason for that is the fact that video games consist not only from source code but also from assets such as 3D models, textures, sound, music, maps, puzzles etc. [6]. Traditional software quality assurance techniques are not applicable in this case.

The goal of this paper is to explore the most promising and up-to-date research on AI applications for video game testing to serve as a reference for anyone starting in the field. This analysis is the first step in research on creating a framework for automated video game level testing using AI, that would be applicable to procedurally generated video game level testing and validation with as little external involvement as possible. Such an approach would allow game developers to allocate more development time to other parts of the project and provide more value for their customers.

Print ISSN 1691-5402
Online ISSN 2256-070X
http://dx.doi.org/10.17770/etr2019vol2.4158

II. Related Work

A number of approaches have been proposed in the literature to test video games. There is a considerable number of video game testing techniques available that do not rely on traditional software testing techniques. Iftikhar et al. [7] in their paper propose a model-based testing approach for automated black box functional testing of platform games. Peterson et al. [8] present a system and method for performing external and automated testing of video games. Cho et al. [9] propose a system which supports black-box testing and scenario-based testing as well as simple load testing.

Only automatic or semi-automatic approaches focusing on those that use AI were of interest for purposes of this research. Nantes et al. [10] in their work propose a general software framework that integrate Artificial Intelligence Agents and Computer Vision technologies to support the test team and help to improve and accelerate the test process. The agent can replicate the user actions previously tracked in an older version of the game to check for visual anomalies in a newer build of the game. This approach allows making regression test process for environments automatic with no need for any other information about the internal architecture of the game. Gudmundsson et al. [11] present an approach to learn and deploy human-like playtesting in computer games based on deep learning from player data. The proposed approach is able to learn and predict the most "human" action in a given position through supervised learning on a convolutional neural network. The learned network can be used to predict key metrics of new content. The main focus of the approach is on estimating the difficulty of a new level instead of quality assurance of it. Chan et al. [12] present an approach to use evolutionary learning of behaviour to improve testing of commercial computer gamers.

A. Human playing style imitation

One of the uses of human playing style imitation in video games is to understand how a particular player would have played some game content without having the player taking the time to play through the game content [13]. This is especially useful in search-based procedural content generation, where a simulation-based evaluation function uses an AI to play through the candidate game content, assigning a numerical fitness value depending on how playable the content is. The fitness of the level might depend on whether an AI can play through the level or not. This can be used to evaluate content, to test game levels to see if they have bugs and whether they could be completed by a human player. In their paper Ortega et al. compare several different methods for imitating human player behaviour outlined in the following paragraphs.

1) Heuristic

A very simple approach that is based on hand-coded rules that features no learning and ignores the game environment. An example of this approach would be an NPC which simply moves in a certain direction and jumps whenever possible.

2) Artificial neural networks

An artificial neural network (ANN) can be used to simulate human behaviour. A supervised learning ANN approach makes use of direct representation by using the game environment information obtained from human gameplay as training set [14]. This approach uses backpropagation to minimize the error between human player actions and ANN outputs. A neuro-evolutionary approach attempts to minimize a fitness value corresponding to the mean squared error distance from the desired output (human actions) [15].

3) Dynamic scripting

Dynamic scripting (DS) is an online competitive machine-learning technique for game AI, that can be characterized as stochastic optimization [16]. DS contains a rule base with the possible rules that can be applied to a game. Each rule has a weight which reflects how well that rule made the agent perform in prior games. In every game, a script is generated using roulette-wheel in order to select a small subset of the rules in the rule base. The agent will play according to the rules contains in the script and those will have their weights updated via a standard Widrow-Hoff delta rule which is based on the immediate reward received by the environment.

4) REALM

REALM is a rule-based evolutionary computation agent for playing a modified version of Super Mario Bros [17]. REALM follows the principle of learning classifier systems, by which rules are evolved according to a fitness value. Each rule contains conditions based on different information obtained from the game. REALM classifier includes high-level plans of action instead of simple reactive combinations of key presses.

5) Grammatically evolved behaviour trees

Behaviour trees provide a top-down organization from the root of the tree down to the leaves [18]. The control nodes are those that decide which branches of the tree will be executed next, while the leaf nodes contain the actions that are going to be carried out. The different elements of the tree are specified in a grammar which is evolved by applying genetic operations to the sub-tree created. While the evolutionary mechanism is similar to that used in neuroevolution, the behaviour tree representation differs significantly from both neural networks and dynamic scripting.

