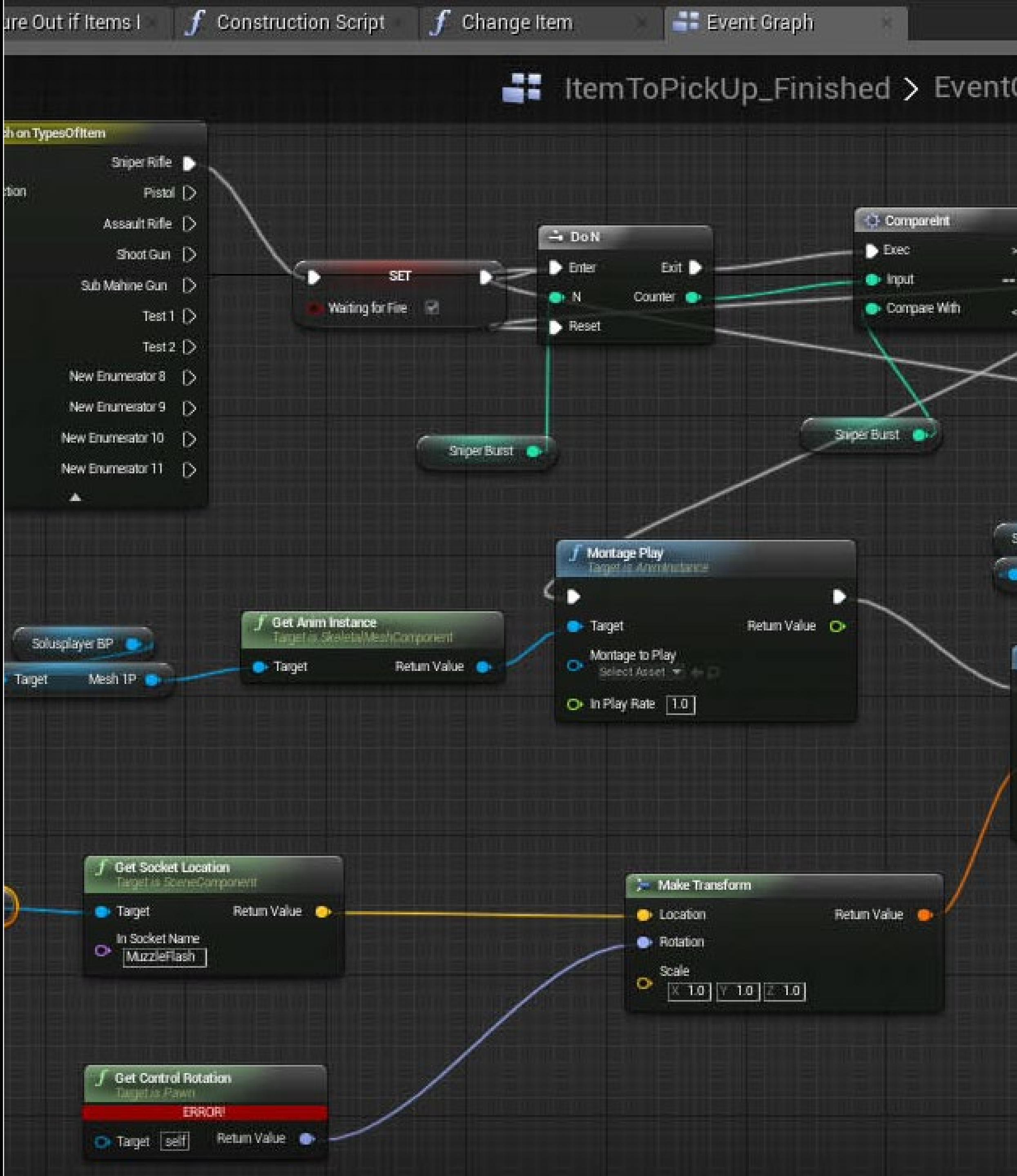
Videogame AI

Course: AI for Media, Art & Design
(A.k.a. “Intelligent Computational Media”)

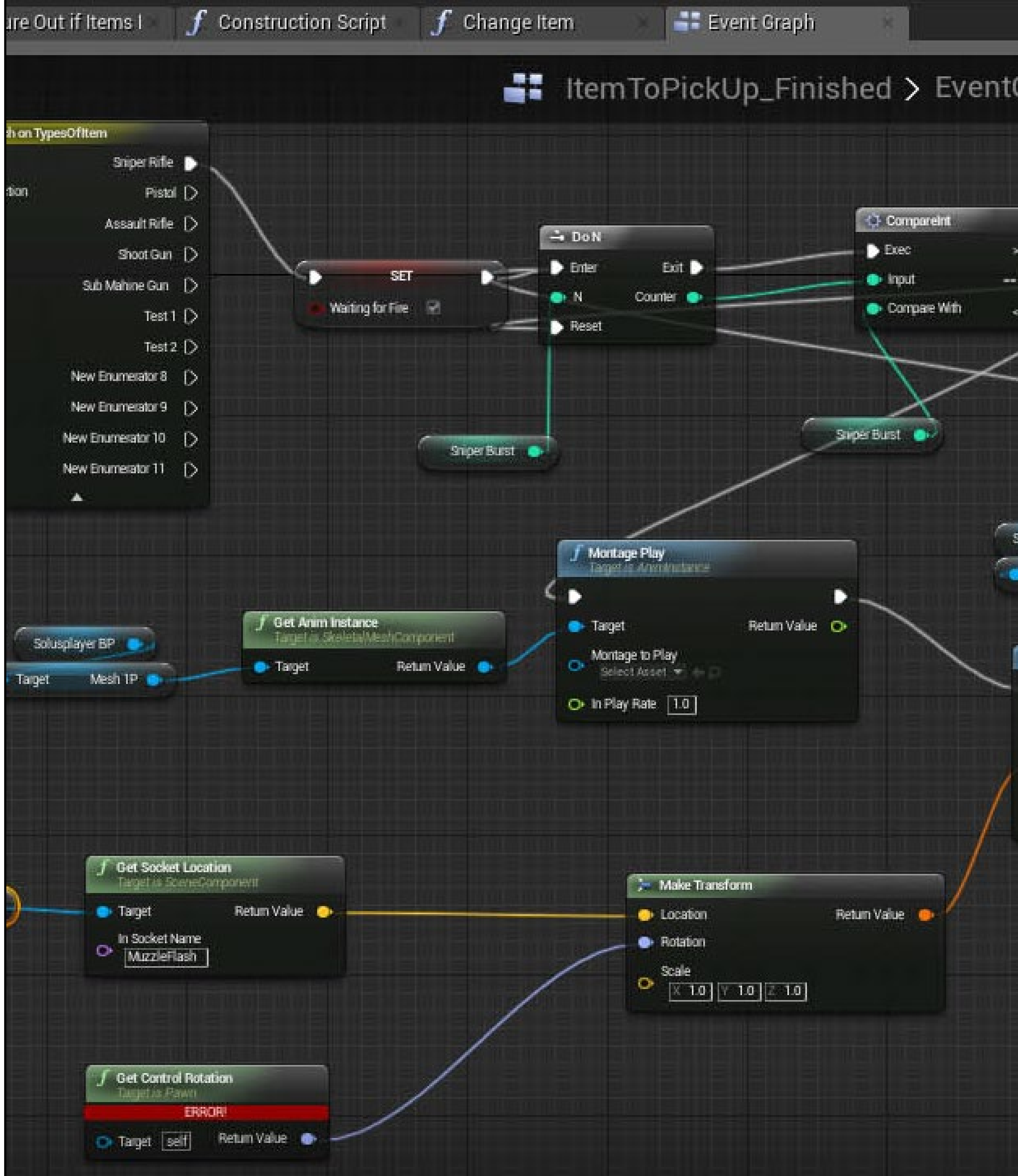
Christian Guckelsberger
christian.guckelsberger@aalto.fi
@creativeEndvs

Aalto-yliopisto
Aalto-universitetet
Aalto University



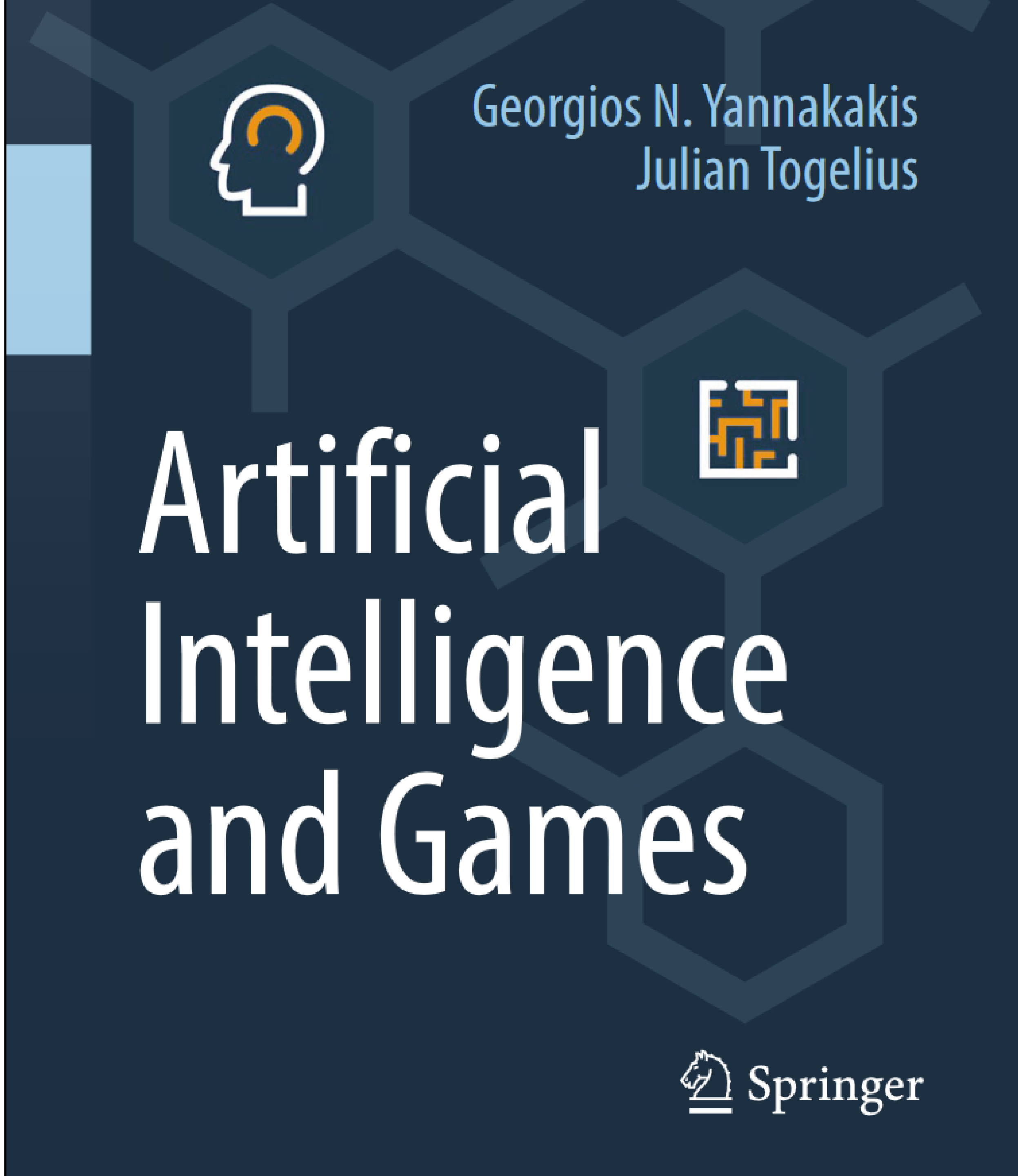
Contents

- What is Game AI?
- Game AI in Industry and Academia
- Overview: 4 Core Areas of Game AI
- Deep Dive: Predicting Player Engagement and Churn with DRL and MCTS
- Summary & further reading



What is Game AI?

