Report from Dagstuhl Seminar 22251

Human-Game AI Interaction

Dan Ashlock*†1, Setareh Maghsudi†2, Diego Perez Liebana†3, Pieter Spronck†4, and Manuel Eberhardinger‡5

1 **University of Guelph, CA**
2 **Universität Tübingen, DE.** setareh.maghsudi@uni-tuebingen.de
3 **Queen Mary University of London, GB.** diego.perez@qmul.ac.uk
4 **Tilburg University, NL.** p.spronck@gmail.com
5 **Hochschule der Medien – Stuttgart, DE.** eberhardinger@hdm-stuttgart.de

Abstract

People interact with semi-intelligent machines during their daily lives. They desire systems to respond intelligently to requests. While improvements to the interaction between humans and AI have been made over the years, these systems are a long way from responding like a human partner. Virtual (game) worlds are an ideal environment in which to experiment with the interaction between humans and AI, due to their similarity with real world environments and the presence of agents that represent "real people" that make decisions and interact among them.

In recent years, the number of ways in which players can interact with games have increased considerably: from the traditional mouse, keyboard, and controller, to responding to natural movements, facial expressions, voice, eye movements and brain signals, among others. This seminar brought together scientists, researchers, and industrial developers who specialize in intelligent interaction between humans and computer agents in virtual (game) environments. This report documents the program and its outcomes.

Seminar June 19–24, 2022 – http://www.dagstuhl.de/22251

2012 ACM Subject Classification Computing methodologies → Artificial intelligence; Human-centered computing → Human computer interaction (HCI); Applied computing → Computer games

Keywords and phrases Computational intelligence, artificial intelligence, games, modeling, interaction

Digital Object Identifier 10.4230/DagRep.12.6.28

1 Executive Summary

Pieter Spronck (Tilburg University, NL)
Daniel Ashlock
Setareh Maghsudi (Universität Tübingen, DE)
Diego Perez Liebana (Queen Mary University of London, GB)

Over the past decades, artificial intelligence has evolved from esoteric techniques used mainly in computer science research to an integral and ever-growing part of the daily lives of most humans. People regularly interact with semi-intelligent machines during their daily lives, whether it is via smartphone applications, embedded systems in cars and household electronics,

* in memoriam; † April 5, 2022
† Editor / Organizer
‡ Editorial Assistant / Collector

client support systems, or helpful technology installed on personal computers. People wish and expect systems to respond intelligently to their requests, and even to anticipate their actions. While improvements to the interaction between humans and intelligent systems in this respect have been made over the years, there is still a long way to go before these systems exhibit a level of understanding and intuition which can be expected from a human partner.

Human-computer interfaces (HCI) are a well-established scientific research domain. We noted that HCI research generally neglects the use of artificial intelligence as a integral part of an interface. Almost any person that uses computers can quickly recall multiple frustrating interactions with the current state of the art in artificial intelligence in interfaces. Since annoyance and apparent incompetence can derail the adoption of otherwise promising and potentially transformative technology, research into improving interfaces using AI is timely.

An AI assistant is broadly recognized as being a key factor in increasing human productivity, but it must be an AI assistant that the user either enjoys working with or that the user barely notices, not one that must be bludgeoned into useful behavior or constantly fought with. Perfection of assistants, companions, and even opponents that correctly anticipate and collaborate in the relatively controlled domain of games provides a smooth path to such developments in broader contexts.

We argue that virtual worlds, as found in computer games, are an ideal environment in which to experiment with the interaction between humans and artificial intelligence. There are at least three reasons for this. First, virtual worlds often approach the complexity of the "real world", while still being under the control of the researcher and completely observable. Second, the agents in virtual worlds are supposed to represent "real people" and are approached as such by the humans who "play" with the virtual world. Third, the potential interactions that players have with the virtual worlds are highly diverse and wide-ranging, which presents a substantial challenge for artificial intelligence to respond to in a reasonable fashion.

In recent years, the number of ways in which human players can interact with games have increased considerably. While ten years ago interaction was almost exclusively through mouse and keyboard or controllers, nowadays games can potentially respond to natural movements and facial expressions captured by a camera, to spoken language, to eye movements, and to signals captured by a variety of sensors. Brain-computer interface (BCI) technology has become more mainstream, offering possibilities for games to respond to a users' brain activity. Using VR technology, games can respond to movements of players in natural space. AI that can use all these interface elements to make game agents a natural and appreciated partner or opponent for humans can form the basis for advanced AI agents that interact with humans not only in games, but also in the real world.

The research area in which the proposed seminar is rooted is the interaction between humans and game AI, aiming for natural and appropriate responses of computational agents in virtual worlds to human behavior, making use of both traditional interaction technology as well as modern sensor and interaction technology.

The research area lends itself for a wide range of research topics. For the preparation of this seminar, we proposed the following set of sub-topics:

- **Personalized Human-Game AI Interaction:** Humans have different backgrounds, interests, and goals. As such, there is no "one-size-fits-all" interference and interaction form. Under this topic, we explore game adaptation as a type of automatic game design. The goal is to permit the AI to adapt the game environment to the player based on the observed features and received feedback. Instead of fully automatic game design, a sophisticated game design leaves scope for an AI to adapt to a broad variety of players. Such personalized adaptation could be extended to adaptation of the actual game interface – in games, usually complex interactions are possible, which novice players are not capable of employing. Therefore, automatically adapting the interface to the observed experience level of the player may be a valid approach to effective personalization.

- **Human-Game AI Interaction for People with Disabilities:** People with disabilities require special attention when designing interfaces, to mitigate adverse effects of disabilities, so that a suitable experience is ensured for everyone. Game AI can potentially help to diagnose disabilities, both physically and psychologically. There is also the potential for game AI to create awareness of issues faced by those with disabilities, by intelligently adapting the interface in such a way that the player experiences it as a person with disabilities would.
- **Multimodal Interfacing and Interaction:** Multimodal systems offer a flexible and efficient interaction environment that consists of several input/output possibilities including text, speech, and vision. How to effectively use these possibilities in game design is still an open problem. A compelling application of artificial intelligence is to rapidly learn which modes a given player finds natural and enjoyable. The type of interface a user is comfortable with is likely to cross boundaries between different applications, meaning that an "interface fingerprint" may be derivable that can be carried with the user, permitting the re-use of information gained.
- **Enhancing Human Creativity with Artificial Intelligence:** Computational Creativity is a field of AI where automatic AI systems design and create various forms of art, which may include images, drawings, poetry and music. In the broader sense, these systems create new content either completely by themselves, or with the human providing input at specific points. Research into this fusion of the creative skills of humans and AI systems would move the state of the art a step forward: from being inferior content creators the AI systems would become a tool for amplifying and augmenting the superior creative abilities of a human being, in a bi-directional collaboration process. AI systems should be able to learn from the human, anticipate what they intend to do, and understand the domain of discourse. They would provide advice on content creation and help when the user struggles with certain techniques or creative methods. By learning the skills of the human, AI systems would be able to propose alternatives that lie outside their expertise, allowing the humans to learn, refine and improve their capabilities. The users would experience a system that adapts to their skills, needs and pace, and becomes a personalized companion in their learning process.
- **Trustful and Reliable Human-Game AI Interaction:** We often observe that humans feel uncomfortable with AI recommendations. Moreover, mistakes made by humans are deemed more tolerable than those made by an AI. While there is no objective rationale for this difference, it is hard to justify the use of AI for humans by arguing that AI offers a lower mistake probability compared to humans. It is therefore imperative to find new ways to convince humans to interact with the game AI and to take its advice seriously. Moreover, it is crucial to minimize any effect that might harm such trust, regardless of its origin.
- **Information Flow in Human-Game AI Interaction:** A game AI must observe the human player and, in turn, provide players with information that they find helpful, valuable, or interesting. Even the most potentially helpful information is not actually helpful if the player cannot understand it or if it is not useful to their particular style of play. The flow of information is particularly important between the human player and an

AI companion. Reliable metrics that ascertain if the human uses information offered by the AI, that check if the AI fails to provide information that the human tries to find in other ways, and assessment of defects in the human's play that suggest which information is needed, are potential goals of research in this area.

- **Believable Human-Game AI Interaction:** In the last decade, contests have been held at several conferences where human judges voted on the "humanity" of both human game players and AI players in an effort to score the ability of the AI players to behave in a plausibly human manner. Attempts to make AIs interact in a way that is indistinguishable from human interaction are a natural way to structure research into human-game AI interaction. We note that the believability of game AI often suffers because it fails to recognize that it misunderstands the human player, or that the human player misunderstands the AI. How to recognize misunderstanding, followed by how to correct for misunderstanding, are important steps in making game AI more believable.
- **Ethics of Human-Game AI Interaction:** Several of the aforementioned research directions rely heavily on big data analysis. Acquiring such a massive amount of data is a challenging task. Perfect anonymization is hard to achieve, and often undesirable as multiple parties are involved in data collection and integration. To what extent is it ethical to collect personal interaction information? Are there ethical restrictions to the extent to which an AI is allowed to analyze a player's personality and demographics? These questions need answering even if a player gives permission to collect and use such data.
- **Novel Forms of Interaction and Interfaces in Game AI:** New technology gives rise to new possibilities in game interaction and interfacing. While developers often try to restrict themselves to small adaptations in tried-and-true forms of interaction, it makes sense to consider the interaction possibilities originating with novel technology, such as virtual reality and brain-computer interfacing. Beyond those, there may be ways for humans and AI to interact with each other that has not yet been imagined, or which can benefit from re-imagining. Player-AI interaction can be implemented in many forms, such as (1) cuing a player with environmental information from music to decor, (2) influencing a player by adjusting game elements such as local architecture, opponents, and rewards, and (3) making a player respond to the social tone of non-player characters. Such alternate forms of player-AI interaction warrant investigation.

This seminar was organized around workgroups, which worked in teams and topics proposed by the participants of the seminar in the areas outlined above. These workgroups were accompanied by plenary sessions for group formation, topic debate and discussions of the deliberation of each group. Workgroups were dynamic, so participants could move between them, and new groups were formed during the week. A Discord server was setup for coordination and announcements, and it was also used by the different groups for document and link sharing. This also has the benefit of providing a place for discussions after the seminar, easing the communication and further work among the members of each workgroup.

It is worthwhile mentioning the work carried out during the invitation process. Due to the COVID crisis, the changes in the political landscape, and the war in Ukraine, many declined the invitation, and many participants dropped out after originally having accepted the invitation. Thus, multiple rounds of invitations were run until two weeks before the seminar. We invited close to 100 people, the full list of invitations having a high diversity (a 50% male-female split, about half invitations for 'junior' people, and invitees hailing from all continents – including South America and Africa, which are usually highly underrepresented). In the end, just over 30 participants attended the seminar (out of the 45 possible). Size-wise this was a slight disappointment. We were fortunate, however, that those that did attend were highly enthusiastic and highly knowledgeable about the topics covered, which made the seminar a great success.

2 Table of Contents

3 Working groups

3.1 Language Models for Procedural Content Generation

Maren Awiszus (Leibniz Universität Hannover, DE), Alexander Dockhorn (Leibniz Universität Hannover, DE), Amy K. Hoover (New Jersey Institute of Technology, US), Antonios Liapis (University of Malta – Msida, MT), Simon M. Lucas (Queen Mary University of London, GB), Mirjam Palosaari Eladhari (Stockholm University, SE), Jacob Schrum (Southwestern University – Georgetown, US), and Vanessa Volz (modl.ai – Copenhagen, DK)

3.1.1 Introduction and Motivation

Recent advances in ML-based image generation via systems like DALL-E [6] beg the question of whether similar tools can be used for generating game content. Specifically, it is desirable to generate game content based on simple text input. As our group was composed of researchers most familiar with level generation, we focused on video game level generation for 2D games first. However, other assets like textures also seem like great examples to use these generative methods on. Our general intuition was, that methods like DALL-E are able to generate impressive previously unseen images due to the strength of the diverse language processing learned with huge datasets of images and their descriptions. For games, in which only little data can be given for a domain like e.g. 2D Super Mario levels, such a large network can not easily be trained from scratch. Therefore, we wanted to investigate the capabilities of a pretrained DALL-E to generate content without any game specific training.

3.1.2 Exploration of Applications

We ran tests using **DALL-E mini** [3] as an intermediate tool for generating content. Figure 1 shows some of those generated examples. Although DALL-E mini can create outputs that look like Mario levels given a prompt like "Mario level", it has problems incorporating specific details suggested from prompts such as "spikes" or "pipes" in "Mario level with a spike pit" or "Mario level with pipes". The problem seems to be that DALL-E's concept of what spikes or pipes are is based on typical photo examples of these items rather than examples of these items in the context of a Mario game. However, prompts that only change the style of the level, like "apocalyptic", can influence the output. We also did some small preliminary tests on content other than platformer levels, like generating images of new "Pokémon" from a textual description. These examples suffer from similar problems. DALL-E can generate a "Pokémon", but adding additional descriptors is less likely to be successful. It will be interesting for future work to find out what kind of prompts can and cannot be mixed with DALL-E and why. Especially if this is a tool to be used by game designers, one needs to make sure that the method does not ignore additional descriptors that are a crucial part of the game's design.

The model **CLIP** [5], which is part of the DALL-E pipeline, can also be used on its own to gauge how well a text description matches an image. We tried matching the images of some original Mario levels to certain prompts that describe a level in more detail, like "under ground Mario level". For that, we used images of Mario levels provided by the Video Game Level Corpus (VGLC) [7]. The results indicate, that while CLIP does seem to distinguish

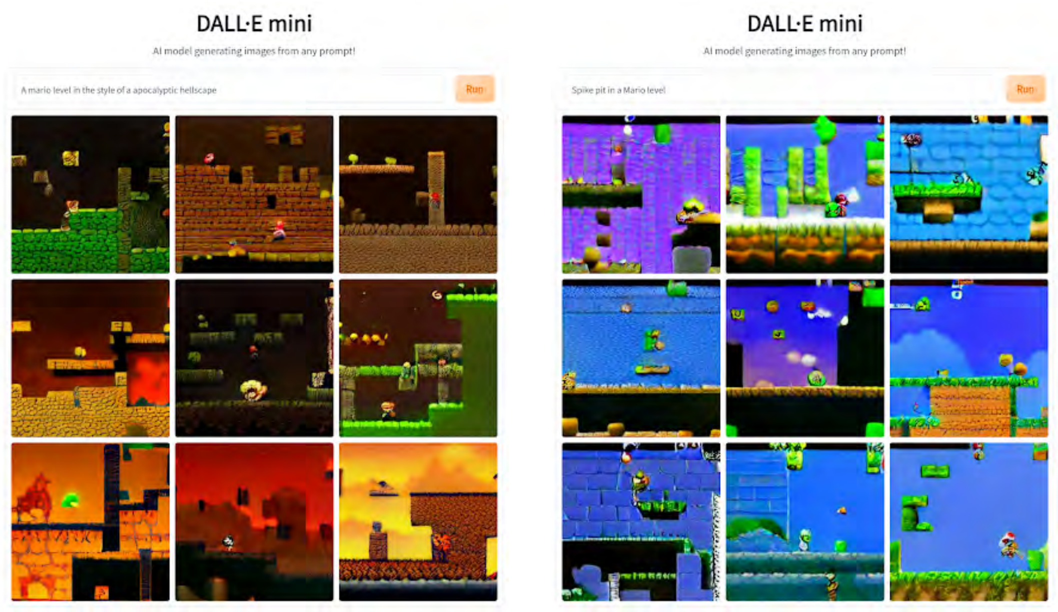

DALL·E mini
AI model generating images from any prompt!
A mario level in the style of a apocalyptic hellscape
Run
DALL·E mini
AI model generating images from any prompt!
Spike pit in a Mario level
Run

Figure 1 Examples of images generated with DALL-E Mini [3]. The generated examples look like Mario levels and certain prompts, such as "apocalyptic" can change the style of the generated images. However, prompts like "spike pit" are ignored.

between "over ground" and "under ground" levels, objects such as pipes do not seem to be recognized as well, which likely explains why pipes cannot be easily generated either. Note, that these results are also without fine-tuning the model at all.

The results of these experiments indicate, that even without any fine-tuning of DALL-E and CLIP, the methods already show some understanding of video game levels. We identify creating a small data set of levels and their appropriate textual descriptions for fine-tuning as an important task to further research in this direction.

3.1.3 A Functional Pipeline

While creating images of levels with DALL-E can indicate whether or not the method can be used for level generation in general, this neglects the problem of creating a playable level from that image. Therefore, we established a prototype pipeline for getting playable levels from text, which is shown in Figure 2. Text can be sent to DALL-E to create a level image. Preexisting tools can derive a structured level segment from the image. For now, only the naive RGB tile matching method provided in [2] is applied to this task. Finally, a larger, more complete level can be made from that segment with TOAD-GAN [1]. As the example of a Mario level snippet in Figure 2 shows, the preliminary pipeline works and can create playable Mario level snippets from DALL-E mini. From here, the pipeline needs to be completed by implementing other options to create a tile map from an image, as well as assembling the implemented parts of the pipeline into one cohesive system.

3.1.4 Conclusion and Future Work

In this workgroup, we investigated the possibility of using current text to image methods like DALL-E for video game content generation. We show promising results for Super Mario level generation while identifying problems of the method ignoring certain prompts that

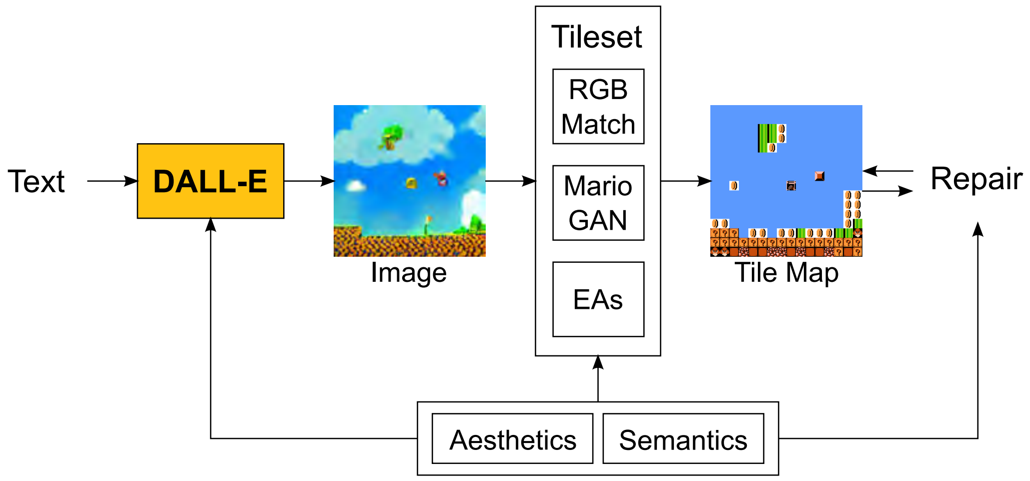


Figure 2 The pipeline established at the end of the group session. As shown with the examples, we implemented the pipeline up to the point of being able to generate a tile map from a text prompt. The image is turned into the tile map with an RGB matching algorithm based on [2].

might be important for a game designer. Additionally, we tested if CLIP, a part of DALL-E, can match certain prompts with given Mario level images, and find a similar result: That it can only distinguish some prompts and might ignore others. This however, is only using the pretrained models as is, and we pose that fine-tuning will improve the results for both experiments. We also established a functional text to level pipeline, which can turn a text prompt into a playable Mario level snippet.