B. Playtesting with procedural personas

Archetypal player models called procedural personas can be used for generative player modelling and automatic testing of game content [19]. The approach uses a variant of Monte Carlo tree search with genetic programming applied to trees instead of Upper Confidence Bound 1 to evolve persona-specific evaluation formulas. This allows finding mappings between persona utility functions and state evaluation algorithms.

In [20] authors present a method where procedural personas act as critics in search-based procedural content generation framework. For this purpose, personas have been evolved on a set of authored dungeons, according to different fitness that matches archetypical decision-making priorities.

C. ICARUS

Intelligent Completion of Adventure Riddles via Unsupervised Solving [21] is a framework for autonomous video game playing, testing and bug reporting which is based on discrete reinforcement learning in a dualistic fashion, encompassing volatile short-term memory as well as persistent long-term memory that spans across distinct game iterations. It can iterate through complete game iterations and detect or aid the detection of all major bug categories.

D. Hyper-heuristics

Hyper-heuristics approach [22] consists of the creation of hyper-agent for general video game playing that utilizes the strengths of multiple individual controllers to play unseen games better than any of them individually. The hyper-agent uses an offline learning approach by acquiring information about controller performance from a set of trained instances and create a selection model that generalizes well for new games. Hyper-agent does not directly control the main character but selects the best controller to do so.

E. Rolling horizon evolution

Rolling Horizon Evolutionary Algorithms (RHEA) [23] are an alternative to Tree Search for action-decision making in real-time games. Evolutionary Algorithms are used in conjunction with a simulator to train a controller offline and the use the already evolved controller to play the game. RHEA approaches employ evolution in a similar way to how it is done in a tree search, using a forward model to evaluate a sequence of actions.

F. Active learning

Active learning selects among a set of possible inputs to get the best output while minimizing the number of inputs tested. Authors of [24] define the best output as a parameter tuning design goal and treat a set of game design parameters as an input. Minimizing the number of inputs tested minimizes the number of playtests performed.

G. Genetic algorithms

In [25] Genetic algorithms are explored to learn levels from the Mario AI simulator, which is based on Infinite Mario Bros game. Agents learn a sequence of actions by using a genetic algorithm with integer encoding, in order to maximize the attained score after ending the level. This approach executes two different stages: in the first, domain-independent genetic operators are used, while in the second knowledge about the domain is incorporated into these operations in order to improve results.

III. Materials and Methods

Existing research in this field was analysed and synthesised based on whether the described approaches were applicable to automated video game testing using AI. First, research on manual video game testing and testing that does not focus on application of AI was discarded. Then research on automated video game playing using AI was included as automating video game playing and testing are similar tasks and partially overlap in many cases. Finally, an OWL ontology containing a semantic representation of the results of this research was constructed using Protégé and is described in the following section. As automated video game testing research field is relatively new (especially using AI, but in general as well) the main purpose of the ontology is to serve as a starting point for future research in the field.

IV. Results and Discussion

Automated video game testing using AI is relatively new research field often lacking established terminology and structure. This research tries to give overview of automated video game testing approaches and proposes a simple categorization of approaches.

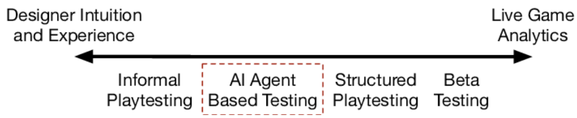

Fig. 1. Spectrum of design testing methods in game development [19].

Fig. 1. Shows the position AI agent-based playtesting takes in the spectrum of video game testing methods right between informal playtesting and structured playtesting.

Analysed video game testing approaches can be broadly categorized into three categories:

1) Human imitation approaches

Human imitation approaches strive to imitate human players in some way to produce results similar to those a human would produce playtesting. This is the most well researched and widely used category of the overviewed approaches. There appears to be a connection between general game AI research that strives to create AI for non-player characters in games that exhibit a behaviour similar to that of human players, and automated video game testing where the quality of video game content must be assured. In both cases, autonomous AI agents can be used to play the game but with different goals in mind.