- Game AI: ‘the study of AI in and for games (Yannakakis & Togelius, 2018).



Yannakakis, G. N. & Togelius, J. (2018). Artificial Intelligence and Games. Springer.

What is Game AI?

- **Game AI**: ‘the study of AI in and for games (Yannakakis & Togelius, 2018).
- 2 meanings:
  - **AI in games**: games as benchmarks for artificial general intelligence.
  - **AI for games**: to benefit game engineers, designers and players.

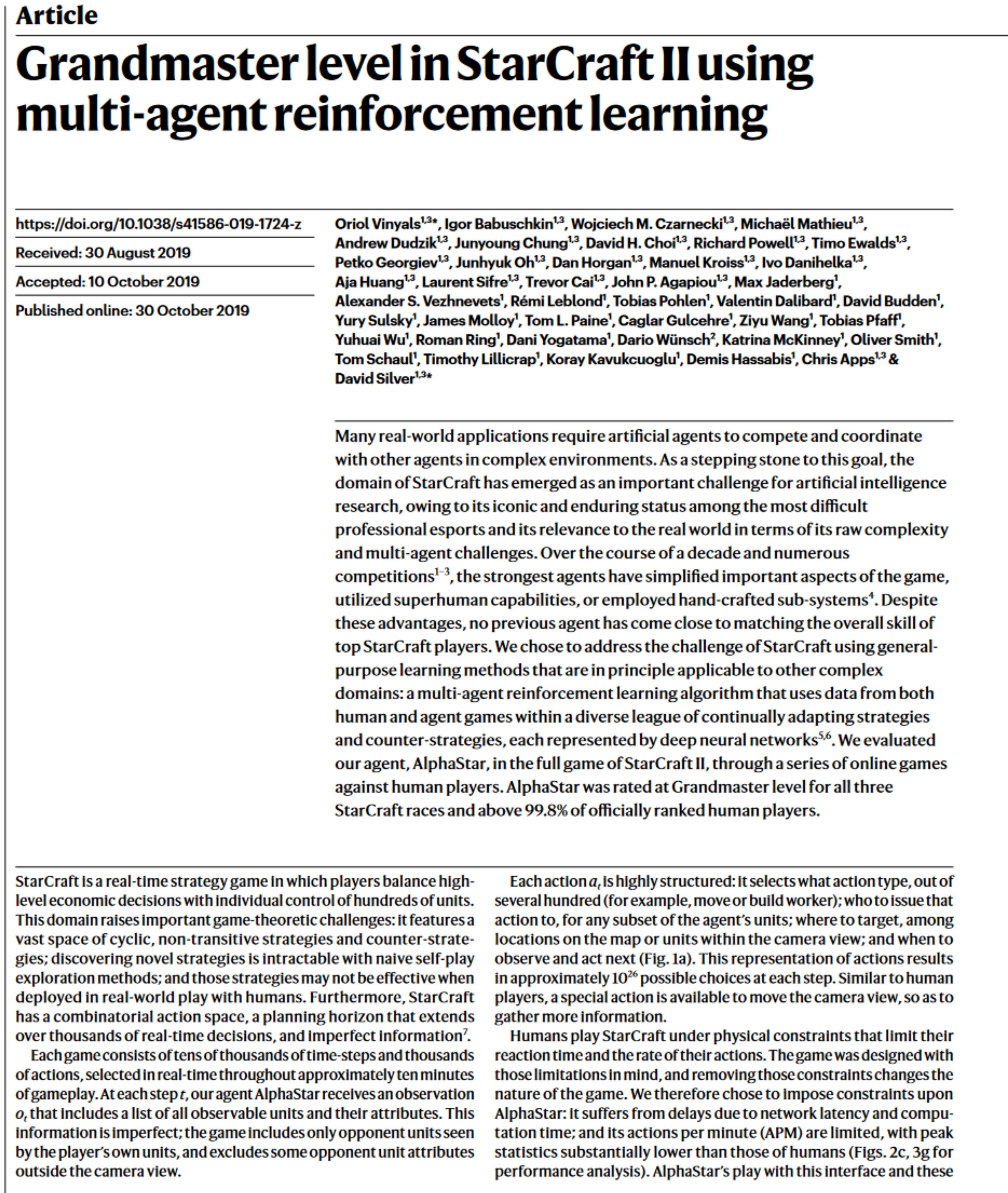
Yannakakis, G. N. & Togelius, J. (2018). Artificial Intelligence and Games. Springer.

What is Game AI?

- **Game AI**: "the study of AI in and for games" (Yannakakis & Togelius, 2018).
- 2 meanings:
  - **AI in games**: games as benchmarks for artificial general intelligence.
  - **AI for games**: to benefit game engineers, designers and players.
- Why game AI research?
  - Games are **motors of technological progress** (Cook, 2015).
  - Advancing game AI can yield **economic, cultural, and societal impact**.
  - Games are **AI/HCI benchmarks**, easing comparison & translation of findings.

Yannakakis, G. N. & Togelius, J. (2018). Artificial Intelligence and Games. Springer.

Cook. (2015). Co-Operative Coevolution for Computational Creativity: A Case Study in Videogame Design. PhD thesis, Imperial College London.



Article

Grandmaster level in StarCraft II using multi-agent reinforcement learning

https://doi.org/10.1038/s41586-019-1724-z

Received: 30 August 2019

Accepted: 10 October 2019

Published online: 30 October 2019

Oriol Vinyals[1,3]*, Igor Babuschkin[1,3], Wojciech M. Czarnecki[1,3], Michaël Mathieu[1,3], Andrew Dudzik[1,3], Junyoung Chung[1,3], David H. Choi[1,3], Richard Powell[1,3], Timo Ewalds[1,3], Petko Georgiev[1,3], Junhyuk Oh[1,3], Dan Horgan[1,3], Manuel Kroiss[1,3], Ivo Danihelka[1,3], Aja Huang[1,3], Laurent Sifre[1,3], Trevor Cai[1,3], John P. Agapiou[1,3], Max Jaderberg[1], Alexander S. Vezhnevets[1], Rémi Leblond[1], Tobias Pohlen[1], Valentin Dalibard[1], David Budden[1], Yury Sulsky[1], James Molloy[1], Tom L. Paine[1], Caglar Gulcehre[1], Ziyu Wang[1], Tobias Pfaff[1], Yuhuai Wu[1], Roman Ring[1], Dani Yogatama[1], Dario Wünsch[2], Katrina McKinney[1], Oliver Smith[1], Tom Schaul[1], Timothy Lillicrap[1], Koray Kavukcuoglu[1], Demis Hassabis[1], Chris Apps[1,3] & David Silver[1,3]*

Many real-world applications require artificial agents to compete and coordinate with other agents in complex environments. As a stepping stone to this goal, the domain of StarCraft has emerged as an important challenge for artificial intelligence research, owing to its iconic and enduring status among the most difficult professional esports and its relevance to the real world in terms of its raw complexity and multi-agent challenges. Over the course of a decade and numerous competitions[1–3], the strongest agents have simplified important aspects of the game, utilized superhuman capabilities, or employed hand-crafted sub-systems[4]. Despite these advantages, no previous agent has come close to matching the overall skill of top StarCraft players. We chose to address the challenge of StarCraft using general-purpose learning methods that are in principle applicable to other complex domains: a multi-agent reinforcement learning algorithm that uses data from both human and agent games within a diverse league of continually adapting strategies and counter-strategies, each represented by deep neural networks[5,6]. We evaluated our agent, AlphaStar, in the full game of StarCraft II, through a series of online games against human players. AlphaStar was rated at Grandmaster level for all three StarCraft races and above 99.8% of officially ranked human players.

StarCraft is a real-time strategy game in which players balance high-level economic decisions with individual control of hundreds of units. This domain raises important game-theoretic challenges: it features a vast space of cyclic, non-transitive strategies and counter-strategies; discovering novel strategies is intractable with naive self-play exploration methods; and those strategies may not be effective when deployed in real-world play with humans. Furthermore, StarCraft has a combinatorial action space, a planning horizon that extends over thousands of real-time decisions, and imperfect information[7].

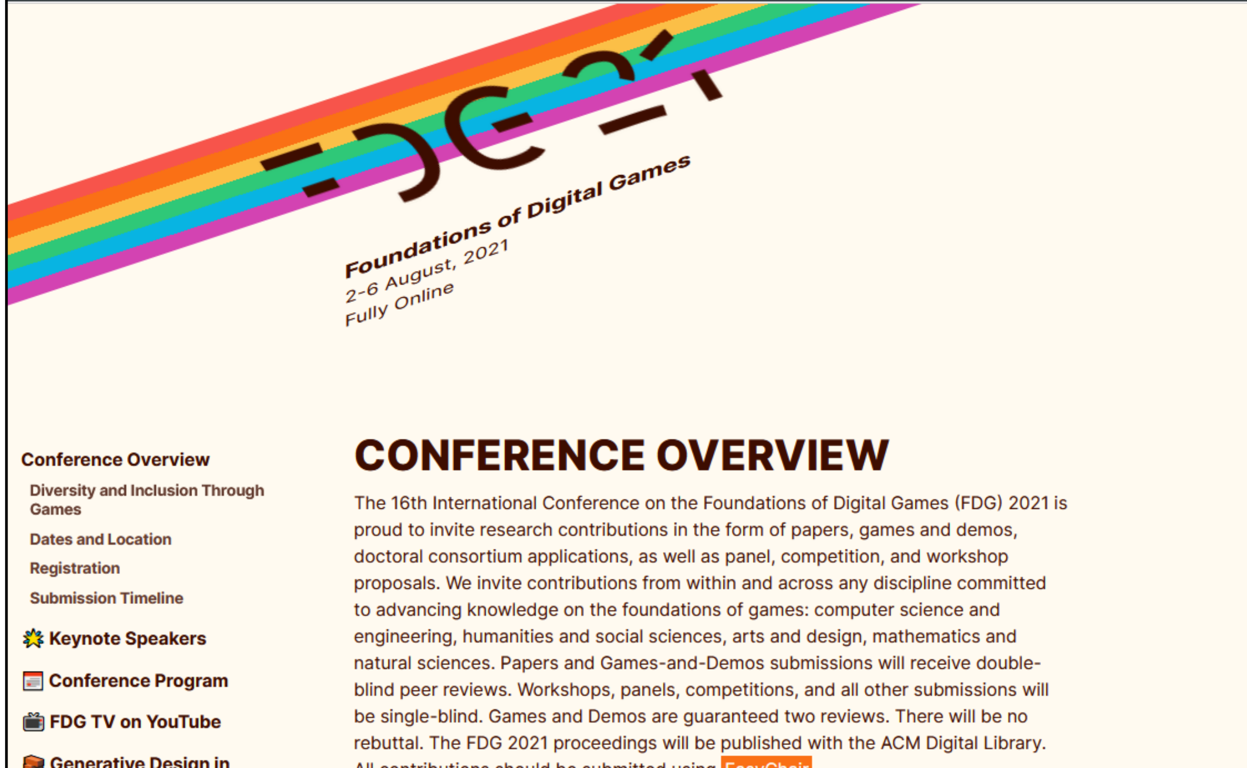
Each game consists of tens of thousands of time-steps and thousands of actions, selected in real-time throughout approximately ten minutes of gameplay. At each step t, our agent AlphaStar receives an observation o_t that includes a list of all observable units and their attributes. This information is imperfect; the game includes only opponent units seen by the player's own units, and excludes some opponent unit attributes outside the camera view.