For future work, there are two distinct goals: creating a data set to allow for fine-tuning a pretrained DALL-E and completing the missing pieces of the pipeline. For the data set, detailed descriptions of level images need to be found or created and a way to convert them into a usable format needs to be found. Also, other kinds of data sets that deal with assets other than levels can be explored, like texture images. The missing pieces of the pipeline include include other tile set representations that generate a tile map from an image, such as Generative Adversarial Networks and Evolutionary Algorithms, and using the Tile-Pattern KL-Divergence [4] as a repair mechanism for the tile maps. Also, the currently still fragmented pieces need to be combined to form one cohesive system for ease of use.

References

1 Maren Awiszus, Frederik Schubert, and Bodo Rosenhahn. *Toad-gan: Coherent style level generation from a single example*. In Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, 2020

2 Eugene Chen, Christoph Sydora, Brad Burega, Anmol Mahajan, Abdullah Abdullah, Matthew Gallivan, and Matthew Guzdial. *Image-to-level: Generation and repair*. In Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, volume 16, pages 189–195, 2020

3 Boris Dayma, Suraj Patil, Pedro Cuenca, Khalid Saifullah, Tanishq Abraham, Phúc Lê Khac, Luke Melas, and Ritobrata Ghosh. *Dall-e mini*, 7 2021

4 Simon M Lucas and Vanessa Volz. *Tile pattern KL-divergence for analysing and evolving game levels*. In Proceedings of the Genetic and Evolutionary Computation Conference, pages 170–178, 2019

5 Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. *Learning trans-*

ferable visual models from natural language supervision. In Proceedings of the International Conference on Machine Learning, pages 8748–8763. PMLR, 2021
6 Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. *Zero-shot text-to-image generation*. ArXiv preprint, abs/2102.12092, 2021
7 Adam James Summerville, Sam Snodgrass, Michael Mateas, and Santiago Onta n'on Villar. *The vglc: The video game level corpus*. Proceedings of the 7th Workshop on Procedural Content Generation, 2016

3.2 AI for Romantic comedies

Michael Cook (Queen Mary University of London, GB), Maren Awiszus (Leibniz Universität Hannover, DE), Duygu Cakmak (Creative Assembly – Horsham, GB), Alena Denisova (University of York, GB), Alexander Dockhorn (Leibniz Universität Hannover, DE), Casper Harteveld (Northeastern University – Boston, US), Antonios Liapis (University of Malta – Msida, MT), Mirjam Palosaari Eladhari (Stockholm University, SE), Diego Perez Liebana (Queen Mary University of London, GB), Lisa Rombout (Tilburg University, NL), and Tommy Thompson (AI and Games – London, GB)

3.2.1 Introduction and Work Process

Both romance and comedy are integral parts of human culture, yet despite the breadth of AI research into games and creativity, little work has been done to explore these themes in the context of games. In AI research, the best examples are games that deal with 'social physics' or human relationships, such as *Prom Week* [4] or *Façade* [3], where both romantic and comedic themes are hinted at. In the games industry, while romance is a key feature in many games (such as *The Sims*), it is often reduced to static linear narratives, while comedy is notoriously difficult to achieve in games and is often achieved unintentionally [2].

In this workgroup, we aimed to explore the possibility that these two things are connected. Due to a lack of AI research into topics such as romance and comedy, there are fewer systems and techniques available to support the exploration of these themes in game design. Our workgroup aimed to explore the potential for AI research in these areas, to think about the open questions and pitfalls ahead, and to collaboratively sketch out some ideas for work that we could act as inspiring examples for future AI research projects. The group began with a short presentation, including a series of tweets from @NightlingBug on Twitter, who made an observation that playing a game such as *Stardew Valley* from the perspective of a character competing for the player's attention would be an interesting idea.

We began with an open discussion of the topic, encouraging perspectives from everyone present, covering both existing examples of technology and games, as well as concerns, questions, and ideas that arose as we thought about the topic. All of the topics that came out of this discussion were interesting and thought-provoking, but a few ideas stood out as something the groups were particularly excited to take forward during the day. The first was the idea of connecting existing AI narrative techniques, such as the Nemesis system in *Shadow of Mordor* [7] to large-group dynamics like the romantic NPCs in *Stardew Valley*. The second idea was to think about how information flow is often crucial in romantic stories, both within the fiction and between the reader and the author. The third was to investigate unusual concepts such as discomfort, embarrassment, or "cringe" as a component of a narrative or social AI system. The workgroup split into three subgroups to explore these ideas separately, before reconvening at the end of the day.

3.2.2 Nemesis Island

The first group proposed an AI-driven spectator sport based on popular reality TV franchises such as *Love Island.* In their prototype, a network of AI agents compete both for the romantic attentions of other AI agents, and the real-world attention of people viewing the game on livestream services, such as Twitch. As a third role, a director can be introduced, whose task it is to steer the narrative by setting hidden internal goals for each agent. Agents respond to the internal social network of the game, the pursuit of their internal goals, as well as their meta-level understanding of the show they are in, intentionally creating drama or showing off to create interest in the audience, in the hope that they will survive rounds of voting and elimination.

3.2.3 JANE (Judicious Artificial Narrator Experience)

The second group proposed a game inspired by Jane Austen's use of free indirect discourse [1], where the author disseminates information to the reader that could be biased by a particular viewpoint, or actual narrative fact [6]. In this approach, the reader always only has partial (and potentially misleading) information on the characters, and they about each other – which leads to both romantic and comedic situations. The setting for this game could be based on shows such as *Bridgerton* or *Gossip Girl.* The player takes the role of a pseudonymous gossip columnist, who must explore and learn about high society by attending events, engaging in gossip, and dealing favours. The columns written by the player impact the knowledge and social simulation of AI socialites, which in turn changes the situations the player finds themselves in. This creates a kind of participatory take on social simulations like *Bad News* [5], with the added complication of allowing the player to engage in high society themselves, potentially manipulating the social scene to help them achieve their personal goals.

3.2.4 #CringeFestival

The third group considered the role of embarrassment and negative emotions in romantic comedies. One issue that came up in our initial discussions was understanding the role of the player in such games. As the audience for a romantic comedy, we have a distance between us and the actions of the characters ("cringe" is defined as experiencing embarrassment on behalf of someone else). If the player is participating as a character then they might feel closer to the negative experience. This group explored the idea of games in which the player acts as an external force, either trying to set up artificially embarrassing moments for AI agents, or acting to save and rescue AI agents from embarrassing situations to gain catharsis.

3.2.5 Conclusion and Outcomes

Our group discussions have yielded a number of new directions to explore, both in terms of prototyping new systems, as well as exploring the affordances and applications of existing technology. We are hoping to pursue some of these ideas a little further and write the results up, and to continue to maintain the working group as an ongoing collaboration.

References

1 Jane Austen. *The complete novels of Jane Austen*, volume 4. Chartwell Books, 2016.
2 Claire Dormann and Robert Biddle. Making players laugh: The value of humour in computer games. In *Proceedings of the 2007 conference on Future Play*, pages 249–250, 2007.
3 Michael Mateas and Andrew Stern. Procedural authorship: A case-study of the interactive drama Façade. In *Digital Arts and Culture*, 2005.
4 Josh McCoy, Mike Treanor, Ben Samuel, Aaron A. Reed, Michael Mateas, and Noah Wardrip-Fruin. Prom Week: Designing past the game/story dilemma. In *Proceedings of the Foundations of Digital Games Conference*, 2013.
5 Ben Samuel, James Ryan, Adam Summerville, Michael Matea, and Noah Wardrip-Fruin. Bad news: An experiment in computationally assisted performance. In *Proceedings of the International Conference on Interactive Digital Storytelling*, 2016.
6 Carmen Smith and Laura Mooneyham White. Discerning voice through austen said: Free indirect discourse, coding, and interpretive (un) certainty. *Persuasions: The Jane Austen Journal On-Line*, 37(1), 2016.
7 Ryan Taljonick. Shadow of Mordor's Nemesis system is amazing–here's how it works. `https://www.gamesradar.com/shadow-mordor-nemesis-system-amazing-how-works/`, 2014. accessed 3 July 2022.

3.3 Pokegen

Alexander Dockhorn (Leibniz Universität Hannover, DE), Manuel Eberhardinger (Hochschule der Medien – Stuttgart, DE), Daniele Loiacono (Polytechnic University of Milan, IT), Diego Perez Liebana (Queen Mary University of London, GB), and Remco Veltkamp (Utrecht University, NL)

The generation of art assets plays a huge part in game development, costing both time and money. We explored how the process of generating game art can be supported using recent advances in generative art.

Machine learning models such as Dall-E 2 [1] and Imagen [2] have demonstrated powerful art generation capabilities. Starting from text prompts, they are able to combine concepts, attributes, and styles to generate artworks of generally high quality. Nevertheless, their usage is restricted and similar projects such as ruDall-E [4] and Mini-Dalle-E [3] do not produce results at the same level of detail, i.e. generating blurry images, struggling to include concepts that are not well represented in the training data, and sometimes creating stock image overlays (c.f. Figure 3). This often results in prompt engineering, a process in which the user adapts the text prompt to guide the black box model to produce the desired outcome [8]. Due to the black-box nature of deep learning models, this process can yield unstable results and is therefore hard to control, making it inefficient and unreliable for creating game assets.

Therefore, we have envisaged several pipelines that may support designers and artists during game development. Starting from possible inputs such as a designer's textual descriptions of the required game asset, some image ideas, sketches, or even expected game mechanics, we have multiple ways to approach the problem of game asset generation. Simple text and image search models may guide the artistic exploration process and spawn new ideas.

Nevertheless, those can only return results that already exist. Given textual descriptions, we can apply text-to-image models for generating new assets. Alternatively, we may use style-transfer models to enforce characteristics described in the text to an existing image (e.g. CycleGAN [9]). The latter may also be used to adjust image characteristics such as drawing style or the choice of colors (e.g. Neural Style Transfer [10]). Especially interesting is the combination of such models, which may allow to tune each component of the processing chain separately.

In our working group, we have worked on implementing a toolchain to generate Pokemon-like creatures. A Pokemon often represents an animal or object and in terms of visual style, does only consist of a few colors as well as simple shapes and textures. Aiming to use existing models without retraining, we started our process by generating images of dragons using ruDall-E [4]. Generated images varied hugely in quality. Since due to our style constraints our final image does not need to include a lot of details we have chosen a rather blurry image of a dragon with a simple background. Having selected a generated image of a dragon we applied style-transfer as a combination of VQGAN [7] and CLIP [6]. Without retraining any of these components to our specific domain (due to time constraints), we were unable to achieve results of high visual quality (see Figure 4). Nevertheless, this process show-cases how mock-ups and ideas may be generated to guide the development process.

While having struggled to develop a multi-stage model for generating Pokemon-like creatures, it has helped us to better understand the main challenges for generating game assets in general. The following challenges have been identified by us and may guide further research in this domain:

- **Copyright:** Generating art from machine learning models poses the question of who owns the copyright of the final result. This may be a complicated question to answer since the result itself is likely to be a product of an enormous training corpus on which the model is based and the user's input. While there is no definitive answer to this question yet, the current suggestion seems to be an evaluation on a case-by-case basis [5].
- **Training data:** Depending on the stage of production, the amount of available training data may be minor in comparison to the variety of elements that need to be generated. Especially in the early stages of development, machine learning models may merely be used to generate interesting mock-ups or explore ideas. Later development stages may allow to train specialized models or refine existing models to produce desired results.
- **Costs:** Creating your own machine learning models or using the models provided by others can come with non-negligible costs. The required hardware, energy, and time for training and inference should be kept in mind while planning a pipeline. Reducing these costs is already a key aspect of machine learning research and further advancements may considerably reduce the related costs.
- **Usability and Explainability:** Each of the envisaged pipelines comes with its own unique challenges. Especially, the usability of black box models may become a problem in case the input space is not well understood. We have tackled this problem by splitting the asset generation into multiple sub-tasks which we were able to control independently with limited success. Better explaining a model's relation between in- and output as well as its parameter space may help in increasing the usability of such models.

While there are still many steps ahead of us, supporting the generation of game assets using machine learning models may have huge impact on the field. At the current stage, existing models may already be used to support the prototyping stage or generate mock-ups and ideas for the human-guided generation process. Having further advanced on the models' capabilities, it may be possible to learn from just a few examples and produce game assets

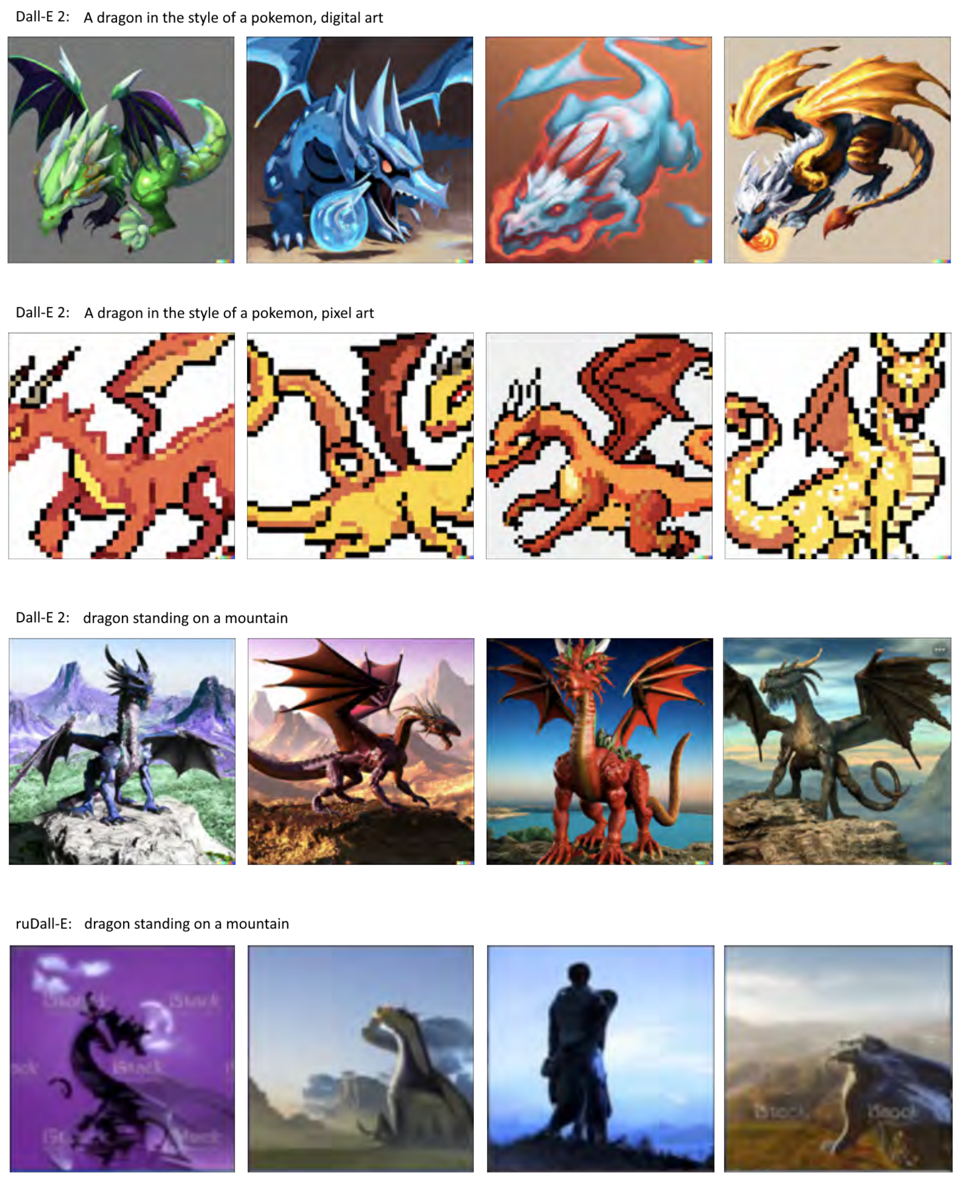

Dall-E 2: A dragon in the style of a pokemon, digital art
Dall-E 2: A dragon in the style of a pokemon, pixel art
Dall-E 2: dragon standing on a mountain
ruDall-E: dragon standing on a mountain

Figure 3 Comparison of Dall-E 2 and and the openly accessible ruDall-E for generating dragons and pokemon-like creatures. Dall-E 2 is able to produce images of higher quality. However, it requires an invitation from OpenAI to be used.

of matching styles. In the long run, combinations of machine learning models may even guide the development of whole game worlds and game mechanics, allowing us to generate complete game experiences given a user's queries.

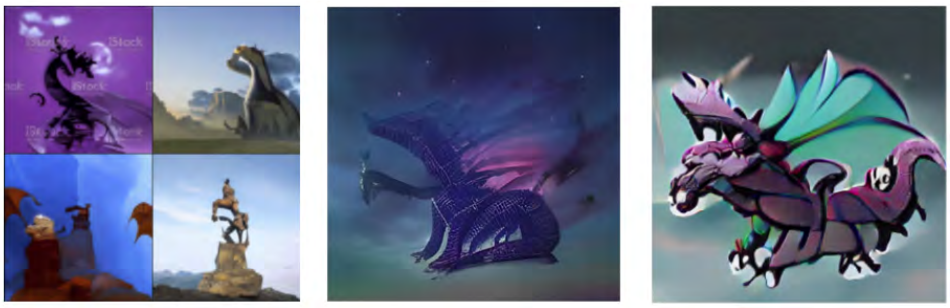


Figure 4 Demo pipeline for generating Pokemon-like creatures. First, generating images of dragons using ruDall-E[4] (left) discarded examples, (middle) chosen example, (right) given the text prompt "A dragon in the style of a pokemon" we used VQGAN [7] and CLIP [6] to produce the final result.