2) Scenario-based approaches

Scenario-based approaches at least partially rely on previously prepared data and rules to decide further course of actions – which scenario to follow, e.g. dynamic scripting, REALM.

3) Goal-based approaches

Approaches that rely on defining game goals for AI agents to reach fall into this category, e.g. hyper-heuristics, reinforcement learning etc.

A. Ontology

An OWL ontology was constructed categorizing video game testing approaches analysed in this paper (see Fig. 2).

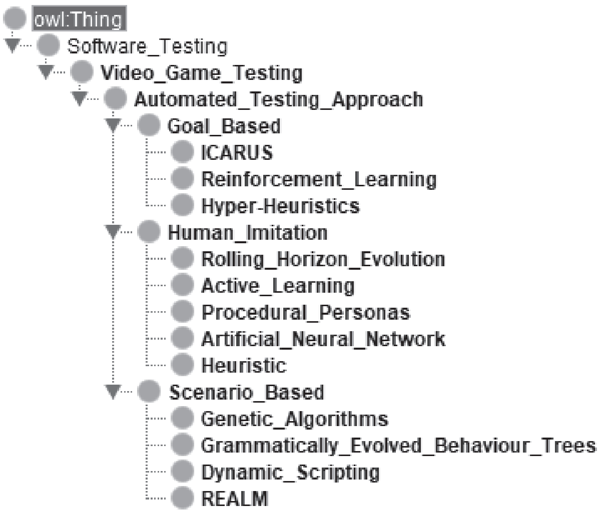

Fig. 2. Automatic video game testing approach ontology

Root node of the ontology starts from “Software Testing” class, which include “Video Game Testing” subclass, followed by “Automated Testing Approach” subclass. Three main subclasses of “Automated Testing Approach” are “Goal Based”, “Human Imitation” and “Scenario Based”. All of the analysed testing approaches are included in one of these categories.

For the sake of visualization simplicity of the ontology video game testing approaches were included as classes instead of individuals.

B. *Performance*

Video game playtesting technique performance and efficiency was out of scope of this paper, the original papers should be consulted for more detail. Alternatively, The General Video Game AI (GVGAI) [26] is a competition to explore the problem of creating controllers for general video game playing. The competition has rankings in several categories:

- Single-player planning;
- Level generation;
- Rule generation;
- Two-player planning;
- Single-player learning.

Depending on the game to be tested exploring these categories may yield well performing appropriate approaches for the task at hand.

V. Conclusions and Future Work

The paper provides an overview of existing automated video game testing approaches and serves as the first step in research of automated procedurally generated game level testing using AI. The vast majority of the analysed approaches of video game testing automation rely on playtesting to produce the results. This research would benefit from more comprehensive analysis including available techniques in connected fields that may be applicable for automated videogame testing but have not yet been adapted for this task.

Future work includes several tasks: defining requirements for the testing automation task to solve; narrowing down the most promising approaches to further analyse, implement and compare their performance; creation of a research prototype of a game which levels an AI agent can test and produce a report; expanding and refining the video game testing approach ontology.

References

[1] J. Batchelor, “GamesIndustry.biz presents... The Year In Numbers 2018,” December, 2018. [Online]. Available: https://www.gamesindustry.biz/articles/2018-12-17-gamesindustry-biz-presents-the-year-in-numbers-2018 [Accessed: Feb. 12, 2019].

[2] L. Levy and J. Novak, “Game Development Essentials: Game QA & Testing,” Cengage Learning, 2009, pp.58-70.

[3] C. P. Schultz and R. D. Bryant, “Game Testing: All in One,” Mercury Learning & Information, 2016, pp.125-128.

[4] M. Hendrikx, S. Meijer, J. V. D. Velden and A. Iosup, “Procedural content generation for games: A survey,” ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), vol. 9, Feb 2013.

[5] A. Bertolino, “Software Testing Research: Achievements, Challenges, Dreams,” Proceedings of The Future of Software Engineering at ICSE 2007, pp. 85-103, 2007.

[6] J. H. Becares, L. C. Valero and P. P. G. Martin, An approach to automated videogame beta testing,” Entertainment Computing, vol. 18, pp. 79-92, Jan 2017.

[7] S. Iftikhar, M. Z. Iqbal, M. U. Khan and W. Mahmood, “An automated model based testing approach for platform games,” The 18th International Conference on Model Driven Engineering Languages and Systems (MODELS), Nov 2015.