Each action a_t is highly structured: it selects what action type, out of several hundred (for example, move or build worker); who to issue that action to, for any subset of the agent's units; where to target, among locations on the map or units within the camera view; and when to observe and act next (Fig. 1a). This representation of actions results in approximately 10^{26} possible choices at each step. Similar to human players, a special action is available to move the camera view, so as to gather more information.

Humans play StarCraft under physical constraints that limit their reaction time and the rate of their actions. The game was designed with those limitations in mind, and removing those constraints changes the nature of the game. We therefore chose to impose constraints upon AlphaStar: it suffers from delays due to network latency and computation time; and its actions per minute (APM) are limited, with peak statistics substantially lower than those of humans (Figs. 2c, 3g for performance analysis). AlphaStar's play with this interface and these

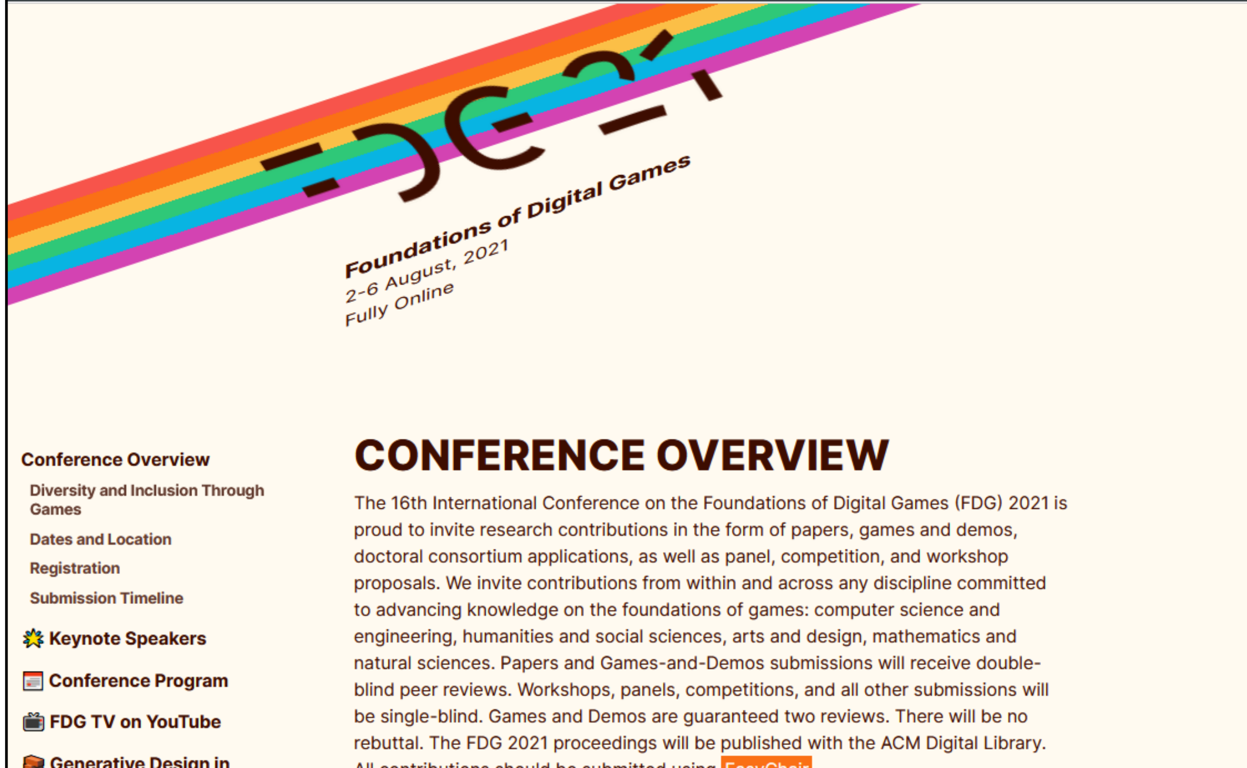
Game AI in Industry and Academia

- Vastly different requirements!
- **Exercise**: which can you think of?



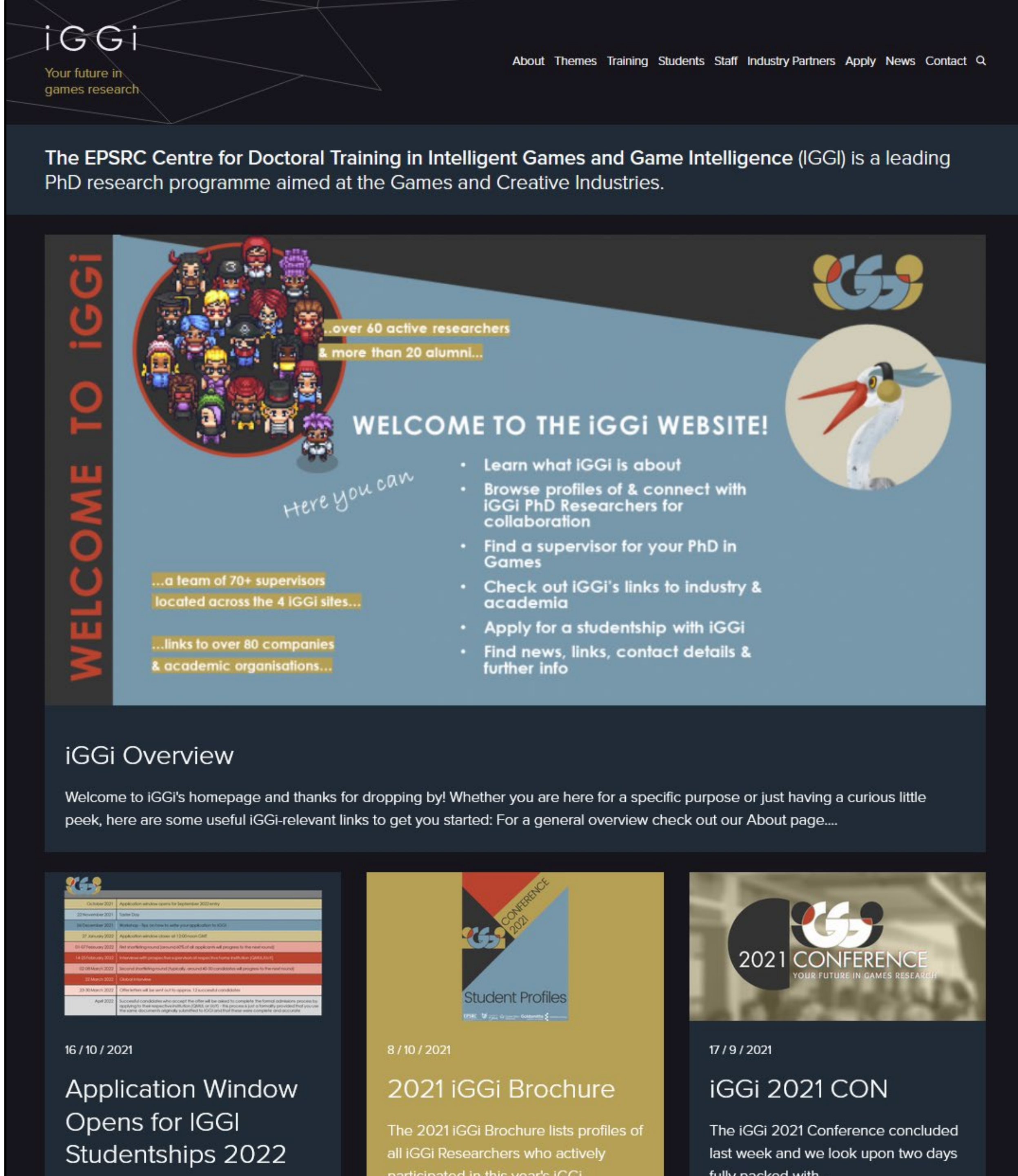
Game AI in Industry and Academia

- Vastly different requirements!
- **Exercise**: which can you think of?
  - **Academia**: experimental, computationally intense, novel, general, publishable.
  - **Industry**: workable, computationally cheap, problem-specific, robust, reliable.



Game AI in Industry and Academia

- Vastly different requirements!
- **Exercise**: which can you think of?
  - **Academia**: experimental, computationally intense, novel, general, publishable.
  - **Industry**: workable, computationally cheap, problem-specific, robust, reliable.
- Bridging the gap:
  - Games **Industry Research Departments**, e.g. SEED (Electronic Arts)
  - **University Spin-Offs**, e.g. Modl.ai
  - **Industry Consultants**, e.g. Tommy Thompson (Lecturer, AI and Games, YouTuber)
  - **PhD Programmes**: Intelligent Games – Game Intelligence (iggi.org.uk)



From: https://www.youtube.com/watch?v=vaHnqXHQvF0 (SEED/Electronic Arts)

Core Game AI (1): (Game-)playing agents

- An artificial agent that can play your game
- **Core requirement:** works reliable even in complex games
- **Exercise**: which use-cases can you think of?

What's the practical use of this technology right now?

Our short-term objective with this project has been to help the DICE team scale up its quality assurance and testing, which would help the studio to collect more crash reports and find more bugs.

In future titles, as deep learning technology matures, I expect self learning agents to be part of the games themselves, as truly intelligent NPCs that can master a range of tasks, and that adapt and evolve over time as they accumulate experience from engaging with human players.

From: www.ea.com/en-gb/news/teaching-ai-agents-battlefield-1

Core Game AI (1): (Game-)playing agents

- An artificial agent that can play your game
- Core requirement: works reliable even in complex games
- Exercise: which use-cases can you think of?
  - Speed up playtesting
  - Explore levels more thoroughly
  - Find bugs / crash reports
  - Simulate different player types
  - Remove learning effects
  - …

Better Quality Content

A faster playtest allows for more iterations of the new levels. It means that level designers can refine more quickly. Because playtesting is not time-consuming anymore, it is possible to get feedback right before release to make sure that all tweaks work as intended. Finally, level designers can focus on the same content throughout the day, reducing the context switching mentioned above which impacts creativity.

More Thorough and Stable Playtests

One issue with human playtesters is that inherently, the more they play the better they get at the game. This introduces a bias into their feedback. Virtual players are version-controlled software, therefore avoiding such bias. On top of that, the measures are both more precise and diverse, since they communicate directly with the game engine.