References

1 Ramesh, A., Dhariwal, P., Nichol, A., Chu, C. & Chen, M. Hierarchical Text-Conditional Image Generation with CLIP Latents. (arXiv,2022), https://arxiv.org/abs/2204.06125

2 Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E., Ghasemipour, S., Ayan, B., Mahdavi, S., Lopes, R., Salimans, T., Ho, J., Fleet, D. & Norouzi, M. Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. (arXiv,2022), https://arxiv.org/abs/2205.11487

3 Dayma, B., Patil, S., Cuenca, P., Saifullah, K., Abraham, T., Lê Khâc, P., Melas, L. & Ghosh, R. DALL E Mini. (2021,7), https://github.com/borisdayma/dalle-mini

4 Shonenkov, A. ruDall-E. (2021), https://pypi.org/project/rudalle/

5 Chiou, T. Copyright lessons on Machine Learning: what impact on algorithmic art?. (2019)

6 Radford, A., Kim, J., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G. & Sutskever, I. Learning Transferable Visual Models From Natural Language Supervision. (arXiv,2021), https://arxiv.org/abs/2103.00020

7 Esser, P., Rombach, R. & Ommer, B. Taming Transformers for High-Resolution Image Synthesis. (2020)

8 Liu, V. & Chilton, L. Design Guidelines for Prompt Engineering Text-to-Image Generative Models. *CHI Conference On Human Factors In Computing Systems*. (2022)

9 Zhu, J., Park, T., Isola, P. & Efros, A. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. *Computer Vision (ICCV), 2017 IEEE International Conference On*. (2017)

10 Gatys, L., Ecker, A. & Bethge, M. Image Style Transfer Using Convolutional Neural Networks. *Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition (CVPR)*. (2016,6)

3.4 Program Synthesis for Explaining Strategies

Manuel Eberhardinger (Hochschule der Medien – Stuttgart, DE), Cameron Browne (Maastricht University, NL), Jakob Foerster (University of Oxford, GB), Daniele Loiacono (Polytechnic University of Milan, IT), Ana Matran-Fernandez (University of Essex – Colchester, GB), and Remco Veltkamp (Utrecht University, NL)

3.4.1 Introduction & Motivation

Artificial intelligence (AI) in games has attracted a lot of public attention by defeating world champions in board games such as Go or Chess [1, 2]. In eSports, OpenAI trained multiple agents simultaneously that defeated the world champion team in Dota 2, a real-time strategy multiplayer online game where two teams of five members compete against each other [3]. Additionally, DeepMind developed an AI model called AlphaStar that defeated the world champion in the real-time strategy game StarCraft II [4].

Nevertheless, most strategies of AI models are not explainable or interpretable because a trained neural network is a black box or the search space is too large. This makes it difficult for humans to follow the path of the agent as it traverses the search tree to choose the best action. In this work, we are investigating new ways to make agent behavior interpretable by using program synthesis to explain strategies of agents by distilling black-box policies into programmatic policies.

The rest of this abstract gives a brief introduction to program synthesis for generating programmatic policies and unsupervised environment design (UED), a new approach to providing agents with increasingly difficult environments to create a curriculum for the agent. We conclude this abstract by proposing a method how to combine UED with program synthesis to make game strategies interpretable.

3.4.2 Program Synthesis & Programmatic Policies

In recent years, more and more work investigated program synthesis in reinforcement learning to create programmatic policies for making agent behavior in games or other environments interpretable [5, 6, 7]. Most methods use a form of imitation learning by trying to imitate the behavior of an oracle such as a neural network policy or a human demonstrator. The generation of programs is only possible if the domain-specific language (DSL) is specified and adapted to the task at hand. If the DSL is too general, like a normal programming language, the search for programs is not feasible and leads to no program being found at all [5]. Another way to increase the chances of finding a correct program is to reduce the search space by providing sketches of the program structure where only the missing gaps need to be filled [6]. The problem of finding programmatic policies without defining a task-specific DSL or given prior knowledge about the structure of the program is still an open problem.

However, recent work showed that it is possible to learn a library of functions from previously solved problems. These functions are then reusable in an updated DSL to solve more difficult problems [8, 9]. This leads to a form of curriculum learning by the agent, similar to self-play, as the agent is able to find programs for problems it could not solve before.

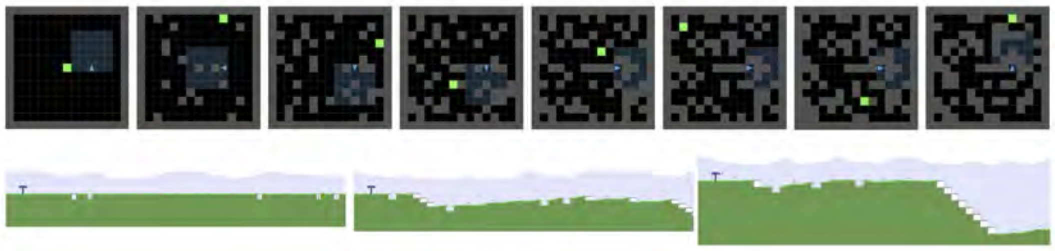


Figure 5 Two examples of evolving environments with increasingly difficult levels using the ACCEL algorithm (from [11]).

3.4.3 Unsupervised Environment Design

Unsupervised environment design is a method for reinforcement learning in which the agent is given increasingly difficult environments that are still solvable for the agent, but also challenging enough so that the environment is not too easy to master. This discovers a curriculum for agents by always providing the agent with environments that are hardly solvable in the current training process. Dennis et al. [10] train two opposing agents with minimax regret, with one agent coupled to the environmental designer. Regret is the difference between the performance of the two agents, namely how good the agent could be and how good it actually is. This ensures that the levels generated are still solvable.

ACCEL [11] improves this method by using an evolutionary approach to adapt the difficulty of the environments and only trains a single agent for calculating the minimax regret. Figure 5 shows two examples of evolving environments that are increasingly difficult to solve for the agent. The upper environment is the MiniGrid environment [12], which is used to create mazes that the agent has to solve. The lower environment is the bipedal walker environment introduced in [13], where an agent must learn to run over obstacles.

3.4.4 Proposed Method

We propose to train a teacher agent that discovers a curriculum of increasingly hard problem sets to challenge a student agent in combination with a program synthesis system such as DreamCoder [8]. This should enable the program synthesis system to explain the behavior of the strategies found, while at the same time the student agent learns to solve increasingly difficult problems. As DreamCoder solves more and more levels by imitating the student agent, the DSL is updated with more representative functions for the environment. This will bootstrap the entire system and makes it possible to learn a custom DSL for the problem, which can be used by human experts to examine agent behavior.

In general, this method proposes a new idea for learning an end-to-end system that can explain game strategies or reinforcement learning policies by finding a tailored DSL for a given problem without using too much prior knowledge, since this knowledge should be found by the system itself. One challenge is the combination and interaction of all mentioned methods into a single system, that can generate programmatic policies and is trainable from scratch.

References

1 Silver, D., Huang, A., Maddison, C., Guez, A., Sifre, L., Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V. & Others, Mastering the Game of Go with Deep Neural Networks and Tree Search. *Nature*. **529**, 484-489 (2016)

2 Campbell, M., Hoane, A. & Hsu, F. Deep Blue. *Artif. Intell.*. **134**, 57-83 (2002,1)
3 Berner, C., Brockman, G., Chan, B., Cheung, V., Dębiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C. & Others, Dota 2 with large scale deep reinforcement learning. *ArXiv Preprint ArXiv:1912.06680.* (2019)
4 Vinyals, O., Babuschkin, I., Czarnecki, W., Mathieu, M., Dudzik, A., Chung, J., Choi, D., Powell, R., Ewalds, T. & Others, Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature.* pp. 1-5 (2019)
5 Silver, T., Allen, K., Lew, A., Kaelbling, L. & Tenenbaum, J. Few-Shot Bayesian Imitation Learning with Logical Program Policies. *The Thirty-Fourth AAAI Conference On Artificial Intelligence, AAAI 2020, New York, NY, USA, February 7-12, 2020.* pp. 10251-10258 (2020)
6 Verma, A., Murali, V., Singh, R., Kohli, P. & Chaudhuri, S. Programmatically Interpretable Reinforcement Learning. *Proceedings Of The 35th International Conference On Machine Learning.* **80** pp. 5045-5054 (2018,7,10)
7 Inala, J., Bastani, O., Tavares, Z. & Solar-Lezama, A. Synthesizing Programmatic Policies that Inductively Generalize. *International Conference On Learning Representations.* (2020)
8 Ellis, K., Wong, C., Nye, M., Sablé-Meyer, M., Morales, L., Hewitt, L., Cary, L., Solar-Lezama, A. & Tenenbaum, J. DreamCoder: bootstrapping inductive program synthesis with wake-sleep library learning. *PLDI '21: 42nd ACM SIGPLAN International Conference On Programming Language Design And Implementation.* pp. 835-850 (2021)
9 Hewitt, L., Anh Le, T. & Tenenbaum, J. Learning to learn generative programs with Memoised Wake-Sleep. *Proceedings Of The 36th Conference On Uncertainty In Artificial Intelligence (UAI).* **124** pp. 1278-1287 (2020)
10 Dennis, M., Jaques, N., Vinitsky, E., Bayen, A., Russell, S., Critch, A. & Levine, S. Emergent Complexity and Zero-shot Transfer via Unsupervised Environment Design. *Advances In Neural Information Processing Systems.* **33** pp. 13049-13061 (2020)
11 Parker-Holder, J., Jiang, M., Dennis, M., Samvelyan, M., Foerster, J., Grefenstette, E. & Rocktäschel, T. Evolving Curricula with Regret-Based Environment Design. *International Conference On Machine Learning.* (2022)
12 Chevalier-Boisvert, M., Willems, L. & Pal, S. Minimalistic Gridworld Environment for OpenAI Gym. *GitHub Repository.* (2018), https://github.com/maximecb/gym-minigrid
13 Wang, R., Lehman, J., Clune, J. & Stanley, K. POET: open-ended coevolution of environments and their optimized solutions. *Proceedings Of The Genetic And Evolutionary Computation Conference.* (2019)

3.5 Artificial Intelligence for Time-Travelling Games

Ana Matran-Fernandez (University of Essex – Colchester, GB), Manuel Eberhardinger (Hochschule der Medien – Stuttgart, DE), Jakob Foerster (University of Oxford, GB), Simon M. Lucas (Queen Mary University of London, GB), Paris Mavromoustakos Blom (Tilburg University, NL), and Pieter Spronck (Tilburg University, NL)

Time-travel has long been explored as a mechanism in science-fiction, particularly books, movies, and, to a lesser extent, video games, all with different degrees of success by reviewers and consumers. In this report we explore different types of time-travel mechanisms that could be offered in video games and discuss some design aspects that need to be considered in such video games, as well as the components that might need artificial intelligence implementations and the considerations for a successful time-travel experience. A companion report in this collection considers practical implementation and multi-player aspects of time travel.

3.5.1 Introduction

There are many types of time-travelling operations that can enhance a player's experience of a video game. Many aspects of time travel have been explored in film and literature, with time-travel paradoxes being fundamental to the plot in films such as *Terminator*, *Back to the Future* and *Source Code*. As far as we are aware only a handful of games have significant time-travel elements, beyond the *Save Game* facility which we discuss below. A key difference between games compared with other narrative media (and indeed normal lived experience), is the assumption of a linear timeline and single common reality in the latter, whereas it is standard to analyse games in terms of game trees (or more generally graphs). Hence what appears as a paradox in a film may simply be an alternative branch of a game-tree in a game.

Notable examples of time-travel games include the following – here we just comment on the time-travel aspects of them, they are all covered extensive Web coverage including WikiPedia entries which we recommend for further reading.

5D Chess with Multiverse Time Travel: a mind-bending game where play proceeds on multiple time-lines and moves are made in four dimensions: the normal x-y of a chess-board, plus time and time-line.
Life is Strange: The player can rewind recent actions to play them out differently.
Millennia: Altered Destinies: The interplanetary civilisation game plays out through 10,000 years with the player time travelling to nurture four races throughout this period with multiple time-related mechanics to help foster their development.
Deathloop: Like *Ground Hog Day* each day resets for most characters, and depending which character we play as the aim is to either break this loop, or to maintain it.

Although limited in number, time travel games have a distinct appeal with very positive reviews.

When talking about travelling to the past, the simplest, and least interesting, form of time-travelling would be equivalent to loading a previously saved game, which can be seen as a trip to the past in which a player retains only their memories, but their inventories, etc., remain as they were at the point where the game was saved. For example, in a poker game, this would be equivalent to loading the game you just lost with the knowledge you have from having played it, and using it to change your own strategy and win this second time.

More interesting approaches are those in which a player can travel back in time with items from the future that can be used in the past, or use the trip to the past to influence the further progress of the game, while still being only one copy of the avatar. Following the poker example above, this could mean that one can travel to the past and manipulate the deck of cards before playing the game, so that the player wins.

In a further level of complexity, we explore also the cases in which the trip to the past (in any of the cases mentioned above) involve the cloning of the avatar, so that there are at least two copies of the player's avatar in the game. Whether or not both can be manipulated, and if one is terminated after reaching the point where the trip to the past was triggered, are further design decisions that influence the game play. In the poker scenario, an example of this level would be one in which the player travels back in time and joins the game with their clone, being able to manipulate it in some way to enable their clone to win.

3.5.2 Time-travel as a state transition

One of the reasons why time-travelling content is tricky to execute well is the time-travel paradox that could arise when a character travels back in time and leads to logical inconsistencies in the future [1].

To illustrate the time-travel paradox options, let us assume that we are currently in Universe A, which contains a diamond D_A, and that we can travel back in time and carry it with us to a previous time. If the time-travel operation is implemented so that we are in the same universe A, the options are that we either have two copies of this diamond (which may be a paradox), or that the original diamond D_A that was in the universe at that time is in some way destroyed, changed, or teleported so that only one D_A exists. If instead the time-travel operation transfers our character to a different universe B, which contains its own diamond D_B, then there is no paradox, as the two diamonds can exist in this universe.

Time-travel in a game is an operation that can be implemented as a state transition which needs an additional state. Namely, this is the state we want to travel to. Let a typical game operator be represented as $S_{n+1} = f(S_n, a_n)$, where S_n is the state of the game at time n, and a_n is the action taken at that time. Then, we can define a low-level time-travel operator as $S_k = f(S_n, k, a_n)$, where k is a time in the past (we will discuss time-travelling to the future below). Here we envisage that k is an absolute time, but an alternative is to have k as a delta to the current time. The choice of types of time-travel that are allowed to a player are therefore embedded in the state transitions, and these specify the rules for the universe of the game. The time travel action a_n could indicate the inventory the avatar takes back with them to time k, for example. Therefore, even when visiting a state in the past (i.e. $k < n$), the state may be different as it includes the inventory (and perhaps avatar state) from the future.

Although we are considering time-travel in the most obvious sense of the player's avatar travelling through time, there are other possible time-warping mechanisms such as sending objects or messages through time. All of these have interesting game-play possibilities.

One should also consider which aspects of the game should be fixed and which should be stochastic. For example, a gamer could go back in time once they know the winning combination of the lottery and buy the correct ticket. However, this might also be an unintended consequence of bad design if the seed for certain simulations remains fixed. This could potentially be offset if there was a cost to using time-travelling as an action. For example, there could be a limit to how many times an avatar can time-travel (thus being a limited resource), or perhaps the avatar could be slower or older after each trip in time.

3.5.3 Artificial Intelligence in Time-travel

First we consider the role AI can play in the more mechanical aspects of time travel i.e. the enabling of the time-travelling state transition operator, when realistic agent actions are required to reach the required state.

Whereas travelling in time to the past does not necessarily require AI agents, the need for these is clear when the time-travel is in the forward direction, so we will look at these two cases (briefly) separately, and then consider other applications of AI.

3.5.3.1 AI when travelling to a previous state

One of the clear cases where AI agents are needed when time-travelling to the past is when a clone of the player's avatar is created in the trip, and there are now two versions of the avatar during gameplay (the original and the one that has travelled to the past, which is typically the one that will be controlled by the player). In this case, the actions of the original avatar need to be recreated (no AI needed), but if the clone is acting in a way that interferes with actions already taken by the avatar in the first gameplay, logical inferences about what the player would have done are needed. This highlights the need for AI agents that can model the player's behaviour and act in similar ways. This also raises the question of how intelligent the AI is required to be – for example, should the AI controlled avatar show surprise on encountering a twin they never knew they had?

Furthermore, if the original avatar persists beyond the point where the trip to the past started, their new actions need to be inferred.

3.5.3.2 AI when travelling to a future time

The main motivation for implementing AI in a game when time-travelling to the future is the need for extrapolating a new future from the current state in a way that seems plausible. Note that even if intermediate steps are not observed, the generated future should still appear plausible to the person playing the game.

The complexity of this problem depends on the nature of the game – for games with rich narratives it could be both complex and interesting. For example, if the actions of the player have long term effects, then ensuring those actions are in line with the human player's personality is important in order to present compelling visions of the future. To stay with the lottery example, winning the lottery should be unlikely for a character that shuns any form of gambling.

When travelling to a future time, unless we specify all the actions of all agents, or fix all random seeds, then it's reasonable to have a distribution over possible future states, which can be filtered to meet certain criteria before being presented to the player.

3.5.3.3 Other applications of AI in Time Travel Games

Beyond enabling the mechanics of achieving plausible states, there is great potential for AI in play-testing time travel games. While AI agents can in principle be used to play-test any game, time-travel games can be especially confusing, making it hard for human designers and players to spot game-breaking loopholes.

AI for play-testing serves two main roles: one is to find bugs, included crashes, and the other is to check the quality of the game-play. The latter is harder to do well, and often relies on having agents of sufficient intelligence to explore the richness of the gameplay and strategic depth. The extra challenges posed by time-travel for AI are as yet unclear.

Adding time travel actions to a game would most likely increase its complexity, as we are increasing the action space. However, this is not necessarily so, as time travel could also break a game, to the point of rendering it trivial from a competitive viewpoint.

3.5.4 Conclusions

Time-travelling in games has the potential to be a fun mechanic that could be added to many games, but there are many considerations that need to be taken into account when designing the time-travelling component of the game, particularly where stochasticity is involved. Multi-player games also require special attention, which we cover in a companion report.

References

1 Tobar, G. & Costa, F. Reversible dynamics with closed time-like curves and freedom of choice. *Classical And Quantum Gravity.* **37**, 205011 (2020).

3.6 Multiplayer Time Travel

Jakob Foerster (University of Oxford, GB), Duygu Cakmak (Creative Assembly – Horsham, GB), Simon M. Lucas (Queen Mary University of London, GB), Setareh Maghsudi (Universität Tübingen, DE), Ana Matran-Fernandez (University of Essex – Colchester, GB), Paris Mavromoustakos Blom (Tilburg University, NL), Diego Perez Liebana (Queen Mary University of London, GB), Lisa Rombout (Tilburg University, NL), and Pieter Spronck (Tilburg University, NL)

Being able to travel in time is one of the ancient dreams of humanity and a topic that is explored broadly in popular culture and science-fiction. However, due to the first law of thermodynamics ("the entropy always increases") it is unlikely that time travel in the real world will ever be possible. In contrast, simulated worlds like computer games do not need to obey the laws of physics and thus, in principle, can offer the ability to time travel. Indeed, there are a number of examples of games in which players can use time travel as part of the gameplay. Crucially though, currently time travel is both limited to specific games and to the single player case. In this report we put forward a proposal for an API that allows extending time travel to arbitrary games and to the multi-player case.