[8] K. Peterson, S. Behunin and F. Graham, “Automated Testing on Multiple Video Game Platforums,” U.S. Patent 13/020,959, 4 Feb., 2011.

[9] C. S. Cho, K. M. Sohn, C. J. Park and J. H. Kand, “Online Game Testing Using Scenario-based Control of Massive Virtual Users,” The 12th International Conference on Advanced Communication Technology (ICACT), Feb. 2010.

[10] A. Nantes, R. Brown and F. Maire, “A Framework for the Semi-Automatic Testing of Video Games,” Artificial Intelligence and Interactive Digital Entertainment (AIIDE) 2008.

[11] S. F. Gudmundsson, P. Eisen, E. Poromaa, A. Nodet, S. Purmonen, B. Kozakowski, R. Meurling and L. Cao, “Humsn-Like Playtesting with Deep Learning,” IEEE Conference on Computational Intelligence and Games (CIG), Aug. 2018.

[12] B. Chan, J. Denzinger, D. Gates, K. Loose and J. Buchanan, “Evolutionary behavior testing of commercial computer games,” Proceedings of the 2004 Congress on Evolutionary Computation, Jun. 2004.

[13] J. Ortega, N. Shaker, J. Togelius and G. N. Yannakakis, “Imitating human playing styles in Super Mario Bros,” Entertainment Computing, vol. 4, pp. 93-104, Apr. 2013.

[14] A. J. Champandard, “The Dark Art of Neural Networks,” in “AI Game Programming Wisdom,” S. Rabin, Charles River Media, 2002, pp. 640-651.

[15] D. Floreano, P. Durr and C. Mattiussi, “Neuroevolution: from architectures to learning,” Evolutionary Intelligence, Springer, vol. 1, pp. 47-62, Mar. 2008.

[16] P. Spronck, M. Ponsen, I. Sprinkhuizen-Kuyper and E. Postma, “Adaptive game AI with dynamic scripting,” Machine Learning, Springer, vol. 63, pp. 217-248, Mar. 2006.

[17] S. Bojarski and C. B. Congdon, “REALM: A rule-based evolutionary computation agent that learns to play Mario,” Proceedings of the 2010 IEEE Conference on Computational Intelligence and Games, Aug. 2010.

[18] A. J. Champandard, “Understanding Behavior Trees,” Spetember, 2007. [Online]. Available: http://aigamedev.com/open/article/bt-overview/ [Accessed: Feb. 17, 2019].

[19] C. Holmgard, M. C. Green, A. Liapis and J. Togelius, "Automated Playtesting with Procedural Personas through MCTS with Evolved Heuristics," CoRR, 2018.

[20] A. Liapis, C. Holmgard, G. N. Yannakakis and J. Togelius, "Procedural Personas as Critics for Dungeon Generation," European Conference on the Applications of Evolutionary Computation, pp. 331-343, 2015.

[21] J. Pfau, J. D. Smeddinck and R. Malaka, "Automated Game Testing with ICARUS: Intelligent Completion of Adventure Diddles via Unsupervised Solving," CHI PLAY Extended Abstracts, pp. 153-164, 2017.

[22] A. Mendes, J. Togelius and A. Nealen, "Hyper-heyristic general video game playing," IEEE Conference on Computational Intelligence and Games (CIG), 2016.

[23] D. Perez, S. Samothrakis, S. Lucas and P. Rohlfshagen, "Rolling horizon evolution versus tree search for navigation in single-player real-time games," Proceedings of the 15th annual conference on Genetic and evolutionary computation, pp. 351-358, Jul. 2013.

[24] A. Zook, E. Fruchter and M. O. Riedl, "Automatic playtesting for game parameter tuning via active learning," Foundations of Digital Games, 2014.

[25] A. Baldominos, Y. Saez, G. Reico and J. Calle, "Learning Levels of Mario AI Using Genetic Algorithms," Conference of the Spanish Association for Artifical Intelligence, pp. 267-277, 2015.

[26] D. Perez, S. Samothrakis, J., Togelius, T. Schaul and S. M. Lucas, "General Video Game AI: Competition, Challenges and Opportunities," Association for Advancement of Artificial Intelligence (AAAI), 2016.