A QA Byproduct

By building an automated playtesting platform for content balancing purposes, we actually created a QA byproduct for developers. They can use the platform to explore levels and find bugs. They can also check that new features don't break the rest of the game. It is a powerful tool to increase the game's quality as a whole.

From: www.ea.com/en-gb/news/teaching-ai-agents-battlefield-1

Core Game AI (1): **(Game-)playing agents**

AI Playtesting:

”Attempting to maximize coverage of a game via human gameplay is laborious and repetitive, introducing delays in the development process. Despite the importance of quality assurance (QA) testing, QA remains an underinvested area in the technical games research community. In this paper, we show that relatively simple automatic exploration techniques can be used to multiplicatively amplify coverage of a game starting from human tester data”.

Reveal-More: Amplifying Human Effort in Quality Assurance Testing Using Automated Exploration

Kenneth Chang
University of California, Santa Cruz
Santa Cruz, CA, USA
kchang44@ucsc.edu

Batu Aytemiz
University of California, Santa Cruz
Santa Cruz, CA, USA
baytemiz@ucsc.edu

Adam M. Smith
University of California, Santa Cruz
Santa Cruz, CA, USA
amsmith@ucsc.edu

***Abstract*—Attempting to maximize coverage of a game via human gameplay is laborious and repetitive, introducing delays in the development process. Despite the importance of quality assurance (QA) testing, QA remains an underinvested area in the technical games research community. In this paper, we show that relatively simple automatic exploration techniques can be used to multiplicatively amplify coverage of a game starting from human tester data. Instead of attempting to displace human QA efforts, we seek to grow the impact that a human tester can make. Experiments with two games for the Super Nintendo Entertainment System highlight the qualitative and quantitative differences between isolated human and machine play compared to our hybrid approach called Reveal-More. We contribute a QA testing workflow that scales with the amount of human and machine time allocated to the effort.**

I. Introduction

In quality assurance (QA) testing for videogames, conventional wisdom holds that automated approaches answer *software* questions (e.g. does processing this sequence of inputs yield the expected output?) and manual testing answers *gameplay* questions (e.g. will the game crash if I collect this item?). Nascent research efforts in automatic testing have tried to apply artificial intelligence (AI) methods to the problem of demonstrating interesting possibilities in play that developers might interpret to answer design and implementation questions that impact gameplay. So far, separated human and machine testing processes have shown complementary strengths [1], as expected [2]. In this paper, we are interested in directly amplifying human tester effort to answer gameplay questions by using recordings of their play as the seeds for automated exploration.

Without automation, identifying inputs that lead to gameplay issues is a massive exploratory search problem that requires significant resource expenditure. Even in the simplest of videogames, there may be an astronomical number of distinct gameplay paths, only a few of which trigger a bug. In an ideal world, QA testers would indicate which span of a game is most relevant to them, and a system would quickly show them what was possible (or impossible) in that part of the game. Testers would save their efforts for directing, rather than enacting, repetitive gameplay experiments. Towards this goal, we formulate our problem as maximizing game state coverage in the service of encountering game design problems.

While there has been high profile successes in automatic gameplaying research [3], only recently has exploration specifically drawn attention [4]. Score optimization techniques such as Reinforcement Learning (RL) [5] and Monte-Carlo Tree Search (MCTS) [6] are setup to solve a different problem from the one faced in exploration. Techniques like MCTS may systematically avoid exploring certain play styles of interest simply because they earn lower scores. Additionally, the timescale on which automated gameplay techniques achieve useful results (i.e. minutes versus years of simulated gameplay) has only recently drawn attention [4]. For exploration to be useful in the QA process, useful reports need to be generated on timescales comparable to the pace of game design cycles (such as being able to provide feedback on weekly or daily game builds).

In this paper, we demonstrate a new technique, Reveal-More, that combines automatic exploration with just minutes of human gameplay, resulting in game state coverage that is superior to using each individual method alone. In such a manner, an automated method of exploration is used to amplify what a person can contribute to testing, thus lowering the strain placed upon testers to find all the paths in a game. To anchor our work in game development practice, we carry out experiments in the commercial implementation of two culturally significant games. In several experiments with *Super Mario World* and *The Legend of Zelda*, we demonstrate up to a 5X increase in our quantitative exploration metric, and qualitatively illustrate the significance of increased coverage. Furthermore, we show that this amplified coverage can be helpful in visualizing design changes and, in turn, help characterize the impact of design changes.

II. Related Work

Common practice in game QA testing involves having many people play the game with the goal of covering the most ground in it. There exists some automation towards this goal [7], however the majority of the technical games research community has focused on creating algorithms that aim to maximize in-game score. In the search for the best QA practices, whether through automation or manual testing, many have agreed that maximizing some sense of *coverage* is a central concern [8]–[10].

978-1-7281-1884-0/19/$31.00 ©2019 IEEE

Core Game AI (1): **(Game-)playing agents**

Reveal-More: Amplifying Human Effort in Quality Assurance Testing Using Automated Exploration

Kenneth Chang
University of California, Santa Cruz
Santa Cruz, CA, USA
kchang44@ucsc.edu

Batu Aytemiz
University of California, Santa Cruz
Santa Cruz, CA, USA
baytemiz@ucsc.edu

Adam M. Smith
University of California, Santa Cruz
Santa Cruz, CA, USA
amsmith@ucsc.edu

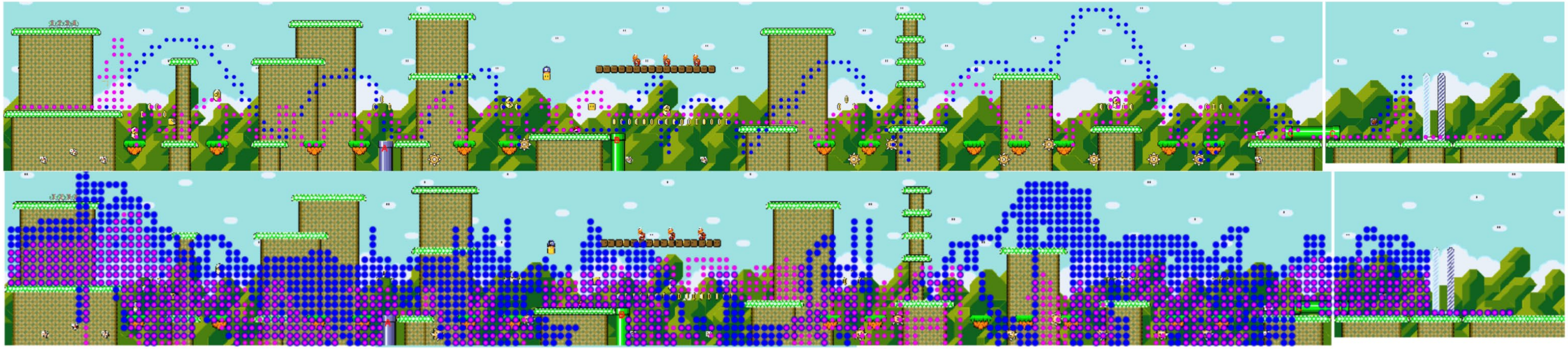


Fig. 8: A comparison of two different versions of SMW. The top image shows the human gameplay trace in the original (magenta) and modified (blue) designs while the bottom image shows the amplified coverage discovered with Reveal-More.

multiplicatively amplify coverage of a game starting from human tester data”.

play issues is a massive exploratory search problem that requires significant resource expenditure. Even in the simplest of videogames, there may be an astronomical number of distinct gameplay paths, only a few of which trigger a bug. In an ideal world, QA testers would indicate which span of a game is most relevant to them, and a system would quickly show them what was possible (or impossible) in that part of the game. Testers would save their efforts for directing, rather than enacting, repetitive gameplay experiments. Towards this goal, we formulate our problem as maximizing game state coverage in the service of encountering game design problems.

978-1-7281-1884-0/19/$31.00 ©2019 IEEE

II. Related Work

Common practice in game QA testing involves having many people play the game with the goal of covering the most ground in it. There exists some automation towards this goal [7], however the majority of the technical games research community has focused on creating algorithms that aim to maximize in-game score. In the search for the best QA practices, whether through automation or manual testing, many have agreed that maximizing some sense of *coverage* is a central concern [8]–[10].

Core Game AI (2): Non-Player Characters

- AI to steer Non-Player Character, potentially in interaction with the player
- Many **types**, **integral** to most games: (v.d. Herik et al., 2005; Warpefelt, 2016)
  - Aversaries: Enemies, Opponents.
  - Friends: Team-mates, Companions, Pets, …
  - Neutral: Vendors, Quest Givers, Conversation Partners …

Van den Herik, H. J., Donkers, H. & Spronck, P. H. (2005). Opponent Modelling and Commercial Games. Proc. Conference on Computational Intelli gence and Games (CIG), 15–25.

Warpefelt, H. (2016). The Non-Player Character: Exploring the Believability of NPC Presentation and Behavior (Doctoral dissertation). Stockholm University.

Core Game AI (2): Non-Player Characters

- **Core requirement:** believability! Determined by visual appearance + behaviour (AI!)
- **Games Industry Reality:**
  - Manual scripting
  - Finite state machines
  - Behaviour trees

Warpefelt, H. & Verhagen, H. (2017). A Model of Non-Player Character Believability. Journal of Gaming & Virtual Worlds, 9(1), 39–53.

Emmerich, K., Ring, P. & Masuch, M. (2018). I'm Glad You Are on My Side: How to Design Compelling Game Companions. Proc. SIGCHI Annual Symposium on Computer-Human Interaction in Play (CHI'Play), 141–152.

Determinant	Reference	Description
Characterhood	Warpefelt (2016)	Actively and rationally portray role in a way that convinces the player.
Behavioural Diversity	Yannakakis and Hallam (2004)	Avoid the repetition of behaviour in the same or similar situation.
Sensitivity to Body and Surroundings	Lankoski and Björk (2007)	Perceive and respond to changes in own body and surroundings.
Own Agenda	Lankoski and Björk (2007); Lee and Heeter (2012)	Follow own agenda and take initiative independently of player actions.

Table 6.1: Determinants of believable NPC behaviour summarised from the literature.

Core Game AI (2): Non-Player Characters

- **Core requirement:** believability! Determined by visual appearance + behaviour (AI!)
- **Games Industry Reality:**
  - Manual scripting
  - Finite state machines
  - Behaviour trees
- **Academic Research:**
  - General Companion / Enemy NPCs: plug into arbitrary game, get support/antagonism (Guckelsberger et al., 2016, 2018)
  - …

Warpefelt, H. & Verhagen, H. (2017). A Model of Non-Player Character Believability. Journal of Gaming & Virtual Worlds, 9(1), 39–53.

Emmerich, K., Ring, P. & Masuch, M. (2018). I'm Glad You Are on My Side: How to Design Compelling Game Companions. Proc. SIGCHI Annual Symposium on Computer-Human Interaction in Play (CHI'Play), 141–152.