3.6.1 Introduction

It is striking that time travel (TT) has captured the imagination of writers and scientists for centuries and is yet only rarely present in computer games, where it is *actually* possible. Of course, there are exceptions to the rule, but by and large existing computer games do not take advantage of TT. In this proposal we investigate what it would take to "time-travel-fy" arbitrary games. We will also investigate how we can extend this idea to the multi-player case. Imagine a world in which a game designer can easily import the "time-travel package" and use standard functions to deal with the state-keeping, game play logic etc. associated with time travel. In particular, we focus on two aspects of this logic: First of all, we investigate the role of randomness and, secondly, we address multi-player time travel.

3.6.2 Of Lottery Tickets and Dice

One of the crucial issues is that TT allows a player to potentially "hack" the game logic as long as there is any randomness in the game. There are two different potential problems with opposite impact: The first case is the "lottery ticket", whereby a player could travel back in time and guess the correct lottery ticket which they had observed in the future. In this instance a simple fix is to *re-randomise* the lottery draw during each instance of time travel. However, re-randomisation has a separate issue: It allows the player to "keep trying" until they obtain the outcome that they want and continue the gameplay from there. For example, when a unit attacks a stronger unit it might still have a finite probability of success and the player could travel back in time until they "get lucky". To mitigate this, some circumstances require *freezing the randomness* across time, rather than re-running random events on every path forward through time.

Finally, addressing both issues requires a higher-level "semantic understandig" of outcomes. A player should not be able to *ceteris paribus* obtain a better outcome by traveling back in time and *hacking the randomness*. In other words, if the player didn't win the lottery on the initial travel through time, they should not be able to do so on the second attempt. How to implement this using game AI is an open problem that we hope to address in future work.

3.6.3 Multi-Player Time Travel

Time-travel in the single player case closely resembles saving the game state and reloading past checkpoints later on. In contrast, in the multiplayer case things get a lot more interesting. When a number of players co-exist in the same environment it is unclear how time travel of one player should change the current time step of other players. A naive approach is to simply use the single-player option, whereby all players are "dragged through time" by the time travel decisions of any player. However, this will likely make for a confusing playing experience and also break game dynamics since any player might be incentivised to travel in time when things are not working well for them.

Instead, we suggest a new approach for multi-player TT which relies on *branching timelines*: Any player can travel back in time *independently* while all other players have the option to continue playing on their current timeline or TT independently. This leaves a crucial question: What are the characters of other players doing on branches that the player is not currently playing? We suggest to use "zombie-actors", i.e. machine learning models that predict the actions of a player in the *alternate* reality given their realised actions in the played reality. This problem is similar to the issues caused by time-delay in multi-player games, which are solved e.g. with *Rollback* which is now being improved using machine learning [1].

To reduce computational overhead and avoid pure "zombie-games", branches that have been abandoned by all players get frozen in time until a player rejoins said branch.

References

1 Anton Ehlert. *Improving input prediction in online fighting games.* 2021

3.7 Artificial Intelligence for Audiences

Antonios Liapis (University of Malta – Msida, MT), Maren Awiszus (Leibniz Universität Hannover, DE), Alex J. Champandard (creative.ai – Wien, AT), Michael Cook (Queen Mary University of London, GB), Alena Denisova (University of York, GB), Alexander Dockhorn (Leibniz Universität Hannover, DE), Tommy Thompson (AI and Games – London, GB), and Jichen Zhu (IT University of Copenhagen, DK)

Artificial Intelligence (AI) has been leveraged for assisting individual players [20, 12] and individual designers or creators [9], but the rise of for-profit content creation platforms [3], and games as a spectacle [1] opens a new and exciting opportunity for AI support. In this working group, we explore applications, algorithms, and interfaces for *AI for audiences*.

The simplest inception of an AI application in this vein would be as mediator between a content creator (e.g. a YouTuber or a Twitch streamer) and the consumers that may be enjoying this content in real-time (e.g. during a stream) or asynchronously (e.g. watching a YouTube video). Focusing on the communication between audience and content, the working group identified the following non-exhaustive list for possible AI roles:

- **AI as mediator.** For instance, the AI may inform a viewer when the content changes (e.g. a new game area is entered or the creator changes the discussion topic), or inform a live-streamer when audience engagement shifts (in tone, volume, or discussion topic).

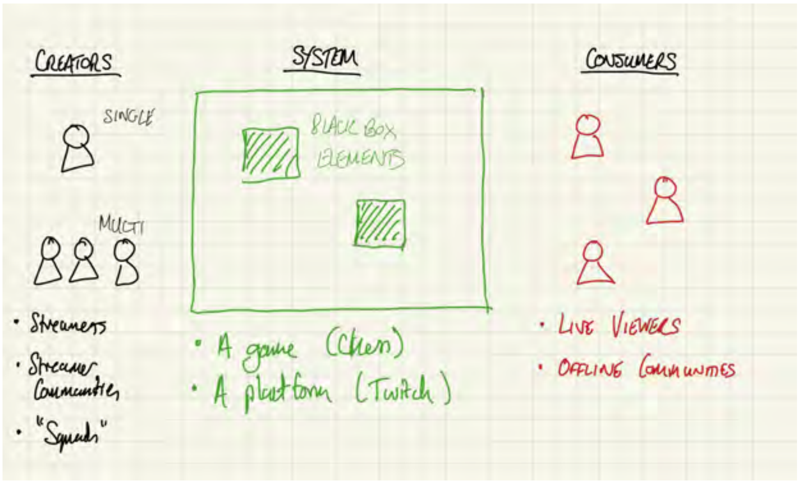
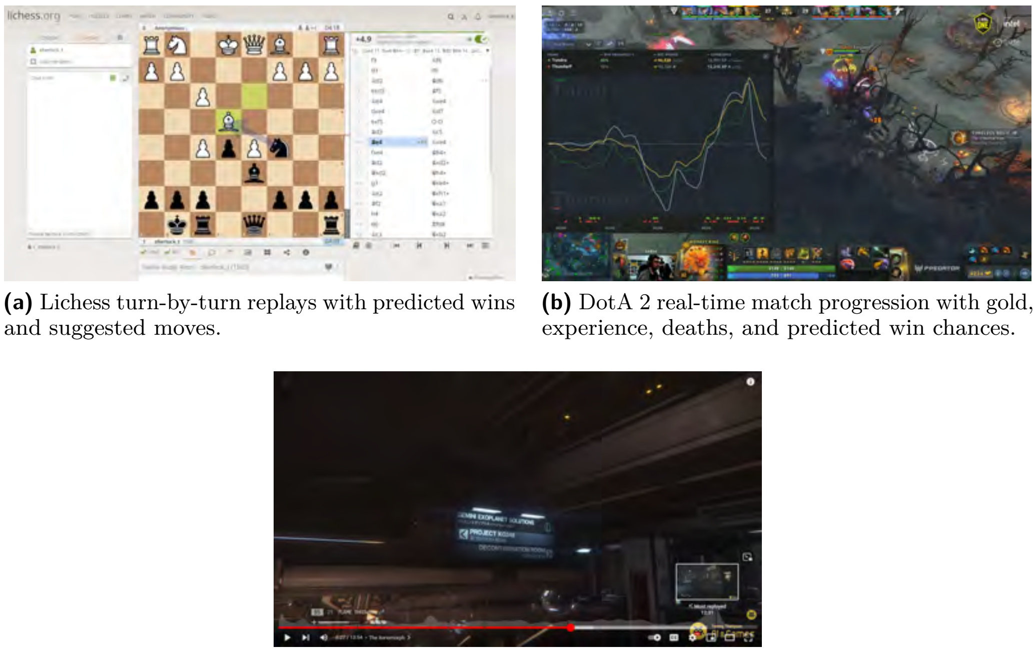


Figure 6 Envisioned AI as mediator between an audience and one or many content creators.

- **AI as entertainer.** For instance, the AI can add a (textual) commentary to a playthrough in real-time. In this role, the AI may act as an *unreliable narrator*, in which case the state of the game need not be described reliably in order to increase engagement through uncertainty and curiosity. Similar patterns are observed in e.g. e-sport competitive matches, where (human) casters give more "optimistic" predictions for a comeback of the currently losing team.
- **AI for hype.** For instance, the AI can algorithmically generate audio, visual, or text assets to promote content scheduled in the future by connecting it with past content from the same creator or a broader context. Similarly, the AI can promote existing content to the audience based on more in-depth patterns (e.g. gameplay progression) and player/viewer models than current recommender systems.
- **AI as tutor.** For instance, when requested by a viewer an AI could explain game mechanics and their interactions as relevant to the current context. The issue of personalisation is pertinent here, as modeling the viewer's expertise (based on the number of similar content they have viewed or games they have played, as well as questions they have asked the AI) could impact the level of explanation and possible examples or anchor points to scaffold the explanation.
- **AI as filter of needless data.** For instance, an on-demand AI can jump to the highlights in the video, or an always-on AI can remove uninteresting or toxic chat between audience members.

The issue of synchronous versus asynchronous engagement can heavily impact the affordances and constraints for both the AI algorithms and the user interfaces. Beyond the obvious fast-response and low-latency requirements, the issue is pertinent because synchronous viewing may foster shorter but more direct interactions between content creator and audience and between members of the audience (e.g. chat). Synchronous viewing opens additional opportunities for AI assistance, such as a personalized recap of the stream so far in case a viewer joins late, or a recap of events while the user was away in case they leave and rejoin. On the other hand, asynchronous viewing allows for more thoughtful discussions to emerge in comments; at the same time interaction with the content is more granular and controlled as viewers can choose which parts of the video to view, rewind, etc.



(a) Lichess turn-by-turn replays with predicted wins and suggested moves.

(b) DotA 2 real-time match progression with gold, experience, deaths, and predicted win chances.

(c) YouTube viewership analytics, including the a "most replayed" label for popular video sections.

Figure 7 Current examples of visualizations, analytics, and predictions intended for audiences.

Note, that the data format of the content that is made available to the AI should ideally not be simply the end-product (e.g. a video) but additional meta-data regarding game actions, context, and potentially even game-specific AI game players. An example of such rich data is provided in *lichess*[1] where viewers (or players after the game is completed) can watch replays of chess matches along with AI-based predictions of win versus loss after every move, as well as suggested moves instead of the one played. Beyond chess, having access to such granular game data could allow for highlight detection (e.g. at points where the predictions shift dramatically between players), summarization (e.g. grouping similar moves together and focusing on highlights), or tutoring (e.g. showing the causal links between early choices and later outcomes). To maximize the potential of such an approach, however, the game developers would need to provide not only game state and action events but also ideally some game-specific AI that could provide nuanced context-specific metrics such as predicted win probability or chosen next moves. Such meta-data and AI-predicted game metrics are already made available for certain games that embrace the game as spectacle philosophy, especially e-sports such as *Dota 2* (Valve, 2013).

However, AI for audiences need not rely on the assumption of a *one-to-many* interaction, or the implicit assumption that the audience consists of passive consumers with no agency over the content or how they interact with it. AI for audiences can be used to promote and support *augmented communities*, where some or all of the audience members can take more proactive roles (indicatively, live commentators with AI visualization assistance or cinematographers by creating custom camera positions in live or replay game data). Audience interactions

[1] https://lichess.org/

with the AI itself can also lead to improved computational models, including player models [26, 19] that can provide personalized tutoring (based on detected expertise level) but also for matchmaking between audience members (especially those with proactive roles). Similarly, the AI can operate on a *many-to-many* assumption and find similar content with similar game-states from other streamers to propose to viewers, but also for matchmaking between content creators. The simplest form of AI for content creators could suggest scheduling clashes with popular content creators in the same genre (or followed by the same audience) or niche topics that have not been explored by other content creators. A more proactive AI could also act as a matchmaker between content creators, suggesting ideas on how and on what topic this collaboration could be built on. Algorithms and interfaces for this type of AI assistance can have broader ramifications, as similar many-to-many relationships can be found in crowdfunding platforms (e.g. Kickstarter), virtual crowd working platforms (e.g. Fiverr or creative.ai), and service providers more broadly (e.g. Uber, Wolt).

Several existing algorithmic advancements can be leveraged towards the goals laid out above, including recommender systems [19, 12], text summarisation [13, 21], personalisation [10] and personas [6], highlight detection [12], video indexing and matching [22], viewership analytics [7], coordination and scheduling [2], monetisation and churn prediction [8], expressive range analysis [16] and quality-diversity search [4], AI directors [11, 17], and more. However, novel AI research will be warranted in this vein tailored to the format (video, speech, and game meta-data) and user requirements of such applications. Example directions for AI research include question-answering systems (including natural language processing), text summarisation of real-time expanding datasets (of comments or gameplay), context-aware detection of video segments (e.g. based on text mentions in the comments), or causal models [14] based on audio, visual, video, gameplay, and comment/chat data.

References

1 Dave Boling. How The International became a global "Super Bowl for nerds". `https://www.espn.com/esports/story/_/id/20343989/super-bowl-nerds-dota-2-fans-globe-lured-spectacle-camaraderie-international-7`, 2017. Accessed 27 July, 2022.

2 Elisabeth Crawford and Manuela M. Veloso. Learning to select negotiation strategies in multi-agent meeting scheduling. In *Proceedings of the Portuguese Conference on Artificial Intelligence*, 2005.

3 Cecilia D'Anastasio. Amazon's Twitch seeks to revamp creator pay with focus on profit. `https://www.bloomberg.com/news/articles/2022-04-27/amazon-s-twitch-seeks-to-revamp-creator-pay-with-focus-on-profit`, 2022. Accessed 27 July, 2022.

4 Daniele Gravina, Ahmed Khalifa, Antonios Liapis, Julian Togelius, and Georgios N. Yannakakis. Procedural content generation through quality-diversity. In *Proceedings of the IEEE Conference on Games*, 2019.

5 Fabian Hadiji, Rafet Sifa, Anders Drachen, Christian Thurau, Kristian Kersting, and Christian Bauckhage. Predicting player churn in the wild. In *Proceedings of the IEEE Conference on Computational Intelligence in Games*, 2014.

6 Christoffer Holmgård, Michael Cerny Green, Antonios Liapis, and Julian Togelius. Automated playtesting with procedural personas through MCTS with evolved heuristics. *IEEE Transactions on Games*, 11(4):352–362, 2019.

7 Andrew Hutchinson. Youtube rolls out activity graph to all videos, ups the maximum price of channel memberships. `https://www.socialmediatoday.com/news/youtube-rolls-out-activity-graph-to-all-videos-ups-the-maximum-price-of-ch/624036/`, 2022. Accessed 27 July, 2022.

8 Erik Johnson. A deep dive into Steam's Discovery Queue 2. `https://www.gamedeveloper.com/business/a-deep-dive-into-steam-s-discovery-queue`, 2019. Accessed 6 July, 2022.

9 Antonios Liapis, Gillian Smith, and Noor Shaker. Mixed-initiative content creation. In Noor Shaker, Julian Togelius, and Mark J. Nelson, editors, *Procedural Content Generation in Games: A Textbook and an Overview of Current Research*, pages 195–214. Springer, 2016.

10 Santiago Ontanon and Jichen Zhu. The personalization paradox: The conflict between accurate user models and personalized adaptive systems. In *Companion Proceedings of the International Conference on Intelligent User Interfaces*, page 64–66, 2021.

11 Mark O. Riedl, H. Chad Lane, Randall Hill, and William Swartout. Automated story direction and intelligent tutoring: Towards a unifying architecture. In *Proceedings of the AIED Workshop on Narrative Learning Environments*, 2005.

12 Charlie Ringer and Mihalis A. Nicolaou. Deep unsupervised multi-view detection of video game stream highlights. In *Proceedings of the International Conference on the Foundations of Digital Games*, 2018.

13 Alexander M. Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015.

14 Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Toward causal representation learning. *Proceedings of the IEEE*, 109(5):612–634, 2021.

15 Adam M. Smith, Chris Lewis, Kenneth Hullet, Gillian Smith, and Anne Sullivan. An inclusive taxonomy of player modeling. Technical Report UCSC-SOE-11-13, 2011, University California Santa Cruz, 2011.

16 Gillian Smith and Jim Whitehead. Analyzing the expressive range of a level generator. In *Proceedings of the FDG workshop on Procedural Content Generation in Games*, 2010.

17 Tommy Thompson. In the directors chair: The AI of Left 4 Dead. `https://medium.com/@t2thompson/in-the-directors-chair-the-ai-of-left-4-dead-78f0d4fbf86a`, 2014. Accessed 27 July, 2022.

18 Tommy Thompson. How Forza's Drivatar actually works. `https://www.gamedeveloper.com/design/how-forza-s-drivatar-actually-works`, 2021. Accessed 6 July, 2022.

19 Hao Wang, Naiyan Wang, and Dit-Yan Yeung. Collaborative deep learning for recommender systems. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015.

20 Georgios N. Yannakakis, Pieter Spronck, Daniele Loiacono, and Elisabeth André. Player modeling. In Simon M. Lucas, Michael Mateas, Mike Preuss, Pieter Spronck, and Julian Togelius, editors, *Artificial and Computational Intelligence in Games*, volume 6 of *Dagstuhl Follow-Ups*, pages 45–59. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2013.

21 Kevin Yauris and Masayu Leylia Khodra. Aspect-based summarization for game review using double propagation. In *Proceedings of the International Conference on Advanced Informatics, Concepts, Theory, and Applications*, 2017.

22 Xiaoxuan Zhang, Zeping Zhan, Misha Holtz, and Adam M. Smith. Crawling, indexing, and retrieving moments in videogames. In *Proceedings of the Foundations of Digital Games Conference*, 2018.

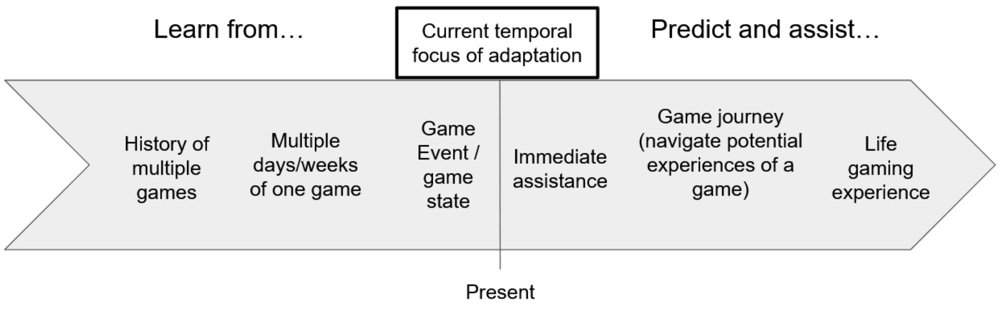

Learn from…
Current temporal focus of adaptation
Predict and assist…
History of multiple games
Multiple days/weeks of one game
Game Event / game state
Immediate assistance
Game journey (navigate potential experiences of a game)
Life gaming experience
Present

Figure 8 Horizon of past user activities and horizon of predicted future trends, used for choosing actions to take on the part of the assistive AI.