New And Surprising Ways to Be Mean: Adversarial NPCs with Coupled Empowerment Minimisation

Christian Guckelsberger[1,*], Christoph Salge[2,3] and Julian Togelius[3]

***Abstract*— Creating Non-Player Characters (NPCs) that can react robustly to unforeseen player behaviour or novel game content is difficult and time-consuming. This hinders the design of believable characters, and the inclusion of NPCs in games that rely heavily on procedural content generation. We have previously addressed this challenge by means of empowerment, a model of intrinsic motivation, and demonstrated how a coupled empowerment maximisation (CEM) policy can yield generic, companion-like behaviour. In this paper, we extend the CEM framework with a minimisation policy to give rise to adversarial behaviour. We conduct a qualitative, exploratory study in a dungeon-crawler game, demonstrating that CEM can exploit the affordances of different content facets in adaptive adversarial behaviour without modifications to the policy. Changes to the level design, underlying mechanics and our character's actions do not threaten our NPC's robustness, but yield new and surprising ways to be mean.**

behaviour or changes in the game world. The latter aspect is partly alleviated by reinforcement learning, evolutionary approaches or planning. However, there is two caveats. Algorithms such as Monte Carlo Tree Search (MCTS) are typically targeted towards maximising adversarial efficacy against the player, resulting in blunt and single-faceted behaviour. Procedural personas [3] contribute to the impression of more multi-faceted behaviour by optimising a set of pre-specified utilities. However, even these advanced approaches usually rely on objective functions, rewards and training samples which are strongly tied to specific affordances of the game world. As soon as this world changes, the basis for their behaviour and thus their believability is lost.

An alternative approach is to use models of intrinsic motivation [4] to drive NPC behaviour. Models of intrinsic

Intrinsically Motivated General Companion NPCs via Coupled Empowerment Maximisation

Christian Guckelsberger, Computational Creativity Group, Goldsmiths, University of London, London, United Kingdom, Email: c.guckelsberger@gold.ac.uk

Christoph Salge, Adaptive Systems Research Group, University of Hertfordshire, Hatfield, United Kingdom, Email: c.salge@herts.ac.uk

Simon Colton, The MetaMakers Institute, Falmouth University, Falmouth, United Kingdom, Email: s.colton@gold.ac.uk

***Abstract*—Non-player characters (NPCs) in games are traditionally hard-coded or dependent on pre-specified goals, and consequently struggle to behave sensibly in ever-changing and possibly unpredictable game worlds. To make them fit for new developments in procedural content generation, we introduce the principle of *Coupled Empowerment Maximisation* as an intrinsic motivation for game NPCs. We focus on the development of a general game companion, designed to support the player in achieving their goals. We evaluate our approach against three intuitive and abstract companion duties. We develop dedicated scenarios for each duty in a dungeon-crawler game testbed, and provide qualitative evidence that the emergent NPC behaviour fulfils these duties. We argue that this generic approach can speed up NPC AI development, improve automatic game evolution and introduce NPCs to full game-generation systems.**

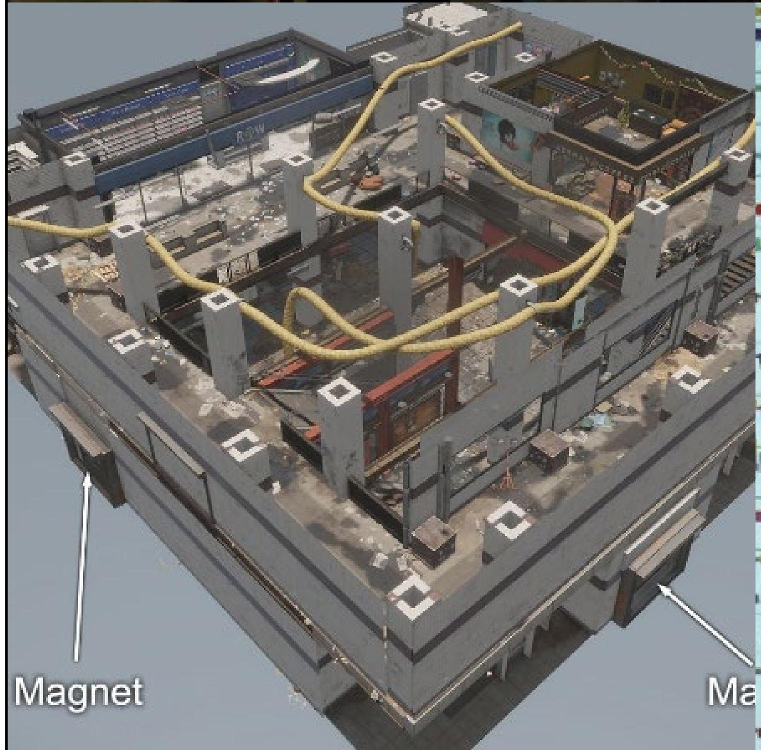
I. Introduction

in the agent's sensorimotor relationship with the world [8], so changes to the world or the agent's embodiment are reflected in potentially new behaviour. A curious mouse and a curious bird would consequently behave differently, moderated by their embodiment and environment. In this paper, we will work with the intrinsic motivation of *empowerment* [10], a measure of how much an agent is in control of the world it can perceive. We have previously argued [11] that empowerment reflects an agent's drive to maintain its own precarious existence, and allows them to adapt to changes in their embodiment and environment. But while empowerment might be very useful to produce an intrinsically motivated *general NPC*, we have to look specifically into how to turn it into a good *companion*.

Players seem to expect a companion to behave differently than a general NPC. For instance, in a qualitative study

Core Game AI (3): Procedural Content Generation (PCG)

- Generate game content algorithmically: level layouts, music, mechanics, character animations, graphics, …
- Core requirement: consistent quality
- Games Industry Reality:
  - Speed Tree (yawn…)
  - Grammar-based approaches (e.g. No Man's Sky)
  - Experiential chunks (e.g. The Division, …)



Core Game AI (3): Procedural Content Generation (PCG)

- **Generate game content algorithmically**: level layouts, music, mechanics, character animations, graphics, …
- **Core requirement:** consistent quality
- **Games Industry Reality:**
  - Speed Tree (yawn…)
  - Grammar-based approaches (e.g. No Man's Sky)
  - Experiential chunks (e.g. The Division, …)
- **Academic Research**:
  - ANGELINA: full automatic game generation (Cook, …)
  - Search-Based PCG (e.g. genetic algs)
  - ML-/RL-based PCG

IEEE TRANSACTIONS ON GAMES, VOL. 10, NO. 3, SEPTEMBER 2018 257

Procedural Content Generation via Machine Learning (PCGML)

Adam Summerville, Sam Snodgrass, Matthew Guzdial, Christoffer Holmgård, Amy K. Hoover, Aaron Isaksen, Andy Nealen, and Julian Togelius

***Abstract*—This survey explores procedural content generation via machine learning (PCGML), defined as the generation of game content using machine learning models trained on existing content. As the importance of PCG for game development increases, researchers explore new avenues for generating high-quality content with or without human involvement; this paper addresses the relatively new paradigm of using machine learning (in contrast with search-based, solver-based, and constructive methods). We focus on what is most often considered functional game content, such as platformer levels, game maps, interactive fiction stories, and cards in collectible card games, as opposed to cosmetic content, such as sprites and sound effects. In addition to using PCG for autonomous generation, cocreativity, mixed-initiative design, and compression,**

increasingly prominent within both game development and technical games research. It is employed to increase replay value, reduce production cost and effort, to save storage space, or simply as an aesthetic in itself. Academic PCG research addresses these challenges, but also explores how PCG can enable new types of game experiences, including games that can adapt to the player. Researchers also address challenges in computational creativity and ways of increasing our understanding of game design through building formal models [1].

In the games industry, many applications of PCG are what could be called "constructive" methods, using grammars or

Proceedings of the Sixteenth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE-20)

PCGRL: Procedural Content Generation via Reinforcement Learning

Ahmed Khalifa, Philip Bontrager, Sam Earle, Julian Togelius
New York University
ahmed@akhalifa.com, philipjb@nyu.edu, smearle93@gmail.com, julian@togelius.com

Abstract

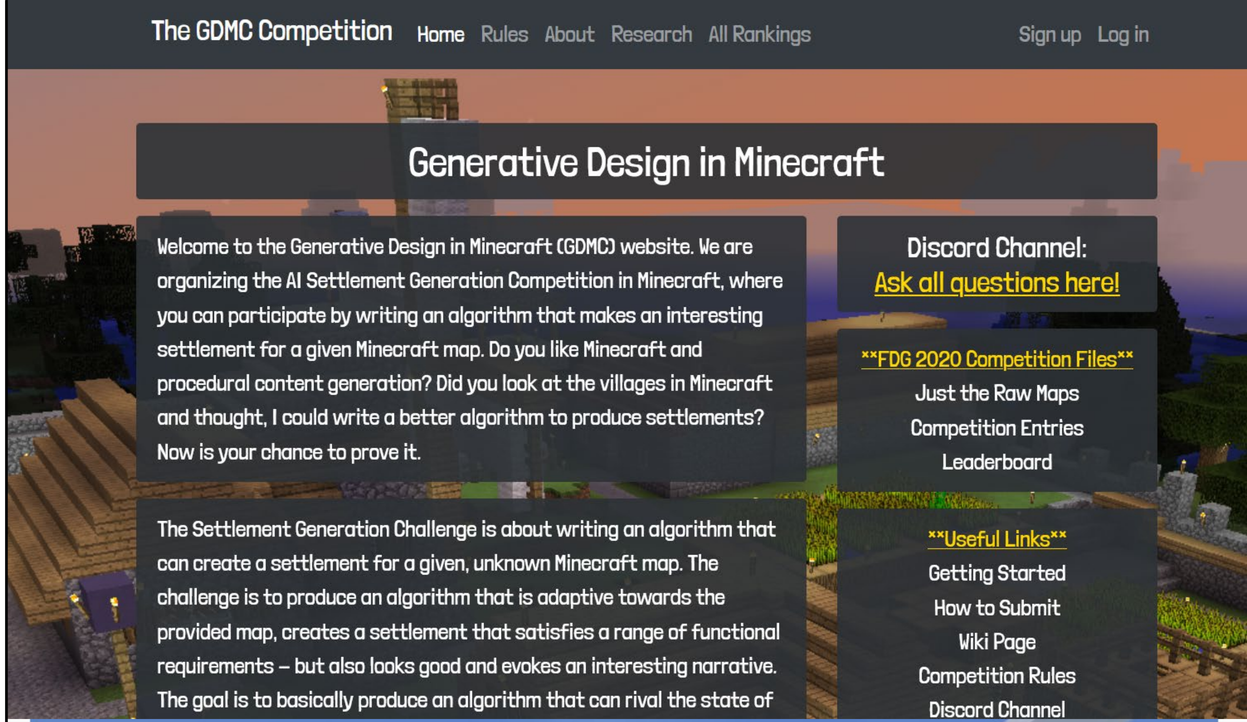
We investigate how reinforcement learning can be used to train level-designing agents. This represents a new approach to procedural content generation in games, where level design is framed as a game, and the content generator itself is learned. By seeing the design problem as a sequential task, we can use reinforcement learning to learn how to take the next action so that the expected final level quality is maximized. This approach can be used when few or no examples exist to train from, and the trained generator is very fast. We investigate three different ways of transforming two-dimensional level design problems into Markov decision processes, and

Conceptually, the main difference to existing approaches to Procedural Content Generation (PCG) is that we do not search the space of game content, but rather the space of policies that generate game content. At each step, the policy is asked to take the action that leads to the highest expected final level quality. This can be contrasted to search-based approaches where each "action" generates a complete level, or to approaches based on supervised or unsupervised learning where complete levels are sampled from a learned model.

Reinforcement learning approaches to PCG have several potential advantages over existing methods. Compared to search-based methods (Togelius et al. 2011), machine learn-

Core Game AI (3): Procedural Content Generation (PCG)

- Shameless Plug: **Generative Design in Minecraft Competition** (GDMC)
- **Task**: write an algorithm to produce cities in the popular game Minecraft.
- Bringing together researchers and public to advance PCG. **Challenges**:
  - Greater adaptivity to existing content (terrain, biotope, cities)
  - Holistic PCG or orchestration
- Competition running in its 5th year.
- Big, welcoming community. https://gendesignmc.engineering.nyu.edu



Core Game AI (4): User Modelling

- **Distinguish**: player experience vs. player behaviour modelling
- **Core requirement:** speed, accuracy
- **Exercise**: what use-cases can you think of?

Core Game AI (4): User Modelling

- **Distinguish**: player experience vs. player behaviour modelling
- **Core requirement:** speed, accuracy
- **Exercise**: what use-cases can you think of?
  - Content quality assurance offline/online: synthetic players
  - Increasing player engagement, especially for free-to-play games / in-game purchases
  - Personalise content online (e.g. dynamic difficulty adjustment)
  - …

Core Game AI (4): User Modelling

- **Distinguish**: player experience vs. player behaviour modelling
- **Core requirement:** speed, accuracy
- **Exercise**: what use-cases can you think of?
  - Content quality assurance offline/online: synthetic players
  - Increasing player engagement, especially for free-to-play games / in-game purchases
  - Personalise content online (e.g. dynamic difficulty adjustment)
  - …
- **Power in combination**:
  e.g. user experience modelling + PCG = experience-driven PCG.

Experience-Driven Procedural Content Generation

Georgios N. Yannakakis, *Member, IEEE,* and Julian Togelius, *Member, IEEE*

Abstract—Procedural content generation (PCG) is an increasingly important area of technology within modern human-computer interaction (HCI) design. Personalization of user experience via affective and cognitive modeling, coupled with real-time adjustment of the content according to user needs and preferences are important steps towards effective and meaningful PCG. Games, Web 2.0, interface and software design are amongst the most popular applications of automated content generation. The paper provides a taxonomy of PCG algorithms and introduces a framework for PCG driven by computational models of user experience. This approach, which we call *Experience-Driven Procedural Content Generation* (EDPCG), is generic and applicable to various sub-areas of HCI. We employ games as an indicative example of rich HCI and complex affect elicitation, and demonstrate the approach's effectiveness via dissimilar successful studies.

Index Terms—procedural content generation, user affect, user experience, personalization, adaptation, computer games.

1 Introduction

As information about users is becoming more readily available for all kinds of digital services and modern games becomes increasingly difficult. Game engines [4] that are able to recognize and model the playing style and detect the affective state of the user will be necessary milestones

champion (Fig. 2(b))[6] contains large and challenging gaps whereas the generated level of maximum fun value for the human (Fig. 2(a)) contains more gaps placed in a more unpredictable manner.

Assuming the playing style of a particular player is known, the level of each of the six affective states can be predicted for any particular level (expressed as a parameter vector) by simply feeding the level parameters together with the player parameters to the neural network. This means that the neural network can act as an evaluation function for black-box search or optimization, using for example evolutionary algorithms or exhaustive search.

While emotional response is only measured via self-reports (and not bodily reactions, for instance) in this study, our affective models rely upon the assumption that player emotions can be inferred via the association of user self-reports and game context variables [30], [31].

2.2 This paper

Below, we survey the four main components of EDPCG, and provide a taxonomy of different approaches to each and outline the main research challenges faced. We also give a non-exhaustive number of examples that fully or partly adopt the principles of EDPCG. Each component of EDPCG has its own dedicated literature and the extensive review of each would be beyond the scope of this paper. Thus, the survey attempts to highlight representative work that relates to the key components of EDPCG and discuss, in part, studies that

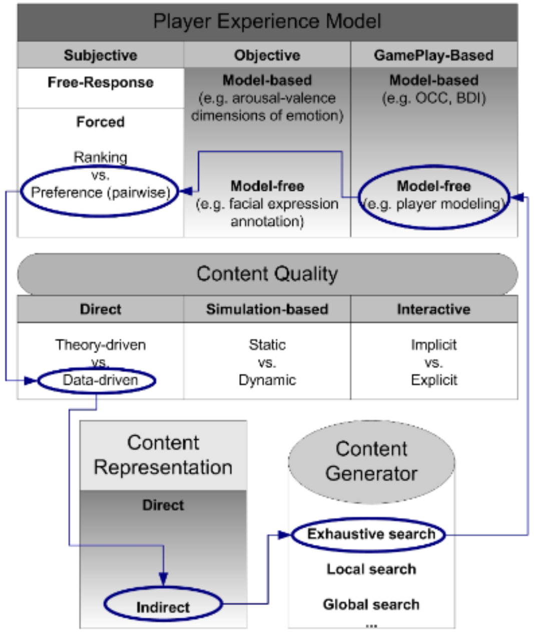


Fig. 3. The EDPCG framework in detail. The gradient

Aalto University
School of Science
and Technology

ROVIO

Predicting Game Difficulty and Engagement

Using AI players

Shaghayegh Roohi[2,1], Christian Guckelsberger[2], Asko Relas[1], Jari Takatalo[1], Henri Heiskanen[1], Perttu Hämäläinen[2]
1) Rovio Entertainment, 2) Aalto University

An extension of our CHI PLAY 2020 work: Predicting Game Difficulty and Churn Without Players

Motivation

- **Human playtesting:** repetitive, tedious, expensive, time-consuming
- AI players can reduce need for human playtesters. **Here: AI-driven player experience / behavior modelling**
- **Potential applications:**
  - Inform the selection of the best from a range of existing designs, e.g. levels
  - Procedurally generate optimal designs, using predictions from the experience / behavior model as objective function

Playtesting for Indies, https://www.gamedeveloper.com/production/playtesting-for-indies

Motivation

Extending our previous CHI Play work on predicting game difficulty and churn using deep reinforcement learning (DRL) agents:

Roohi, S., Relas, A., Takatalo, J., Heiskanen, H., & Hämäläinen, P. 2020. Predicting Game Difficulty and Churn Without Players. In Proc. *CHI Play,* pp. 585-593.

Predicting Game Difficulty and Churn Without Players

Shaghayegh Roohi
shaghayegh.roohi@aalto.fi
Aalto University
Espoo, Finland

Asko Relas
asko.relas@rovio.com
Rovio Entertainment
Espoo, Finland

Jari Takatalo
jari.takatalo@rovio.com
Rovio Entertainment
Espoo, Finland

Henri Heiskanen
henri.heiskanen@rovio.com
Rovio Entertainment
Espoo, Finland

Perttu Hämäläinen
perttu.hamalainen@aalto.fi
Aalto University
Espoo, Finland

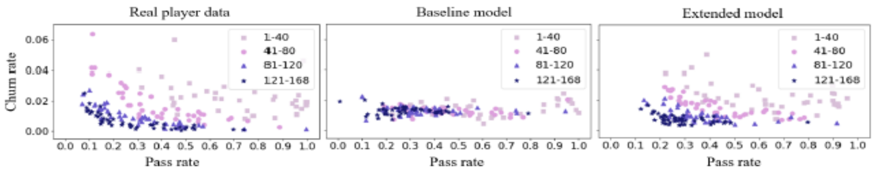


Figure 1: Scatter plots depicting the relation of pass rate (a measure of level difficulty) and churn rate over 168 game levels of Angry Birds Dream Blast, in both real player data and our simulations. Here, churn is defined as not playing for 7 days. The colors denote level numbers. The baseline simulation model predicts pass rate and churn directly from AI gameplay. Our proposed extended model augments this with a simulation of how the player population evolves over the levels.

ABSTRACT

We propose a novel simulation model that is able to predict the per-level churn and pass rates of Angry Birds Dream Blast, a popular mobile free-to-play game. Our primary contribution is to combine AI gameplay using Deep Reinforcement Learning (DRL) with a simulation of how the player population evolves over the levels. The AI players predict level difficulty, which is used to drive a player population model with simulated skill, persistence, and boredom. This allows us to model, e.g., how less persistent and skilled players are more sensitive to high difficulty, and how such players churn early, which makes the player population and the relation between difficulty and churn evolve level by level. Our work demonstrates that player behavior predictions produced by DRL gameplay can be significantly improved by even a very simple population-level simulation of individual player differences, without requiring costly retraining of agents or collecting new DRL gameplay data for each simulated player.

CHI PLAY '20, November 2–4, 2020, Virtual Event, Canada

ACM ISBN 978-1-4503-8074-4/20/11…$15.00
https://doi.org/10.1145/3410404.3414235

CCS CONCEPTS

• **Human-centered computing → User models**; • **Computing methodologies →** *Modeling and simulation.*

KEYWORDS

Player Modeling; Game AI; Churn Prediction

ACM Reference Format:
Shaghayegh Roohi, Asko Relas, Jari Takatalo, Henri Heiskanen, and Perttu Hämäläinen. 2020. Predicting Game Difficulty and Churn Without Players. In *Proceedings of the Annual Symposium on Computer-Human Interaction in Play (CHI PLAY '20), November 2–4, 2020, Virtual Event, Canada.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3410404.3414235

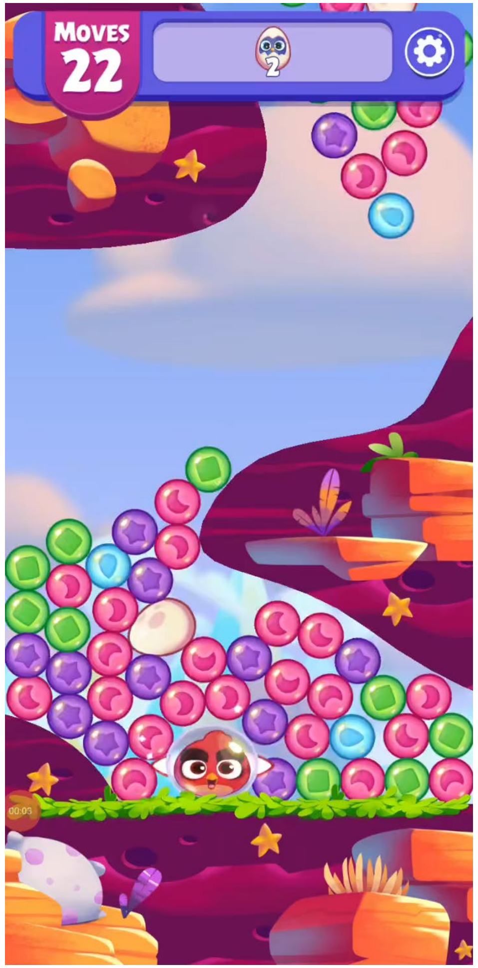
1 INTRODUCTION

One of the primary difficulties of game design and development is that player behavior is hard to predict. This leads to an iterative design process of prototyping and testing, which is slow and expensive. Ideally, research should produce models and tools that allow evaluating the effect of design decisions early on, before committing resources to real-life game testing. This is one of the foundational motivations of player and user modeling [29, 31, 49].