3.8 Personalized Long-Term Game Adaptation Assistant AI

Antonios Liapis (University of Malta – Msida, MT), Guillaume Chanel (University of Geneva, CH), Alena Denisova (University of York, GB), Casper Harteveld (Northeastern University – Boston, US), Mike Preuß (Leiden University, NL), and Vanessa Volz (modl.ai – Copenhagen, DK)

Artificial Intelligence (AI) in games has already been extensively used for the purposes of modeling players [26, 19]. This working group viewed issues of player modeling through the lens of **personalized assistance**, focusing on the horizon(s) that such models could use to **learn from the past** and to **predict the future**. While the scope and purpose of player models can vary [19], this working group focused on models of an individual person/player which can be generative in scope, i.e. generating "data where a human player could otherwise be consulted" [19].

What constitutes long-term personalization?

As a central issue of the topic tackled by the working group was the "long-term" aspect, it is important to define the scope of such temporal and contextual information. As depicted in Figure 8, the main questions on this aspect revolve around (a) the horizon of past user actions (and their context) that the model will learn from, and (b) the horizon of future expectations that the AI can predict and assist towards.

The working group identified that a personalized computational model could learn patterns from a very long-term history of player behavior at low granularity (with metrics such as game purchase behavior and/or playtime), a mid-term history within one game (spanning e.g. multiple days or weeks), or short-term history spanning a few actions or game-states within the current game context. In terms of what predictions the model could make, the working group similarly identified short-term predictions regarding actions within the current game session (e.g. whether the player would fail in an upcoming challenge), mid-term predictions regarding behaviors within the same game (e.g. which parts of future game content the player would enjoy and how), or longer-term predictions (e.g. when the player would quit this game, or which games they would pick up after it).

We note here that an assistant AI does not necessarily require prediction of future states in order to provide assistance, as it can operate without output [27] by detecting patterns between this player and a broader player corpus through unsupervised learning (as would be the case for recommender systems, for instance). However, the context of the assistance similarly fits the same time-scales as player predictions, from short-term assistance regarding e.g. a current problem the player is facing in this phase/location of the game versus long-term assistance in terms of e.g. similar games they can play once they finish this game.

At this level of granularity, there is an abundance of examples to draw from in commercial and research applications modeling past and future horizons. Indicatively, player profiles on Steam take into account game purchases in order to provide similar games to recommend in the player's discovery queue [12] (long-term past for long-term future). On the other end of the spectrum, AI replicants [16] that follow the low-level decision-making of a single player in a singular context (e.g. level layout and opponent) can be used to simulate the next in-game actions (short-term past for short-term future). On a more realistic mid-term assistance, the Drivatar models [20] learn how to drive as a player would and can generate complete playthroughs through sequences of short-term decisions in the style of the player, even in unseen tracks (mid-term past for short-term future). While not explicitly aimed to assist players, similar work on churn prediction [8] focuses on learning patterns from a player's long- or mid-term gameplaying and purchase history in order to predict when players may quit playing the game (long-/mid-term past for mid-term future); these predictions are often used to provide players with interesting content or power-ups in the short-term in order to delay players from dropping out.

How can the AI assist a player?

While to a large extent the algorithms for player modeling are already mature, a more pertinent issue relates to the type of assistance that such models can offer to players. Through extensive brainstorming, the working group identified the following non-exhaustive list of assistant AI actions:

- **Game Selection:** The assistant AI can suggest new games for the player to explore. In this level of granularity, the AI does not adapt any game content and relies on human-authored games that are better suited for this player.
- **Modification of an initial game state:** The assistant AI provides the player with new content within the same game, modifying the initial state via e.g. a new game level to explore [25], making a new mechanic available [1, 3], or new opponent abilities [13, 18]. The distinction here is that the assistant AI adapts the *possibility space* offered to the player without hand-holding the player on how they should take advantage of these possibilities. This assistance is also highly relevant in terms of generating *end-game content* through e.g. recombining existing hand-authored content in novel ways that match player preferences, performance, or expectations.
- **Mechanics adaptation:** The assistant AI interferes in a more granular manner on the moment-to-moment playthrough by adjusting the game mechanics themselves. This could for example take the form of aim assistance by increasing the leniency on what constitutes a hit in a shooter game (e.g. [23]). This same adaptation, due to its subtle nature, could also be used for increasing accessibility in games that would normally require fast reflexes [21].
- **Adapting the player's behavior towards a normative gameplay goal:** Rather than changing the game according to the player's preferences, this assistant AI shoehorns the player into playing the game as-is according to the designers' (rather than the players') intentions. This assistive AI can take two complementary roles, guiding players during

their playthrough towards intended outcomes by *scaffolding and mediating their learning* and by *nudging them towards desirable behaviors.* Scaffolding can be done through generated tutorials on overlooked game mechanics [7] or on-demand hints regarding actions – or in-game knowledge – needed to overcome a current challenge (e.g. a puzzle). Nudging and priming on the other hand can be achieved by making certain decisions seem more appealing, and has been used extensively in both advertising [24] and teaching [4]. In order to guide a player towards a specific level traversal path, for example, a specific path may be adapted to have less clutter in order to guide players through it [22], sound emitters could be used to guide players towards their source [14], or user interface elements (e.g. quest markers) could be adapted to be more or less prominent. This type of assistance is perhaps the least explored academically in the context of games while also the most promising and realistic from the perspective of the game industry. This is relevant to the game industry as game developers can thus streamline a singular play experience, while taking advantage of existing – carefully crafted – assets without requiring unpredictable generation or adaptation.
- **Providing Reflection and Explainability:** The assistant AI provides feedback to the player regarding their performance or playstyle, allowing players themselves to reflect on how to improve the former or diversify the latter. While post-game summaries abound in games, the AI aspect can be leveraged to provide personalized feedback (e.g. focusing on presenting metrics that are important to this player, based on their personal model) or for highlighting aspects or portions of the playthrough where alternative decisions or actions could have led to better results – as a form of post-game scaffolding. Moreover, post-game visualizations can also serve to explain certain AI decisions or prompts during the game covered in other AI actions above. For example, in a racing game the player's trajectory on the track is shown in a post-game summary, juxtaposed with an AI driver's trajectory and highlighting the points where the AI detected (and verbalized) a hint towards course correction. Therefore this AI action can be a standalone component or an accompanying explanation for other AI actions.

The issue of assistance was central in this particular envisioned application of AI, specifically regarding how (in)visible the assistant would be (e.g. performing difficulty adjustment or aim assistance behind the scenes versus coaching players to better handle the game's challenges). Relatedly, whether the assistance would be on-demand by the player or always in effect would impact the type of assistance the AI can provide as well as issues of players' perception of the AI and explainability requirements. Based on the type of assistance and how it is presented to the player, such AI could take the role of salesperson, tutor, gamemaster, commentator, tour guide, and even as general on-demand virtual assistant similar to Siri or Google Assistant but within the game.

What can the AI assist towards?

As a final dimension regarding the goals of the assistant, the player model could be trained to focus on a variety of metrics or key performance indicators (KPIs) of the player. The following non-exhaustive list covers some KPIs of interest:
- **Emotional state:** a player's emotional state in the game. AI assistance that keeps track of and aims to improve such a KPI could tailor content towards the intended emotional state (e.g. fear [6] or stress [15] in horror games) or in order to course-correct in case experienced emotions are overwhelming (e.g. in games for rehabilitation [9]).

- **Performance:** the difficulty or challenge a player faces in the game and how they overcome it (e.g. number of retries or game score). AI assistance can be used to tailor the experience to the player's skills. This is the most traditional application through dynamic difficulty adjustment [10], but can be enhanced beyond invisible rubber-banding through e.g. on-demand coaching and personalized assistance.
- **Coverage:** how much of the game a player has explored (or tends to explore). Coverage can refer to spatial coverage (e.g. heatmap of the level), action coverage (e.g. whether the player makes use of all mechanics and dynamics [11] available to them), narrative coverage (e.g. which non-player-character relationships the player has focused on and how), or temporal coverage (e.g. build orders in strategy games).
- **Learning:** how much the player has mastered the game's mechanics (or concepts) and improved their repertoire. This KPI could be especially meaningful for assistance on mechanics that the player has overlooked (e.g. via hints and tutorials) or has trouble with (e.g. via aim assist that progressively becomes less pronounced). The aspects of the game that the player has mastered can also inform recommendations for future games that specifically offer additional challenges (or content) in these specific aspects.
- **Intentions:** why and how a player likes to engage with the game. This is particularly meaningful for detecting cases where certain play behaviors are not due to poor performance but due to conscious decisions to play subversively or towards the player's own goals. A broader example of this would be speedrun challenges [17], where assistance should be tailored not to guide players towards maximizing spatial coverage but towards shortcuts or skill-based shorter traversal paths.
- **Social experience:** how a player interacts with other players in the game. This KPI is mostly relevant to multi-player games with a social component, such as massively multiplayer online games rather than competitive brawlers or racing games. The AI model could capture cases of toxic behavior or interactions within and outside the game, and its "assistance" could include warnings in cases of toxic behavior [2] either to the perpetrator or to new players interacting with them.
- **Monetization:** how a player spends their real-world money in the game, and towards which content. This KPI is less relevant for the research purposes of this working group, but could be relevant for industrial use cases. Moreover, extensive AI research on churn prediction and analytics [5, 8] has been motivated by monetization and thus can not be overlooked. Ethical assistance in terms of this KPI could focus on the explainability and reflection aspect, highlighting to the player which purchase behaviors they tend to make and coaching them towards diversifying or reducing their in-game purchases.

Envisioned use-cases of assistant AI

After these high-level implications of personalized AI assistance were laid out, the working group focused on two practical examples, both including nudging player behaviors to better experience a game as-is (without new content being generated for it). The first example was focused on narrative-based games with explicit role-playing decision points. Speculative work under this more narrow use-case explored different ways of nudging the player's decisions in terms of their *invasiveness* (i.e. how much the AI takes over the decision-making) and the *subconscious nature* of the nudging (i.e. whether the player might understand that they are being manipulated). The second example was focused on assisting players to brake within a racing game, which relies on kinesthetic player behavior and the AI aims to reduce the challenge of an existing game. Speculative work to address this issue identified audiovisual feedback in real-time as the most intuitive way of AI assistance, exploring how audio or visual feedback could be more or less invasive (e.g. a popup foreshadowing the type of upcoming turn, versus a ghost trail of the ideal trajectory for taking the turn). The two practical examples allowed for some more in-depth discussion on the specific challenges that would need to be addressed when developing personalized AI assistants.

References

1 Eric Butler, Adam M. Smith, Yun-En Liu, and Zoran Popovic. A mixed-initiative tool for designing level progressions in games. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, page 377–386, 2013.

2 Alessandro Canossa, Dmitry Salimov, Ahmad Azadvar, Casper Harteveld, and Georgios Yannakakis. For honor, for toxicity: Detecting toxic behavior through gameplay. *Proceedings of the ACM CHIPLAY Conference*, 2021.

3 Michael Cook, Simon Colton, Azalea Raad, and Jeremy Gow. Mechanic Miner: Reflection-driven game mechanic discovery and level design. In *Applications of Evolutionary Computation*, volume 7835, LNCS. Springer, 2012.

4 Mette Trier Damgaard and Helena Skyt Nielsen. Nudging in education. *Economics of Education Review*, 64:313–342, 2018.

5 Magy Seif El-Nasr, Anders Drachen, and Alessandro Canossa. *Game Analytics: Maximizing the Value of Player Data*. Springer, 2013.

6 Magy Seif El-Nasr, Simon Niedenthal, Igor Kenz, Priya Almeida, and Joseph Zupko. Dynamic lighting for tension in games. *Game Studies*, 7(1), 2007.

7 Michael Cerny Green, Ahmed Khalifa, Gabriella A. B. Barros, and Julian Togelius. "Press Space to Fire": Automatic Video Game Tutorial Generation. In *Proceedings of the AIIDE workshop on Experimental AI in Games*, 2018.

8 Fabian Hadiji, Rafet Sifa, Anders Drachen, Christian Thurau, Kristian Kersting, and Christian Bauckhage. Predicting player churn in the wild. In *Proceedings of the IEEE Conference on Computational Intelligence in Games*, 2014.

9 Christoffer Holmgård, Georgios N. Yannakakis, Karen-Inge Karstoft, and Henrik Steen Andersen. Stress detection for PTSD via the StartleMart Game. In *Proceedings of the Conference on Affective Computing and Intelligent Interaction*, page 523–528, 2013.

10 Robin Hunicke. The case for dynamic difficulty adjustment in games. In *Proceedings of the International Conference on Advances in Computer Entertainment Technology*, 2005.

11 Robin Hunicke, Marc Leblanc, and Robert Zubek. MDA: A formal approach to game design and game research. In *Proceedings of AAAI Workshop on the Challenges in Games AI*, 2004.

12 Erik Johnson. A deep dive into Steam's Discovery Queue 2. `https://www.gamedeveloper.com/business/a-deep-dive-into-steam-s-discovery-queue`, 2019. Accessed 6 July, 2022.

13 Ahmed Khalifa, Scott Lee, Andy Nealen, and Julian Togelius. Talakat: Bullet hell generation through constrained map-elites. In *Proceedings of the Genetic and Evolutionary Computation Conference*, page 1047–1054, 2018.

14 Daryl Marples, Duke Gledhill, and Pelham Carter. The effect of lighting, landmarks and auditory cues on human performance in navigating a virtual maze. In *Proceedings of the Symposium on Interactive 3D Graphics and Games*, 2020.

15 Paraschos Moschovitis and Alena Denisova. Keep calm and aim for the head: Biofeedback-controlled dynamic difficulty adjustment in a horror game. *IEEE Transactions on Games*, 2022.

16 Johannes Pfau, Antonios Liapis, Georg Volkmar, Georgios N. Yannakakis, and Rainer Malaka. Dungeons & Replicants: Automated game balancing via deep player behavior modeling. In *Proceedings of the IEEE Conference on Games*, 2020.

17 Tom Phillips. World record Portal speedrun completed in 8 minutes. `https://www.eurogamer.net/world-record-portal-speedrun-completed-in-8-minutes`, 2012. Accessed 6 July, 2022.

18 Kristin Siu, Eric Butler, and Alexander Zook. A programming model for boss encounters in 2d action games. In *Proceedings of the AIIDE workshop on Experimental AI in Games*, 2016.

19 Adam M. Smith, Chris Lewis, Kenneth Hullet, Gillian Smith, and Anne Sullivan. An inclusive taxonomy of player modeling. Technical Report UCSC-SOE-11-13, 2011, University California Santa Cruz, 2011.

20 Tommy Thompson. How Forza's Drivatar actually works. `https://www.gamedeveloper.com/design/how-forza-s-drivatar-actually-works`, 2021. Accessed 6 July, 2022.

21 Tommy Thompson and Matthew Syrett. "play your own way": Adapting a procedural framework for accessibility. In *Proceedings of the FDG Workshop on Procedural Content Generation*, 2018.

22 Christopher W. Totten. *An Architectural Approach to level Design*, chapter Teaching in Levels through Visual Communication. CRC Press, 2014.

23 Rodrigo Vicencio-Moreira, Regan L Mandryk, and Carl Gutwin. Balancing multiplayer first-person shooter games using aiming assistance. In *2014 IEEE Games Media Entertainment*, pages 1–8. IEEE, 2014.

24 Markus Weinmann, Christoph Schneider, and Jan vom Brocke. Digital nudging. *Business & Information Systems Engineering*, 58:433–436, 2016.

25 Georgios Yannakakis and Julian Togelius. Experience-driven procedural content generation. *IEEE Transactions on Affective Computing*, 2(3):147–161, 2011.

26 Georgios N. Yannakakis, Pieter Spronck, Daniele Loiacono, and Elisabeth André. Player modeling. In Simon M. Lucas, Michael Mateas, Mike Preuss, Pieter Spronck, and Julian Togelius, editors, *Artificial and Computational Intelligence in Games*, volume 6 of *Dagstuhl Follow-Ups*, pages 45–59. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2013.

27 Georgios N. Yannakakis and Julian Togelius. *Artificial Intelligence and Games.* Springer, 2018. `http://gameaibook.org`.

3.9 The Tabletop Board Games AI Tutor

Diego Perez Liebana (Queen Mary University of London, GB), Duygu Cakmak (Creative Assembly – Horsham, GB), Setareh Maghsudi (Universität Tübingen, DE), Pieter Spronck (Tilburg University, NL), and Tommy Thompson (AI and Games – London, GB)

3.9.1 Introduction and Motivation

The aim of game tutorials is to teach a player how to play a game. The design and implementation of tutorials is no easy task, as they will be used by different types of players, with different experiences and prior knowledge of games. It's also hard to make them adaptive and interactive, especially in a way that identifies the actual capabilities and needs of the player as the learning progresses. In recent years, video-game tutorials have been object of study by game AI researchers who approach this problem from multiple angles. Recently, L. Poretski et al. [7] analyzed tutorials from 40 contemporary video games to identify common patterns and strategies used to generate these elements. Previously, E. Andersen et al. [1] studied the effect of tutorials on player engagement and game retention.

Later on, B. Aytemiz [2] investigated the design of tutorials that considered the skill of the player. Here, the authors implemented a Unity plug-in that provided information to the player only when this is needed. The automatic generation of tutorials has also been recently explored by M. Green and colleagues [5, 4] in AtDELFI, a system that procedurally generates instructions to teach players how to play 2D arcade games within the GVGAI framework [6].

While some research has been put into video-game tutorials, to our knowledge, no work has been carried out for table-top board game tutorials. Table top games (TTG) are normally played on a physical surface and involve the use and manipulation of certain elements, such as tokens, cards, dice or counters. TTG can feature relatively complex rules which may be hard to explain and understand, once the relationships and connections between the different game components and how are they used become intricate. Novice TTG players may experience a steep learning curve when facing some complex board games for the first time. Even some expert players may spend a big proportion of their playing session learning the game rules – even if some of the game mechanics are not new to them. Not surprisingly, many TTG players prefer to learn game rules via video tutorials or live explanations given by a colleague, rather than reading long rule-books. An interesting difference with video games is that TTG can't prevent a player *programmatically* to take invalid actions, sum victory points incorrectly or move your tokens not respecting the rules. Video games can enforce this by design and implementation (an avatar is a subjects to the game's physics model), so the tutorials teach the player to play the game *well*. TTG tutorials, however, must ensure that a player understands and applies rules in a context different to that of a digital tutorial. Current TTG frameworks, such as TAG [3], blur the line between TTG and their *digital twins*, but the game's user interface needs to be implemented with this aspect in mind.

Therefore, we propose research on tutorial generation for TTG as an area worth exploring: it would not only help players learn how to play games more effectively, but it will also provide interesting insights in decision making, game interfaces and game analysis. During this workgroup, we discussed the different challenges and opportunities offered by the generation of AI tutors. This chapter summarizes the discussions and considerations that arose from this group during the seminar.

3.9.2 Learning to Play

Once we ask ourselves the question *"how can an AI teach a human to play a Tabletop Board Game?"*, it's important to understand the different dimensions of the problem. One of them is the actual player: what does it mean for a player to know how to play a game? Is it enough that the player understands the rules so they can play the game? Should they be able to explain it to others, or is it sufficient that they make no mistakes when playing? Do players need to know just the valid actions and dynamics of the game, or do they need to have certain skill level to be able to devise good strategies and avoid flawed moves? Do we take into consideration that the player may know other similar games, or their general experience level with TTG?

Additionally, from the point of view of the AI, we may also ask what does the AI need to teach, how is this information conveyed, and when does the AI intervene (if at all) to explain a rule or to correct the player. What sort of capabilities does the AI tutor need to have, and does it build a model of the human to drive the teaching process? Can it analyze the player's gameplay traces to identify crucial decisions and does it take into account any external knowledge – for instance, the game rule-book, or other games that are known to the player? The following categorization analyzes the identified dimensions of this problem, which often overlap each other:

- **Introducing the Rules**: the main objective of a tutorial system is to explain the rules of the game. An important point regarding teaching these rules is *when* are they introduced. One option for the rules to be provided is to enumerate them all at front. This is the simplest scenario, which may suggest shouldn't probably deserve much consideration. However, some TTG *are* simple; if the number and/or the complexity of the rules is small, this approach may actually be the most appropriate. Another option is to provide, at the start, only information about the necessary rules, to then progressively explain new rules when they are needed. A final option would entail not providing any rules beforehand, letting the player *blindly* play the game. An explanation could then appear when a rule is broken, and subtle notifications could reinforce the player when a new rule has been successfully applied. This would have the added benefit of correcting the player mistakes and reinforcing their successes.
- **Contextualizing the Rules**: rules can be provided with different types of context. The simplest option would be, naturally, providing *no* context. For instance, in *Terraforming Mars* (TM; FryxGames, 2016), some cards can only be played if certain pre-requisites have been fulfilled. A rule that explains how this works with no context would just indicate the part in the card to look at when these pre-requisites exist. However, rules can also be explained with a *thematic* context: the card "Anti-gravity Technology", in TM, can only be played if the player has previously played 7 cards with a Science tag. Context can be given in the form of the theme of the game (many scientific projects are needed before Anti-gravity Technology can be discovered) and even hinting a *strategic* context (the game can be played with a Science strategy in mind to allow playing powerful cards later on). Finally, a useful context to certain rules is providing *distribution* information. In this same example, it is useful to indicate that only one card in the deck has such a restrictive requirement, but a given percentage of cards have requirements of having played different Science tags earlier. While the former examples concentrate in the rules per se, the latter are more useful to convey stronger play tactics.
  As hinted earlier, different approaches may be used for different players, or even at different times in the learning process. In complex games, the role of the AI tutor may help determine which rules should be explained when, or which sort of feedback or context should be given to the player. This is something that could be *learned* from data, for instance from previous tutoring sessions or from similar games.
- **Rules as Rule-sets:** some TTG, such as *Gloomhaven* (Cephalofair Games, 2017) or *Mage Knight* (WizKids, 2011), have a large collection of rules. Teaching or learning all these rules at once is challenging, thus in some cases learning is conveyed through different rule-sets. Initially, a version of the game (often simplified) is played using a simple subset of rules, easier to learn. More rules are added to the rule-set for consecutive game sessions, increasing the complexity of the game but providing a smoother learning curve than using the complete rule-set at first[2]. Note that this is different to progressively introducing rules (mentioned in a previous point), as this allows a complete game being played. An interesting line of research could be to investigate if an AI tutor can automatically find and compose the rule-sets and scenarios required for this incremental teaching approach.
- **Level of Play:** an AI tutor may be configured to teach the game at different proficiency levels. The simplest one is to only explain the rules: what's possible and what's not,

[2] *Gloomhaven: Jaws of the Lion* (Cephalofair Games, 2020) used this system to progressively explain the rule-set through 5 consecutive game scenarios.

how is the game played and how the winner is decided. An intermediate approach would require the AI to provide advice and reasoning behind certain decisions at specific points during the game; for instance, that is important to play a certain card because it gives the player further maneuvering in subsequent turns. Finally, an approach that would require a higher skill is to teach the player good strategies, which inform the play-through from start to the end. In this case, the ability of an AI tutor to teach at one or another level can be closely related to its own ability to play the game as an AI player, thus the research into this area could benefit from previous works on AI for game playing.

- **Scenarios:** Teaching can be achieved through complete playthroughs, from game setup to game end, or via different scenarios. In game AI terms, the AI tutor could choose interesting mid-point game states for the player to play in. This is similar to puzzles in Chess, where the player needs to identify the correct move to get to a chess mate or escape from one. An AI capable of automatically generating interesting scenarios would be able to produce valuable *teachable moments*, where expected actions or common mistakes can be identified and corrected.
- **AI Interactions:** When the AI tutor monitors the actions and progress of the learner who is playing the game, a consideration needs to be made about how does the tutor interact with the player when they infringe the rules or take a particularly good or bad action. For the former, it is sensible to think that the AI tutor should intervene immediately to stop a rule for being broken, probably providing extra information about the rule itself and how it was violated. For the latter, however, different approaches could be considered. We could, again, interrupt the game when a mistake is made, or to provide reinforcement for a good move being executed. This could, however, turn the playing session tedious if multiple mistakes are made, so an alternative would be to provide a retrospective summary of good and bad decisions. The AI tutor would recap the "highlights" where the player made a particularly interesting choice (either good or bad) and provide extra information on the quality of the action. Nevertheless, it is important to not lose sight of the player's psychology at this point – a recapitulation of every time a novice player makes a mistake may not be received kindly by some learners.
- **Learning a Human-model:** Following up from the previous point, it is interesting to discuss if the AI tutor should build a model of the human learner. An important aspect to consider could be the prior knowledge of the player about similar and dissimilar games. This could be useful to draw similarities to dynamics found in other games[3]. Additionally, the AI tutor may also require to model the learning process of the player, for instance to identify which rules have already been learned (therefore no more emphasis should be put on them) and which ones have not. An interesting consequence of this is to estimate the learning progress of the player through the session, to change strategy in case the learning is too slow (provide simpler or fewer rules at a time) or too fast (increase the cognitive *load* required so the full game can be learned earlier).
- **AI Priors:** The previous point identifies prior knowledge from the point of view of the player, but the AI system may also have a source of prior knowledge to use. This can come with regards to the game being explained, either as an expert system that provides a series of rules, information taken from the manual, or outsourced from online resources such as forums or Fandom Wikis. If the AI tutor has been used previously to teach

[3] For instance, saying that "the phases in *Terraforming Mars: Expedition Ares* (FryxGames, 2021) are chosen using a similar mechanic to the ones in *Roll for the Galaxy* (Rio Grande Games, 2014)" can be a useful teaching tactic.

the same game, information about previous teaching sessions can be used as data to signal common mistakes or interesting teaching moments identified in earlier attempts. Finally, one could also attempt to extract useful data from other similar games that share common characteristics: for example, the use of *workers* in EuroGames, which is a similar feature in games like *Everdell* (Starling Games, 2018), *Village* (Eggertspiele, 2011), *Istanbul* (Pegasus Spiele, 2014), and many others.

- **Game-play Traces:** An interesting possibility that can aid teaching is extracting information from old games, for instance through the analysis of game-play traces. These can be taken from proficient players, AI agents or the actual human learner. This analysis can be done both in the short-term (actions taken in particular game states with an immediate effect) or in the long-term (actions taken in a game state that have a noticeable effect several turns later). Especially interesting decisions would provide valuable teaching moments for the learner, such as identifying particularly strong moves or missed opportunities where a better action could have been taken. By means of self-play, these missed opportunities can be more illustrative by simulating alternative future states that could have been reached if the better action had been chosen.

In summary, we observe a great possibility for Game AI research in this domain. Stemming from the previous enumeration, we can identify several areas of research that can benefit from work in this area, such as question-answering, generation of interesting situations for teaching and training (game states), identification of teachable moments (game states and actions), detection of weak and strong moves, and the design of interfaces for the communication between the two parts. All this should be investigated in conjunction with building models of the human learner, while adapting the teaching mechanisms to the prior knowledge, progress rate and habits of the player. Research in this area would also shed some light in the capabilities of the current state-of-the-art AI methods, especially in what refers to the Human-AI interaction, and whether it is necessary to design different AI tutor systems for distinct types of games and players.

References

1 Erik Andersen, Eleanor O'rourke, Yun-En Liu, Rich Snider, Jeff Lowdermilk, David Truong, Seth Cooper, and Zoran Popovic. *The impact of tutorials on games of varying complexity*. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pages 59–68, 2012.

2 Batu Aytemiz, Isaac Karth, Jesse Harder, Adam M Smith, and Jim Whitehead. *Talin: a framework for dynamic tutorials based on the skill atoms theory*. In Fourteenth Artificial Intelligence and Interactive Digital Entertainment Conference, 2018.

3 Raluca D. Gaina, Martin Balla, Alexander Dockhorn, Raul Montoliu, and Diego Perez-Liebana. *TAG: A Tabletop Games Framework*. In Experimental AI in Games (EXAG),AIIDE 2020 Workshop, 2020.

4 Michael Cerny Green, Ahmed Khalifa, Gabriella AB Barros, Tiago Machado, Andy Nealen,and Julian Togelius. *Atdelfi: automatically designing legible, full instructions for games*. In Proceedings of the 13th International Conference on the Foundations of Digital Games, pages 1–10, 2018.

5 Michael Cerny Green, Ahmed Khalifa, Gabriella AB Barros, and Julian Togellius. *"Press space to fire": Automatic video game tutorial generation*. In Thirteenth Artificial Intelligence and Interactive Digital Entertainment Conference, 2017.

6 Diego Perez-Liebana, Jialin Liu, Ahmed Khalifa, Raluca D Gaina, Julian Togelius, and Simon M Lucas. *General video game ai: A multitrack framework for evaluating agents, games, and content generation algorithms.* IEEE Transactions on Games, 11(3):195–214, 2019.

7 Lev Poretski and Anthony Tang. *Press a to jump: Design strategies for video game learnability.* In CHI Conference on Human Factors in Computing Systems, pages 1–26, 2022.

3.10 Artificial Intelligence for Alternative Controllers

Lisa Rombout (Tilburg University, NL), Alex J. Champandard (creative.ai – Wien, AT), Ahmed Khalifa (University of Malta – Msida, MT), Paris Mavromoustakos Blom (Tilburg University, NL), and Mark J. Nelson (American University – Washington, US)

Most popular game controllers are relatively straightforward – a collection of buttons, perhaps some touchpads and scroll wheels, a joystick. These inputs give clear, unambiguous signals to the system, and are generally designed to be as unobtrusive as possible. The user should be able to interact so naturally with the controller that they are almost able to forget it exists.

However, controllers can also be an integral part of the gameplay experience, facilitating play that is a combination of physical and digital. Popular examples include the guitar from Guitar Hero and the dance pad from Dance Dance Revolution. Currently widely available hardware platforms, such as Arduino, have made it very easy and accessible to extend a digital game with real-world sensors and actuators of various kinds. As basically anything conductive can be made into a sensor (including bowls of custard, bananas, and people), it follows that anything conductive can be made into a controller. The challenge, then, is to make sense of the data coming in to the system, and make sure the interaction can run smoothly.

We explored the idea of using an Arduino device to a) monitor a player's heart rate in real time and b) translate the heart rate measurement into the rhythm (beats per minute) of a custom-made music tune. This concept could be employed in an adaptive stress management game, where the player is rewarded for maintaining their heart rate at a rest-state level.

3.11 Quality Diversity for Procedural Content Generation

Jacob Schrum (Southwestern University – Georgetown, US), Alex J. Champandard (creative.ai – Wien, AT), Guillaume Chanel (University of Geneva, CH), Amy K. Hoover (New Jersey Institute of Technology, US), Ahmed Khalifa (University of Malta – Msida, MT), Mark J. Nelson (American University – Washington, US), Mike Preuß (Leiden University, NL), and Vanessa Volz (modl.ai – Copenhagen, DK)

Quality diversity algorithms [8, 1, 5, 6] are well-suited for generating game content [4], because most often there is no single optimal piece of content: level, texture, character design, etc. Rather, games need a variety of content, but all of that content needs to be of a reasonably high quality, hence the usefulness of QD algorithms. One of the more popular QD algorithms is the Multi-dimensional Archive of Phenotypic Elites (MAP Elites [6]), which collects an organized archive of diverse but high-quality solutions. Many variants of MAP-Elites exist [3, 7], but all rely on some sort of archive with a user-specified structure.

Unfortunately, there are unresolved questions that make variants of MAP Elites difficult to use, even for skilled practitioners. For any given domain, expert knowledge is required to define potentially beneficial behavior characteristic dimensions, which determine the number of dimensions in the archive. It is unclear how many dimensions should be used, and how many intervals should exist along each dimension. Several experiments were proposed to help guide practitioners in dealing with these issues, particularly for games.

3.11.1 Proposed Experiments

Here are a list of specific and detailed experiments considered as part of the seminar.

3.11.1.1 Restrict Archive Based on Real Game Data

It is assumed that good games already feature a good variety of content. Although interesting content could be discovered by searching outside the bounds of the existing content, there is also a risk of wasting considerable computational resources discovering a wide variety of uninteresting content. Not only is it computationally expensive to search a larger space, but the cost in human effort to analyze an unnecessarily large archive can be prohibitive. Therefore, we propose a way of restricting an archive to focus on areas likely to be relevant to designers, by using existing game content as a basis. This experiment can readily extend previous work applying MAP Elites to evolve levels for Super Mario Bros. and The Legend of Zelda [9], but can be applied to any game or content.

First, behavior characteristics and archive structures taken from previous literature can be used to store original levels from these games. If at least a majority of the levels occupy distinct bins in the archive, it means that the archive is able to capture the diversity of content in the original game. Intervals along different dimensions of the archive could be adjusted to accommodate more of the original game content if levels that share a bin are deemed to be sufficiently different.

Second, the archive storing the original game content is contracted, so that bins outside the boundaries of what is present in the original game are excluded. Specifically, for each archive dimension, the minimum and maximum values are calculated, and the empty bins outside this range are deemed unreachable.

Third, this restricted archive is used to evolve new content with MAP-Elites or one of its variants. Whenever a level is generated that would fall into one of the unreachable bins, it can simply be discarded, or penalized in some way. This restricts the range of what evolution can produce, but it also means that there are no levels in the unreachable bins that can be chosen as a parent for a new level. Therefore, the search will be more focused on what are presumed to be the most relevant areas of the archive.

Finally, results with such restricted archives would need to be compared with results from unrestricted archives. Although an unrestricted archive will likely contain many more occupied bins, we can ask several pertinent questions: 1) which approach does a better job of filling the restricted/contracted portion of the archive with quality solutions? 2) which approach fills this restricted/contracted area faster? 3) How diverse and interesting are levels from outside of the restricted/contracted area? This last question is perhaps the most difficult to answer due to a lack of widely accepted definitions for *diverse* and *interesting*, though a human subject study could help in this assessment. The first two questions can be directly answered using objective measures such as the difference in QD score and archive occupancy over time.

The results of such experiments would be informative, regardless of how they turn out. If restricting the archive allows a pertinent area to be filled better or more quickly, then it means that researchers and practitioners can stop wasting effort on large archives. However, if searching a larger archive actually makes it easier to fill a restricted portion of the archive, it would imply that the extra bins are providing useful stepping stones for searching the evolutionary space. It is possible that these outcomes will be dependent on the domain and archive structure being used, but this outcome would also be informative.

3.11.1.2 Evolving Levels With a Wide Range of Archive Dimensions

There are many properties of content that can be objectively measured, and in principle, any such property can be used as a dimension in a behavior characterization. For a content designer, it is unclear what properties to incorporate into a behavior characterization, and how many, but a designer will presumably have access to many candidate properties worth considering.

This experiment proposes a way of assessing how useful different numbers of dimensions in a behavior characterization are. We can start with a game like Super Mario Bros. for which past work [11] has provided a wide range of level properties that can be measured, and which seem to impact the user experience of a level in meaningful ways.

Given some large set of level properties, a way of using each property in a behavior characterization and within an archive is needed. Given this starting point, all combinations of the different properties within a behavior characterization can be considered. Separate experiments can be conducted with each behavior characterization.

The challenge is in comparing these results in order to produce recommendations for designers on which types of properties and how many to include in a behavior characterization. For any given archive of evolved MAP Elites solutions, one can take those archive's members and transfer them to another archive. If the dimensions used in the target archive are a strict subset of those from the source archive, one would expect that some number of solutions are lost as bins along now missing dimensions are collapsed into one bin within the target archive. The degree of this loss can be measured, and the discarded solutions can be analyzed to see if anything of value was actually lost from using a simpler archive.

Also, results evolved with smaller archives can be compared against those transferred to the same archive structure from a larger archive to see if focusing on fewer behavior dimensions leads to higher quality, at least within that particular range.

A more daunting task is comparing archives whose defining behavior characteristics are made up of disjoint sets of properties. One can still transfer solutions between the different archive structures to track how many solutions are lost in the transfer, but it is less clear what to make of such results. If one archive can contain all of the solutions of another, then it implies that some aspect of the source archive might be superfluous, but it is easily possible for significant loss to occur when transferring in both directions, so further analysis will be needed to see what this information can actually teach prospective users of MAP Elites for games.

3.11.1.3 Diversity of Content and of Playing Styles

The behavior characteristics used to evolve diverse game content using QD methods often include measures of playing style, estimated by simulated AI players. For example, we can evolve levels that require a lot of jumping to clear, and little to no jumping to clear (and everything in between). The result of such evolution is a set of diverse content that supports diverse playing styles – diverse according to whatever measures of play style were chosen for the behavior characterizations.

This approach assumes a 1-to-1 mapping between specific pieces of content and the playing styles they support (at least in the sense of reducing any variation to a single numerical estimate per behavior characterization). Therefore, we evolve a range of content to support a range of different playing styles. However, a range of playing styles can often be supported by a single piece of content, such as a level that is replayable in several very different ways. This leads to a second place we can apply QD methods: to investigate diversity of playing styles supported by fixed pieces of content, and what implications that has for PCG.

To run concrete experiments on diverse playing styles, it is most straightforward to choose a specific parameterized representation for the simulated player. For example, one of the demos of the `pyribs` implementation of CMA-ME [2] evolves agents for Lunar Lander that complete the level with a range of x positions and y velocities.[4] That demo uses a linear policy representation. Linear policies are a good place to start, although evolving weights of a fixed neural network is another option.