Better models and tools are in particular needed for predicting and optimizing business critical behavior such as churn, i.e., a player quitting the game and not coming back to it. Churn matters as many modern games accumulate their revenue gradually from in-game advertisements and purchases, instead of single up-front fee. To prevent churn, free-to-play game companies engage in extensive

Game: Angry Birds Dream Blast

- **Non-deterministic**, **physics-based**, **free-to-play**, **match-3** game
- **Each level has a specified goal**. e.g., collect x bubbles
- Players can tap **adjacent bubbles with the same color** or use **boosters** to collect bubbles
- Players have **limited numbed of moves** to reach the level goal



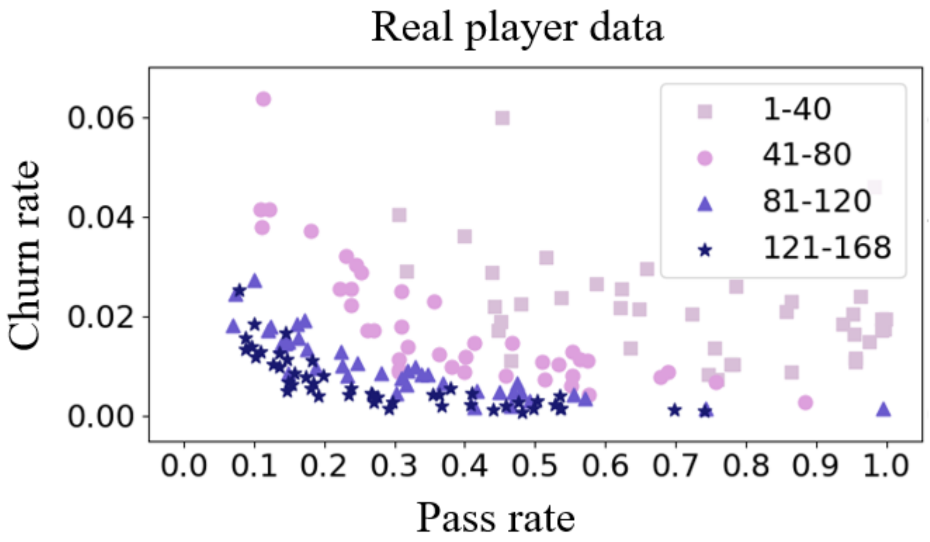
Goal: Predicting Game Difficulty and Engagement

- Data: Pass & churn rates of **168** Angry Birds Dream Blast levels, **95k** players
- Predicting **Game Difficulty** operationalized by **Pass Rate:**

$$\frac{1}{\#Average\ tries\ to\ complete\ level}$$

- Predicting **Engagement** operationalized by **Churn Rate:**

$$\frac{\#Players\ who\ have\ not\ attempted\ the\ level\ for\ at\ least\ 7\ days}{\#All\ players\ who\ have\ reached\ the\ level}$$



Aalto University
School of Science
and Technology

ROVIO

Game-Playing AI (Reinforcement Learning)

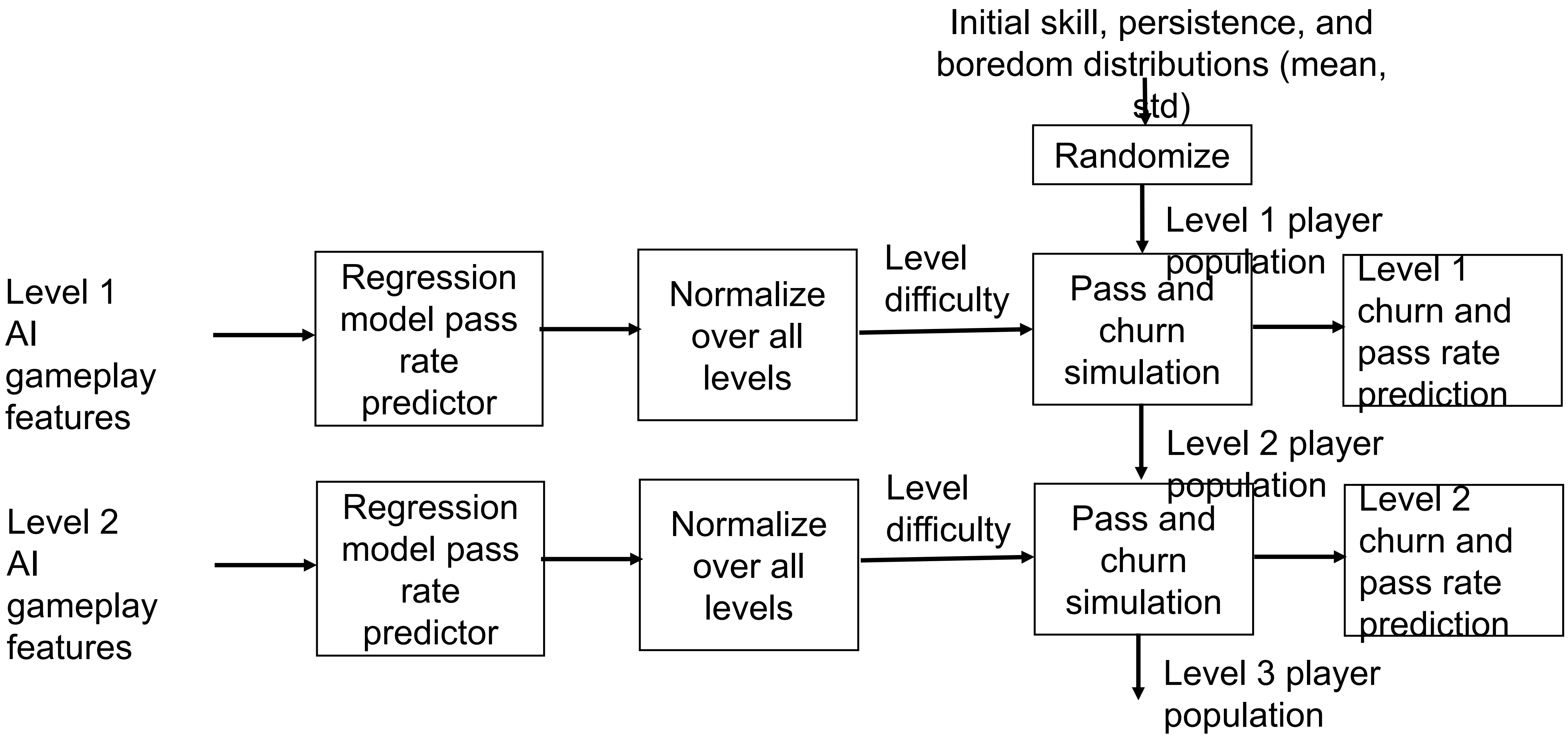
Game

Action

AI agent

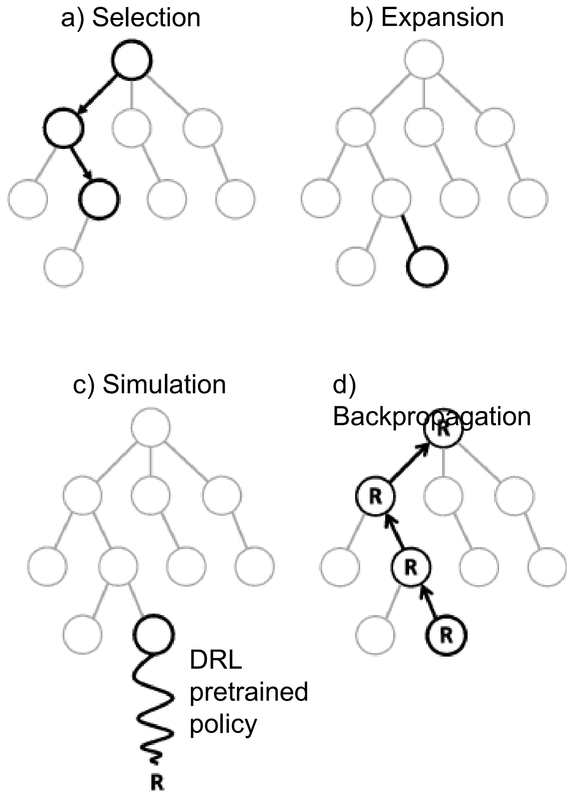
Observation & reward

Previous model: Player Population Simulation



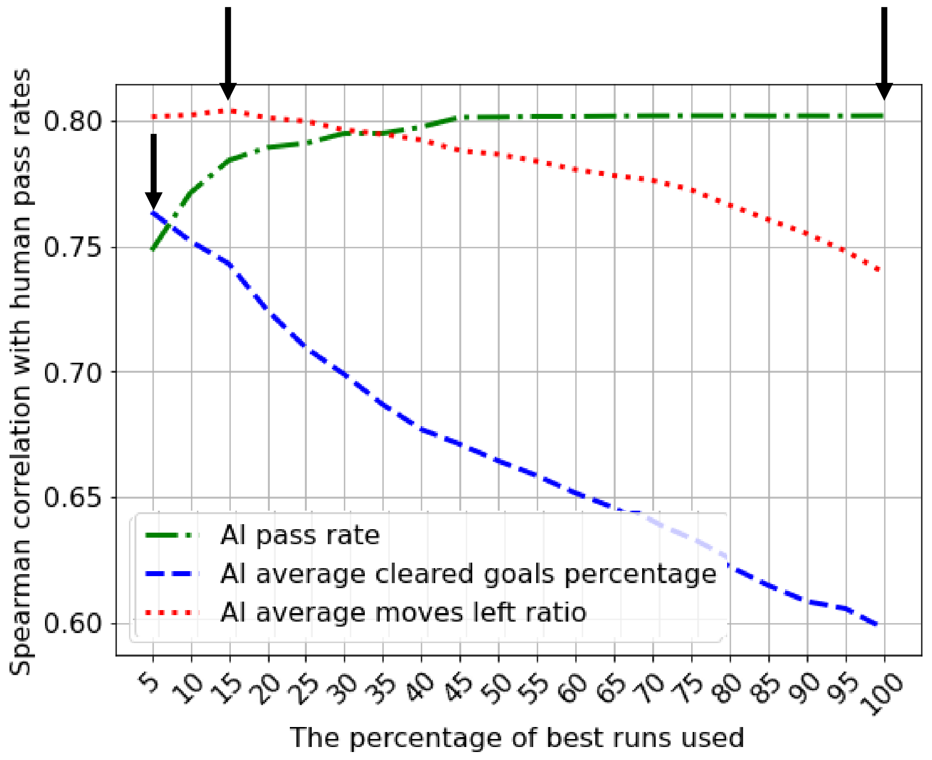
Extension 1: MCTS

- Using **Monte Carlo tree search (MCTS)** for playing the levels
- A forward planning method, it estimates the **optimal action** by building a **tree of possible future game states and rewards**
- MCTS, unlike DRL, **does not require time-consuming training** and can run on an **arbitrary computational budget**
- Use inspired by the popularity of MCTS in **general game-playing** [2, 3] and its recent success in playing difficult games like **Go** [4]
- First to use MCTS for **player behavior and experience modelling**

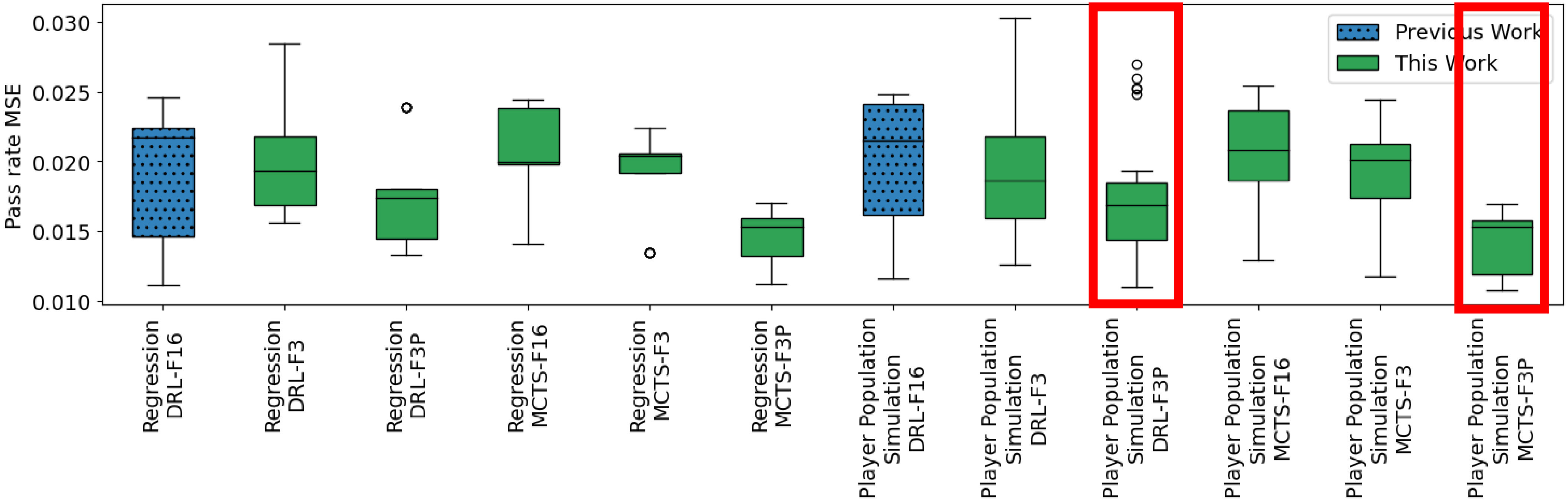


Extension 2: Improved Feature Selection

- **Better feature selection** for our prediction models
- **F3:** Select features with the **highest correlation** with human pass rates
  - AI pass rate, AI average cleared goals percentage, AI average moves left ratio
- **F3P:** Use **the percentage of best runs** with the **highest correlation** with human pass rates

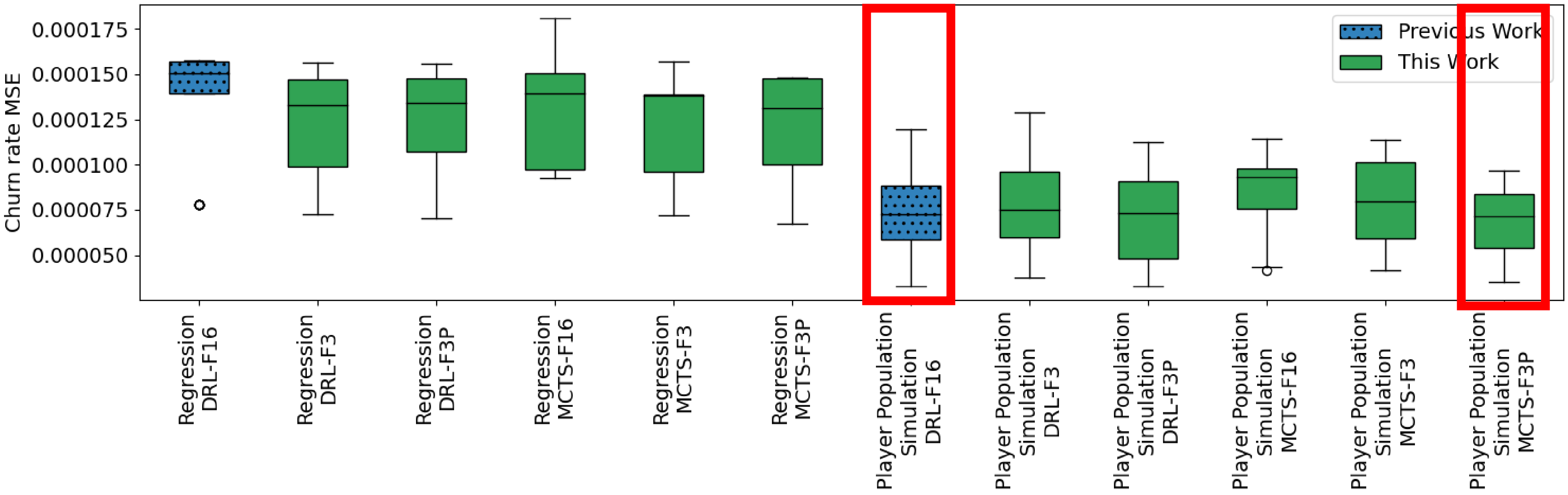


Findings



- Using the DRL best runs features **(F3P)** improves pass rate predictions
- Combining F3 features (highest correlation with human pass rates) from MCTS runs and the F3P features from DRL **(MCTS-F3P)** produces the best results

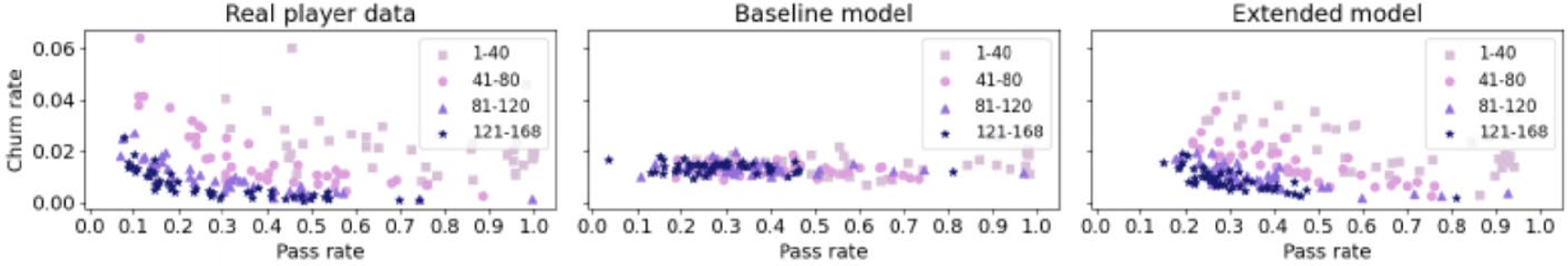
Findings (ctd.)



- **Reproduced the previous work's results**, where the player population simulation method produces better churn rate predictions
- **This work's MCTS+F3P** extension is as good as the best model from previous work

Findings (ctd.)

Roohi et al., 2020. **Baseline** = DRL gameplay features + linear regression; **Extended** model: +population sim



Roohi et al., 2021. **Extended-DRL-F16**: as above. **Extended-MCTS-F3P**: MCTS+DRL+truncated features.

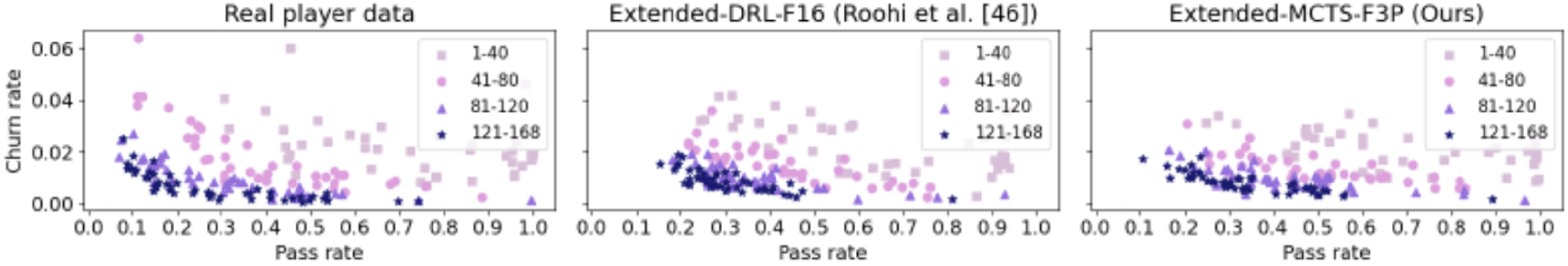


Fig. 5. Human and predicted pass and churn rates over four groups of 168 consecutive game levels, comparing our original [46] best configuration (Extended-DRL-F16) to our best-performing extension (Extended-MCTS-F3P) here. The colors and shapes correspond to level numbers. Our predicted pass rates are slightly less truncated at the lower end.

Conclusion

- **MCTS can improve the predictions of human player data** from the DRL agents
- An **AI agent's best-case performance** can be a **stronger predictor of human player data** than the agent's average performance
- Our improvements in both speed and accuracy **foster the application of automated player modelling** approaches in **games industry**

Additional Material

- **AI based game design patterns**:
  - Treanor et al. (2015). AI-Based Game Design Patterns. Proc. Int. Conf. on the Foundations of Digital Games (FDG), 1–8.
- **Game AI summer school**:
  - Aug 29 - Sept 2, 2022 Greece
  - school.gameaibook.org
- **Game AI Textbook** (Togelius & Jannakakis)
  - Freely available: http://gameaibook.org/
- **AI and Games** YouTube Channel:
  - Industry / research game AI spotlights
  - https://www.youtube.com/user/tthompso
- **Procjam**:
  - ”Make Something that makes something”
  - https://www.procjam.com/

Table 1. An overview of AI-based game design patterns and game examples.

Pattern	What player(s) do	Role of AI (in relation to player)	Example(s)
AI is Visualized	Observe AI state	Gives (strategic) information, showing states	Third Eye Crime
AI as Role-model	Imitate AI	Show agent actionas and behaviors, agents as puzzles	Spy Party
AI as Trainee	Teach AI	Child/student	Black & White
AI is Editable	Edit AI	Artifact/agent that player can author/manipulate	Galactic Arms Race
AI is Guided	Guide/manage the AI	Partly independent inhabitants, with players as their Gods	The Sims
AI as Co-creator	Make artifacts assisted by AI	Co-creator, making artifacts	ViewPoints AI
AI as Adversary	Play game against the opponent	Opponent (symmetric)	Chess, Go
AI as Villain	Combat the Villain(s)	Villain in game; mob, boss mob, NPC (asymmetric)	Alien Isolation
AI as Spectacle	Observe	Spectacle, enacting simulated society	Nowhere

Summary

- **Game AI**: AI with games vs. AI for games
- **AI in Games** vs. **AI for Games**
- **Game AI domains**: (Game-)Playing Agents, Non-Player Characters, Procedural Content Generation, Player Modelling.
- **Industry vs. academia** requirements: speed, robustness, predictability vs. novelty, generality, publication potential…
- **Power in combination**: e.g. user experience modelling + PCG = experience-driven PCG.