The result of running QD on Lunar Lander shows that the game supports a wide range of playing styles (types of landing), which in this case is probably key to the game's popularity. Should this be an explicit target for PCG? In addition to evolving diverse content, should individual pieces of content be generated with a goal that each one also supports diverse playing styles? That is less clear; it may in fact be better in some cases to retain a clearer mapping between individual pieces of content and desired playthroughs. For example, in an educational game it may explicitly *not* be desirable, if the goal of a level is to force the use of a specific concept being introduced [10].

The open question here is: What is the relationship between diverse game content and diversity of play styles supported by a single piece of game content?

3.11.1.4 Conclusion

Although MAP Elites and its variants are powerful tools for creating diverse collections of quality content for games, more research is needed to understand how to apply these tools and how to evaluate the artifacts they produce. In particular, the relation of produced to existing game content is important here. This working group produced several intriguing ideas for future research along these lines.

References

1 Konstantinos Chatzilygeroudis, Antoine Cully, Vassilis Vassiliades, and Jean-Baptiste Mouret. Quality-diversity optimization: A novel branch of stochastic optimization. In *Black Box Optimization, Machine Learning, and No-Free Lunch Theorems*, pages 109–135. Springer, 2021.

2 Matthew C. Fontaine, Scott Lee, Lisa B. Soros, Fernando de Mesentier Silva, Julian Togelius, and Amy K. Hoover. Mapping hearthstone deck spaces through map-elites with sliding boundaries. In Anne Auger and Thomas Stützle, editors, *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2019, Prague, Czech Republic, July 13-17, 2019*, pages 161–169. ACM, 2019.

3 Matthew C. Fontaine, Julian Togelius, Stefanos Nikolaidis, and Amy K. Hoover. Covariance Matrix Adaptation for the Rapid Illumination of Behavior Space. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, New York, NY, USA, 2020. ACM.

[4] `https://docs.pyribs.org/en/stable/tutorials/lunar_lander.html`

4 Daniele Gravina, Ahmed Khalifa, Antonios Liapis, Julian Togelius, and Georgios N Yannakakis. Procedural content generation through quality diversity. In *2019 IEEE Conference on Games (CoG)*, pages 1–8. IEEE, 2019.

5 Joel Lehman and Kenneth O. Stanley. Evolving a diversity of virtual creatures through novelty search and local competition. In Natalio Krasnogor and Pier Luca Lanzi, editors, *13th Annual Genetic and Evolutionary Computation Conference, GECCO 2011, Proceedings, Dublin, Ireland, July 12-16, 2011*, pages 211–218. ACM, 2011.

6 Jean-Baptiste Mouret and Jeff Clune. Illuminating search spaces by mapping elites. *CoRR*, abs/1504.04909, 2015.

7 Olle Nilsson and Antoine Cully. Policy gradient assisted map-elites. In Francisco Chicano and Krzysztof Krawiec, editors, *GECCO '21: Genetic and Evolutionary Computation Conference, Lille, France, July 10-14, 2021*, pages 866–875. ACM, 2021.

8 Justin K Pugh, Lisa B Soros, and Kenneth O Stanley. Quality diversity: A new frontier for evolutionary computation. *Frontiers in Robotics and AI*, page 40, 2016.

9 Jacob Schrum, Vanessa Volz, and Sebastian Risi. CPPN2GAN: Combining Compositional Pattern Producing Networks and GANs for Large-scale Pattern Generation. In *Genetic and Evolutionary Computation Conference*, New York, NY, USA, 2020. ACM.

10 Adam M. Smith, Eric Butler, and Zoran Popovic. Quantifying over play: Constraining undesirable solutions in puzzle design. In *Proceedings of the Foundations of Digital Games Conference*, pages 221–228, 2013.

11 Vanessa Volz. *Uncertainty Handling in Surrogate Assisted Optimisation of Games.* PhD thesis, TU Dortmund University, Germany, 2019.

3.12 Benchmarking Coordination Games

Pieter Spronck (Tilburg University, NL), Duygu Cakmak (Creative Assembly – Horsham, GB), Jakob Foerster (University of Oxford, GB), and Setareh Maghsudi (Universität Tübingen, DE)

Games are often used in AI research as benchmarks for new technologies and developments. For instance, a breakthrough application of deep convolutional neural networks and Monte Carlo tree search was in the game of Go, where the program AlphaGo was able to defeat the human world champion by using the aforementioned techniques [3]. Competitions between different AI approaches to play particular games are common, not only for classic deterministic two-player board games such as Chess, but also for modern board games and video games.

Commonly, the games used are competitive games. One reason why this is the case, is that for competitive games it is relatively easy to determine which AI is "the best," namely the one that wins the most or scores the highest. However, competition only covers part of what AI needs to do. In the modern world, where AI get increasingly integrated in people's lives, the ability of AI to cooperate effectively (with humans or with other AIs) is of great interest to researchers.

If the environment in which cooperation is required is fully observable, the solution to problems of cooperation is, in general, to let the smartest AI propose the most effective approach, and let every participant follow that approach. However, in practice cooperation problems are often only partially observable, either because the participants have only partial information of the environment or because participants have individual goals.

To research AI which is able to cooperate, we therefore need partially observable, fully cooperative benchmarks. Games can be such benchmarks.

3.12.1 Examples of coordination games

A typical example of a coordination game is Hanabi, a game in which players cooperatively need to achieve a goal by playing cards, whereby they have knowledge of the cards of the other players but not of their own cards, while their ability to communicate is limited to very specific statements that they can make [1]. If AIs learn to play Hanabi with only other AIs in the mix, they will discover strong strategies to play the game, reaching a very high cooperative score. However, such strategies are not used by humans, so different approaches must be used to create AIs which are able to cooperate with human players.

Hanabi is a partially observable, turn-based game for 3 to 5 players, who all have a different observation space. In the previous iteration of this series of Dagstuhl Seminars, the "Guess 100" game was developed, which is a fully observable, simultaneous-move game for 2 players. In the Guess 100 game, the two players simultaneously choose a number between 1 and 100; the two numbers are then revealed to the two players, after which they choose a new number. There is no other communication between the players. They continue doing this until they chose the same number, with the ultimate goal to keep the number of turns needed as low as possible.

Another example of a well-known, popular coordination game is Codenames, where a turn for one team with a team leader can be considered a coordination task which is partially observable for the team and fully observable for the team leader, whereby the team leader attempts to give a one-word hint which directs the team to select from a grid of cards those cards which belong to the team (without the team being able to see which cards are theirs).

During the pandemic, a digital version of the board game Wavelength became highly popular. In this game one player, the Psychic, gets two words which are extremes on a range, such as "hot" and "cold." The Psychic also gets a "bullseye", a spot on the line between the extremes. The Psychic needs to give a clue (with certain limitations to what the clue can be) which indicates where the bullseye is. The players then indicate a spot on the line between the two extremes, and the closer they are to the bullseye, the more points they score. Like Codenames, this is a partially observable, turn-based game.

3.12.2 Axes of coordination

The goal of the workgroup was to determine different coordination games, preferably games that are simple to implement, play, and understand, which cover different coordination problems. We therefore started by determining the different "axes of coordination." We arrived at the following list:

- Fully observable vs. partially observable
- Simultaneous moves vs. sequential moves
- Labeled vs. unlabeled states and actions
- Type of labels (e.g., ordered, unordered, semantic)
- Turn-based vs. real-time
- Iterated vs. single-shot
- Shared vs. different observation spaces
- 2-player vs. 3+ players

3.12.3 Variations of the Guess 100 game

As the Guess 100 game is close to the simplest implementation of a fully-observable, simultaneous-move, 2-player game with ordered labels for actions, it makes sense to use it as a template for game variations which touch different axes of coordination.

To turn Guess 100 into a partially observable game (leaving the other axes of coordination unchanged), the two players do not aim to find the same number, but to land on a particular target number that is different for each of them, whereby the target number for each player is known to the other player, while unknown to themselves. The players are not allowed to select the target number of the other player. The game ends when they simultaneously land on their assigned target number.

To turn Guess 100 into a game without ordering, the players would not select numbers but icons, and the icons are ordered differently for the two players, so that even the location on the grid cannot be considered an ordering.

To turn Guess 100 into a game without labels or ordering at all, the playing field can consist of a number of moving balls which are unmarked. The players' goal is still to select the same ball, and they get to see which ball was selected by the other player when both have selected a ball.

3.12.4 The Color-coder game

Since an integral part of the Guess 100 game is that the players move simultaneously, we continued by designing a game that is partially observable, has sequential moves, and no ordered labels. The aforementioned Codenames is such a game, but too complex to be a good basis for fundamental research. We wanted this game to be as simple as possible. We came up with the Color-coder game.

In the Color-coder game there are two players, one of which has the role of "hinter" and the other one the role of "guesser." The hinter observes a "code" which consists of two colored tokens, randomly selected from four colors, e.g., RED-BLUE. The hinter and guesser now take turns, starting with the hinter. The turn of the hinter consists of providing a sequence of black and white tokens, e.g., BLACK-BLACK-WHITE-WHITE-BLACK. The turn of the guesser consists of producing a guess, which consists of two colored tokens, e.g., YELLOW-GREEN. No other communication between the hinter and guesser is allowed. The game ends when the guesser reproduces the code that the hinter observed. The goal is to minimize the number of turns needed for that.

The Color-coder game sounds like Mastermind, but it is substantially different. In Mastermind the meaning of the black and white tokens is predetermined, and the hinter's role is not to cooperate with the guesser, but to simply provide the predetermined hint. In the Color-coder game, the hinter actively wants to lead the guesser to the correct code, and can try to supply the guesser with more information. The hinter can decide upon a particular interpretation of the tokens and stick to it, but might also change the meaning of the tokens while playing.

We tested the Color-coder game several times, and found that different strategies were employed. Often the hinter chose a strategy before the game started, and did not diverge from it, hoping that the guesser would pick it up. Sometimes the hinter changed their way of hinting during the game. In principle there is no reason for the game to last longer than four turns, if the hinter simply always uses black to indicate correct and white to indicate incorrect, and then always places two tokens. However, with smart hinting fewer turns are needed.

One hinter came up with the idea to hint at the correct code by using black tokens for the left color and white tokens for the right color, and then placed as many of the respective tokens as there are letters in the color name. Considering that in our default game we used red, blue, green, and yellow, the colors were encoded in a unique way. If the guesser would pick up on the approach, one guess would suffice.

The Color-coder game felt as slightly too simple to give rise to interesting communication strategies. It can be made more interesting without increasing complexity too much by increasing the number of colors.

3.12.5 The Convergence and Divergence games

We also wanted to define a game that, like the Guess 100 game, is fully observable and has simultaneous moves, but has no numerically ordered labels, because with a numerical ordering players will use calculations to try to reach the same number.

In the Convergence game the players are presented with a list of ten words. The words are randomly selected from a dictionary, e.g., zoom, understood, women, income, joke, scrawny, waiting, bucket, picayune, camera. The list may be presented to the players in a different order, so that the place on the list is not part of the ordering; there is, of course, a semantic ordering. The players simultaneously select a word. If the word is the same, the game ends. If not, they select a different word, whereby the words that were selected before cannot be selected again. The goal is to select the same word as fast as possible.

The players can use the semantic ordering of the words to decide which next word to select. E.g., if from the previous list one player selected "understood" and the other one "women", then they might decide to select "waiting" as their next guess as that is the only word which is between the two previously selected ones. However, with numbers such an approach is far more "natural" than it is with words, and players are more likely to use the "meaning" of words to direct their guesses.

However, we found a slight change to the Convergence game made it much more interesting: in the Divergence game the players undertake the same actions, but they try to avoid selecting the word that the other player selects. As soon as they land on the same word, the game ends, and they try to postpone this as long as possible. With ten words, the game can last no more than five turns, but the game can easily be extended by making the list of words longer, which would also allow the players to try to communicate more as they can select more words before the list becomes so short that the risk of selecting the same word is high.

3.12.6 Next steps

We implemented a digital version of the Guess 100 game during and after the previous Dagstuhl Seminar. We want to use it to collect data on game plays. It can also be used to implement variations of the Guess 100 game, to work on different axes of coordination. The datasets developed this way should be open-sourced. We then want to develop AIs which play these games, both with other AIs and with human players. The most interesting games can form the basis for a challenge paper.

References

1 N. Bard, J.N. Foerster, S. Chandar, et al. *The Hanabi challenge: A new frontier for AI research.* Artificial Intelligence 280, 2020

2 K. Hofmann, D. Cakmak, P. Cowling, et al. Human-AI Coordination. *Artificial and Computational Intelligence in Games: Revolutions in Computational Game AI*, Dagstuhl Seminar 19511, 2020

3 D. Silver, A. Huang, C. Maddison, et al. *Mastering the game of Go with deep neural networks and tree search.* Nature 529, 484–489, 2016. https://doi.org/10.1038

3.13 Explainable AI for Games

Jichen Zhu (IT University of Copenhagen, DK), Maren Awiszus (Leibniz Universität Hannover, DE), Michael Cook (Queen Mary University of London, GB), Alexander Dockhorn (Leibniz Universität Hannover, DE), Manuel Eberhardinger (Hochschule der Medien – Stuttgart, DE), Daniele Loiacono (Polytechnic University of Milan, IT), Simon M. Lucas (Queen Mary University of London, GB), Ana Matran-Fernandez (University of Essex – Colchester, GB), Diego Perez Liebana (Queen Mary University of London, GB), Tommy Thompson (AI and Games – London, GB), and Remco Veltkamp (Utrecht University, NL)

In recent years more and more research has been invested into eXplainable artificial intelligence (XAI) to make machine learning (ML) and AI models more trustworthy and understandable for users. In an earlier vision paper, a new research area for designers and game designers was proposed called XAI for Designers (XAID) [1], which focused on *mixed-initiative co-creation* [2] approaches to help designers better leverage AI methods through co-creation in their respective design tasks. Since then, much development has been made in XAI. In this working group, we investigate whether and how these new methods for XAI can also be used for games.

3.13.1 What is XAI for Games?

There are a large variety of possible use cases for XAI in games or game development, and this largely depends on what one wants to achieve. Some salient use cases include:

- Increasing the transparency for game AI decisions so that these decisions can be understood and trusted by humans.
- Explanations of key game AI decisions can be used as a feedback mechanism for how well a player is performing. For instance, a PCG-based educational game can explain to a player that a new level is generated based on her previous gameplay so that she can continue to practice a certain skill that she has not mastered. This type of explanation can be used as a feedback mechanism to foster player reflection and learning[6].
- Tools for framing in computational creativity and improving the design experience with mixed-initiative co-creativity systems.
- Highlighting to players why a given strategy is relevant, optimal, or exciting.

To further narrow the focus on the different use cases, in this report, we will focus on procedural content generation with ML (PCGML).

3.13.2 Case Study: Mario Level Generation

First, we looked at different possibilities to generate Super Mario levels. TOAD-GAN [3] can be trained using only one example. This method also makes it possible for users to control the output of the generation process by changing the noise vector that represents the input of the generator network. Since noise vectors cannot be interpreted by designers, designers still do not have the ability to design content according to their needs. To accomplish this, one must make the noise vector explainable to designers and map the different areas of the noise vector to the content that would result from a change in the noise vector.

Another method for generating Super Mario levels uses an evolutionary algorithm with tilesets [4]. The tilesets enforce consistency of the output, and the Kullback-Leiber Divergence

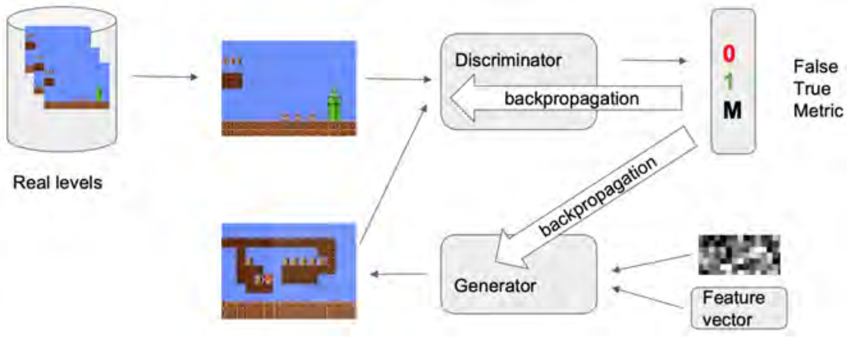


Figure 9 An overview of DOOMGAN architecture.

enables for control of variation and novelty. This method is explainable by design as the history of the gene values and the time steps of the mutation operators could be used to identify *when* something occurred, and *why* it was picked to be modified.

3.13.3 Case Study: DOOMGAN – Improving the PCGML Interpretability by Incorporating Metrics

PCGML[5] has been successfully applied to several kinds of game content. However, it generally has low interpretability to human designers because how the input (e.g., parameter/feature vectors) leads to generated content (or corresponding gameplay metrics) is often opaque. Recent Deep Learning-based generative models exacerbate this problem due to their complexity and blackbox nature. As a relevant case of study, we focused on GAN-based PCGML approaches and proposed to incorporate gameplay metrics (e.g., completion time, win rate) in part of the GAN architecture at the level of the discriminator. Figure 9 provides an overview of the proposed GAN architecture, dubbed DOOMGAN, where the discriminator is extended by adding one or more gameplay metrics as additional outputs. Our research hypotheses are that this method will 1) improve the interpretability of the system by providing meaningful intermediate output to designers and 2) improve the performance of the generative model (e.g., better data quality and data efficiency). Moreover, with the proposed method, existing XAI techniques, such as Saliency Map, LIME, and DeepSHAP, can be used to further open the blackbox of PCGML. An ideal testbed to investigate our ideas would be to extend one of the Mario level generators based on GAN previously introduced in the literature (e.g., TOAD-GAN[3]). To the best of our knowledge, this is among the first approach that connects XAI to PCGML methods.

3.13.4 Open Problems

Explainable AI for games is still a nascent research area. Below we summarize some of the key open problems in this area:

- How to turn explainability into explanation and actionable explanations to players and/or designers?
- How does content representation affect explainability? (e.g., representing a Mario level as tiles vs. objects)
- Whom do we design the explainable system for? What do the human players, designers, or other stakeholders need? Current XAI methods only explain predictive models but not generative models.
- How to capture functionality/playability of a level in XAI, which is absent in image generation?

3.13.5 Conclusion

In summary, this working group found eXplainable AI to be a rich research topic to explore in the context of computer games. Making the underlying AI process more transparent can benefit a wide range of stakeholders, including players, game designers, game analytics/user researchers, and game producers. Since computer games are end user-facing, we believe exploring eXplainable AI in the context of games will expedite the transition from technical explainability to usable human-centered explanations.

References

1 Zhu, J., Liapis, A., Risi, S., Bidarra, R. & Youngblood, G. Explainable AI for Designers: A Human-Centered Perspective on Mixed-Initiative Co-Creation. *2018 IEEE Conference On Computational Intelligence And Games (CIG)*. pp. 1-8 (2018)

2 Yannakakis, G., Liapis, A. & Alexopoulos, C. Mixed-initiative co-creativity. *FDG*. (2014)

3 Awiszus, M., Schubert, F. & Rosenhahn, B. TOAD-GAN: Coherent Style Level Generation from a Single Example. *AAAI Conference On Artificial Intelligence And Interactive Digital Entertainment Best Student Paper Award* . (2020,10)

4 Lucas, S. & Volz, V. Tile Pattern KL-Divergence for Analysing and Evolving Game Levels. *Proceedings Of The Genetic And Evolutionary Computation Conference*. pp. 170-178 (2019), https://doi.org/10.1145/3321707.3321781

5 Summerville, A., Snodgrass, S., Guzdial, M., Holmgård, C., Hoover, A., Isaksen, A., Nealen, A. & Togelius, J. Procedural Content Generation via Machine Learning (PCGML). *IEEE Transactions On Games*. **10**, 257-270 (2018)

6 Zhu, J. & El-Nasr, M. Open player modeling: Empowering players through data transparency. *ArXiv Preprint ArXiv:2110.05810*. (2021)

3.14 Human-AI Collaboration Through Play

Jichen Zhu (IT University of Copenhagen, DK), Guillaume Chanel (University of Geneva, CH), Michael Cook (Queen Mary University of London, GB), Alena Denisova (University of York, GB), Casper Harteveld (Northeastern University – Boston, US), and Mike Preuß (Leiden University, NL)

3.14.1 Motivation

Human-AI collaboration is a rapidly growing research area. As AI becomes an integral part of the workplace as well as home, developing technology that can efficiently collaborate with humans is essential.

Existing psychology research found that successful collaborations between humans need the foundation of 1) a relational interaction (conflict, small talk, emotional exchanges, relationship construction) and 2) efficient cognitive interaction (e.g., building on others' ideas – transactivity, synthesis, building a common ground) [1]. However, in current *Human-AI interaction* (HAI) research, this social-cognitive element and the social experience between human users and the AI is under-explored. This is problematic because since most users, especially novel users of AI, tend to approach AI based on their knowledge of similar human interactions [9, 8].

We argue that computer games, and playful interactions around games, provide a platform to explore new forms of collaborations and social interactions that consider relational aspects and are more cognitively nuanced. Such AI agents can potentially lead to better outcomes such as deeper social engagement when a player collaborates with it towards a common goal (e.g., solving a puzzle or mixed-initiative co-creative game design [10]).

Many games are social by nature as they propose several mechanisms for collaborative and competitive play. In addition, it appears that watching a game together is already sufficient to provide a significant social experience as demonstrated by streamed games [11], probably being not that different from groups of people watching sports games. This implies that there is huge potential to explore human-AI collaboration through play.

3.14.2 Background on Collaborative Game AI

A particularly interesting collaborative AI player is OpenAI Five [7]. It is able to play the MOBA game Dota 2 at the human professional level. Teams in this game consist of 5 players, and close collaboration is mandatory for winning the game. The AI has learned a specific type of collaborative behavior that may be called generous self-sacrifice. It is able to play extremely well when playing as a pure AI team but did not manage to collaborate well with human players, most likely because humans play more selfishly.

In addition, [12] used intention recognition to infer the task the human player was performing at the moment so that the AI could provide the appropriate assistance to the player. In board games, [13] explored agents for games built on collaborative game mechanics between human players. The authors used Rolling Horizon Evolutionary Algorithm to develop an artificial agent to balance gameplay in *Pandemic*. [14] explored how procedural content generation, such as character generation, story sifting, and social simulation, can be used to facilitate collaborative storytelling among human players.

Finally, HCI researchers have used computer games as a platform to study how human perception and gameplay outcomes may be affected when interacting with an AI [15].

3.14.3 Design space of human-AI interactions

Social interaction can take several forms [2, 3], including:

- Competition: the various agents are competing on a limited amount of resources to accomplish goals that are generally orthogonal.
- Collaboration: collaboration is defined as the action of working together on a single shared goal which generally leads to joint and strongly coordinated behaviors.
- Cooperation: during cooperation, some goals might be distinct, and the agents generally dispatch sub-tasks among the group to only assemble the results at the end of the task.
- Mediation: during mediation, a single agent has the goal of reducing the amount of conflict between at least two other agents.

Researchers have found that different AI techniques support different collaborative interaction mechanisms. For example, [16] noticed that generative adversarial networks (GANs) require a modified set of interaction patterns compared to other generative models.

In the context of video games, most artificial agents are designed to be competitive, while some are able to collaborate with other artificial agents. However, AIs that are able to collaborate or cooperate with human players are more scarce. Cooperation would necessitate understanding which goals are shared, dividing the objectives into sub-tasks, and reaching a common agreement on the distribution of those tasks. Collaboration is more complex as there is a need to synchronize actions between the AI and the players at any time. This is achievable only by having a common understanding of the situation and reaching an agreement on which steps to take next. Finally, a very under-studied type of social interaction in games is mediation. Artificial agents could help humans to collaborate better by assisting them in organizing and avoiding conflicts. In all these interaction types, there is a need to measure the interaction to determine the best approach and actions to take next.

3.14.4 Measuring social interactions

Social interactions can be measured in real-time or a posteriori – after the actual interaction has taken place. The former might be useful to provide feedback to AI agents so they can adapt their behavior to the social situation, for instance, by being included as a reward in reinforcement learning. The latter would allow for evaluating the efficiency and outcomes of the proposed approach.

Questionnaires can be used to evaluate the interaction a posteriori but are more difficult to be administered during gameplay. The importance of players' social experience was acknowledged by the Game Experience Questionnaire [4], which includes a social presence module mostly inspired by the social presence theory. The competitive and cooperative presence in gaming questionnaire [5] allows researchers to capture the sense of social presence in different types of interaction.

Measuring a social interaction can also be achieved by measuring in-game (position, firing rate) and reactions outside of the game (for example, players' facial expressions, eye movements, and physiological signals). The advantage of this method is that it can provide insights both during and after the game. Several publications, including [6], have studied the possibility of using joint reactions to identify the type of interaction and collaborative processes. However, the use of in-game features to characterize game social interaction remains an under-studied research area. In addition, there is a need to investigate if these methods, including the usage of questionnaires, can be transferred from the context of human-human interactions to human-AI interactions.

3.14.5 Conclusions and Future Work

In conclusion, computer games and playful interactions are a particularly rich domain to explore and advance human-AI collaboration research. This includes both the technical research of how to build better collaborative AIs and the HCI research of how to design new forms of collaborative interaction between humans and agents.

References

1 Avry, S., Chanel, G., Bétrancourt, M., & Molinari, G. (2020). Achievement appraisals, emotions and socio-cognitive processes: How they interplay in collaborative problem-solving?. *Computers in Human Behavior*, 107, 106267.

2 Arnold, N., Ducate, L. & Kost, C. (2012). Collaboration or cooperation? Analyzing group dynamics and revision processes in wikis. *Calico Journal*, 29(3), 431-448.

3 Bogacz, F., Pun, T. & Klimecki, O.M. Improved conflict resolution in romantic couples in mediation compared to negotiation. *Humanities and Social Sciences Communications* 7, 131 (2020)

4 Jsselsteijn, W. A., de Kort, Y. A. W. & Poels, K. (2013). The Game Experience Questionnaire. *Technische Universiteit Eindhoven.*

5 Hudson, M., & Cairns, P. (2014). Measuring social presence in team-based digital games. *Interacting with Presence: HCI and the Sense of Presence in Computer-mediated Environments*, 83.

6 Chanel, G., & Mühl, C. (2015). Connecting brains and bodies: applying physiological computing to support social interaction. *Interacting with Computers*, 27(5), 534-550.
7 Berner, C., Brockman, G., Chan, B., Cheung, V., Debiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., Józefowicz, R., Gray, S., Olsson, C., Pachocki, J., Petrov, M., Pondé de Oliveira Pinto, M., Raiman, J., Salimans. T., Schlatter, J., Schneider, J., Sidor, S., Sutskever, I., Tang, J., Wolski, F. & Zhang, S. (2018) Dota 2 with Large Scale Deep Reinforcement Learning. `http://arxiv.org/abs/1912.06680`.
8 Zhang, R., McNeese, N., Freeman, G. & Musick, G. "An Ideal Human" Expectations of AI Teammates in Human-AI Teaming. *Proceedings Of The ACM On Human-Computer Interaction.* **4**, 1-25 (2021)
9 Myers, C., Furqan, A., Nebolsky, J., Caro, K. & Zhu, J. Patterns for how users overcome obstacles in voice user interfaces. *Proceedings Of The 2018 CHI Conference On Human Factors In Computing Systems.* pp. 1-7 (2018)
10 Zhu, J., Liapis, A., Risi, S., Bidarra, R. & Youngblood, G. Explainable AI for designers: A human-centered perspective on mixed-initiative co-creation. *2018 IEEE Conference On Computational Intelligence And Games (CIG).* pp. 1-8 (2018)
11 Sjöblom, M., & Hamari, J. (2017). Why do people watch others play video games? An empirical study on the motivations of Twitch users. *Computers in human behavior*, 75, 985-996.
12 Nguyen, T., Hsu, D., Lee, W., Leong, T., Kaelbling, L., Lozano-Perez, T. & Grant, A. Capir: Collaborative action planning with intention recognition. *Seventh Artificial Intelligence And Interactive Digital Entertainment Conference.* (2011)
13 Sfikas, K. & Liapis, A. Collaborative agent gameplay in the pandemic board game. *International Conference On The Foundations Of Digital Games.* pp. 1-11 (2020)
14 Kreminski, M., Acharya, D., Junius, N., Oliver, E., Compton, K., Dickinson, M., Focht, C., Mason, S., Mazeika, S. & Wardrip-Fruin, N. Cozy Mystery Construction Kit: prototyping toward an AI-assisted collaborative storytelling mystery game. *Proceedings Of The 14th International Conference On The Foundations Of Digital Games.* pp. 1-9 (2019)
15 Ashktorab, Z., Liao, Q., Dugan, C., Johnson, J., Pan, Q., Zhang, W., Kumaravel, S. & Campbell, M. Human-ai collaboration in a cooperative game setting: Measuring social perception and outcomes. *Proceedings Of The ACM On Human-Computer Interaction.* **4**, 1-20 (2020)
16 Grabe, I., González-Duque, M., Risi, S. & Zhu, J. Towards a Framework for Human-AI Interaction Patterns in Co-Creative GAN Applications. (2022)

4 Panel discussions

4.1 Discussion and Evaluation

Pieter Spronck (Tilburg University, NL), Setareh Maghsudi (Universität Tübingen, DE), and Diego Perez Liebana (Queen Mary University of London, GB)

On the last day of the seminar only, the participants gathered in the common room to have an evaluation and discussion of the seminar, and to look forward to a possible follow-up seminar. Multiple topics came up, which are discussed below.

4.1.1 Seminar setup

The setup of the seminar was as follows: every day, immediately after breakfast, participants gathered in the common lecture room, where workgroups were formed around themes proposed by the participants. These workgroups consisted of at least three and at most eight people (usually four or five). They worked on their respective themes for the day. This work could consist of a discussion, the building of a prototype, or the running of an experiment. Around five o'clock everyone gathered in the common lecture room for a plenary session, where each workgroup presented their achievements. Usually a workgroup ended after one day, but a few ran a second day, sometimes with different participants, sometimes with a variation on the theme.

After dinner, on Tuesday, Wednesday, and Thursday, an extra activity was planned: on Tuesday and Thursday this concerned a lecture/discussion led by one of the participants on a topic which was of general interest, while on Wednesday this concerned the playing of a game. This game was a roleplaying game to commemorate Daniel Ashlock, one of the organizers and a rather prolific member of the community, who passed away two months before the seminar took place. The game was based on a collection of ideas that Dan developed for roleplaying games (the game has been made available for free from `https://www.drivethrurpg.com/product/400792/Ashlocks-Maze`). Thursday evening also a pub quiz was held, and VR games were made available on several other evenings.

Dagstuhl recommends having a longer walk on one of the days. As was suggested during the previous seminar, we replaced that with a shorter, 45-minute walk, every day after lunch. Still, as many participants also desired a longer walk, time was set aside on Wednesday afternoon to have that.

Before, during, and after the seminar, we used Discord to communicate between participants.

4.1.2 Workgroups

For previous seminars we had participants approach the blackboard to write down ideas for workgroups, after which participants wrote their names next to some of those ideas, to decide which workgroups would start. A problem that was recognized with this approach, is that newcomers to the seminar or the research field might feel intimidated by this process and thus reluctant to bring their ideas forward. In an attempt to resolve this problem, we let participants write their ideas on sheets of paper which we collected, and then anonymously wrote down on the blackboard.

The big disadvantage of this approach was that there were many more ideas formulated than with the old approach (which is good), but with a lot of overlap between them. This made it harder to come up with a good set of workgroups to run, especially since we did not want to have ideas "get lost". In the end everything worked out, but that was also because the number of participants was smaller than for previous seminars. We had already more ideas than fit on six blackboards with this relatively small group.

It was noted that rather than using the blackboard to form workgroups, we could have used Discord for that. It may even be possible to let then people sign up for multiple workgroups and have a semi-automated process divide the workgroups over the seminar days so that every participant can attend an optimal number of workgroups that hold their interest.

One point of stress that was recognized is that participants found it hard to choose between workgroups as there were multiple that they wanted to be part of. When too many people wanted to be part of one workgroup, it was split up along lines of interest, where each "subgroup" worked on their own perspective on the theme. This led to a suggestion that we could actually spend part of the seminar on having most or all participants work on the same theme, but in randomly constituted workgroups, which at the end of the day would all present their own view on the theme. A possible enhancement to this idea is that the groups would reform halfway through the day to stimulate cross-pollination. This idea needs some consideration, as a disadvantage of it would be less freedom in choosing workgroups.

4.1.3 Evening program

One general remark was that the evening program was for many participants "a bit much." A possible reason why many felt this way, may be found in the effects of the recent COVID crisis, where people became less used to interacting with bigger groups for a long period of time. Especially the fact that the talks were immediately after dinner (which was deliberate, to still have social time afterwards) was deemed "intense." The organized games, however, were evaluated positively.

It was suggested that perhaps the Sunday evening could also be used for some activity, though this should be an "unofficial" activity as the seminar officially starts on Monday.

4.1.4 Relaxation time

The short walks after lunch were appreciated by many participants, although for most of the week it was rather hot and therefore not ideal for walks. Some participants asked for extra time to exercise. Ideally, such time should be at the end of the day, before dinner (because exercising right after lunch or dinner is not a good idea). This would shorten the time available for workgroups; therefore, a possible approach could be to leave out the short walk after lunch, and instead have the plenary session between four and five o'clock, and have the time after that available for a walk or exercising.

Optionally, the short walk after lunch could still take place, but it should start around 12.45, as we found that by that time most participants had already finished lunch. That way, not much time is lost from the afternoon sessions.

4.1.5 Recording

Considerable discussion time was spent on "recording the seminar." This discussion started with the suggestion that the plenary sessions at the end of the day could be recorded and made available to people who do not attend the seminar. The suggestion was extended with the idea that a short video which shows off the seminar as a whole (including morning sessions, workgroups, plenary sessions, and social interaction) could be used as a way to show to invitees who have not been at Dagstuhl before what the seminar is like, and make them enthusiastic about accepting an invitation.

There are, however, issues with this. A major issue is privacy: not everyone might agree to being recorded, and even if they agree, they might feel uncomfortable with it. Moreover, due to the friendly atmosphere, people tend to be open about their ideas and how they talk about them, but when a camera is present they may feel guarded and cautious.

Recording the plenary session may be a bridge too far, but there would be a lot of value in a 15-20 minute documentary on a next seminar as an advertising tool. This could, for instance, consist of some soundless recordings (with music and commentary) of moments during one day of the seminar, interspersed with brief interviews with participants and explanations about the seminar's setup. Participants who do not want to be recorded can wear visual labels which indicate that they "opt-out."

4.1.6 Invitation process

The invitation process was rather involved this time around. For previous seminars, we needed at most two rounds of invitations before the seminar filled up. This time, due to the COVID crisis, we had to be cautious in sending invitations, so that early in the process we were very limited in the number of invited participants. Later on this was expanded, and we could invite more participants. At some point the seminar was close to being completely full, but then we ran into a second problem: due to changes in policies at many universities and institutes (particularly in Asia and the US), rising fears of people traveling, and the war in Ukraine, many participants started to drop out again. We frantically invited new people late in the process, up to two weeks before the seminar took place, but that was so late that almost no one could make it.

We invited mostly people from Europe, but we also sent a good number of invitations to Asia and North America, and a few to South-America, Oceania, and Africa. We ended up with just over 30 participants, out of 45 that would have been possible. Over the course of the invitation process, we invited close to 100 people, of which more than half were women, and about half were 'junior' people. While in the end the majority of the participants were people who had visited an earlier seminar, we managed to bring in multiple new faces. Often these were people who had no idea what Dagstuhl Seminars were about, but got a recommendation from someone who was aware of the event.

4.1.7 Organization

As always, from the side of Dagstuhl the organization and support were excellent. We noted a few possible improvements, which were communicated to Dagstuhl. As for the scientific organization, we got one recommendation for a follow-up seminar, which was to place a ballot box in the common room where people can deposit ideas or potential complaints – not that there appeared to be anything to complain about, but the existence of such a box would take away hesitation in reporting complaints as it offers the possibility of anonymity.

4.1.8 Topics for a follow-up seminar

Three topic ideas were brought forth for a potential follow-up seminar: (1) multi-agent social games; (2) benchmarks for game AI; and (3) creativity for games. Clearly, the current group of participants would enthusiastically support a follow-up seminar.

Participants

- Maren Awiszus
Leibniz Universität Hannover, DE
- Cameron Browne
Maastricht University, NL
- Duygu Cakmak
Creative Assembly – Horsham, GB
- Alex J. Champandard
creative.ai – Wien, AT
- Guillaume Chanel
University of Geneva, CH
- Michael Cook
Queen Mary University of London, GB
- Alena Denisova
University of York, GB
- Alexander Dockhorn
Leibniz Universität Hannover, DE
- Manuel Eberhardinger
Hochschule der Medien – Stuttgart, DE
- Jakob Foerster
University of Oxford, GB
- Casper Harteveld
Northeastern University – Boston, US
- Amy K. Hoover
New Jersey Institute of Technology (NJIT) – Newark, US
- Ahmed Khalifa
University of Malta – Msida, MT
- Antonios Liapis
University of Malta – Msida, MT
- Daniele Loiacono
Polytechnic University of Milan, IT
- Simon M. Lucas
Queen Mary University of London, GB
- Setareh Maghsudi
Universität Tübingen, DE
- Ana Matran-Fernandez
University of Essex – Colchester, GB
- Paris Mavromoustakos Blom
Tilburg University, NL
- Mark J. Nelson
American University – Washington, US
- Mirjam Palosaari Eladhari
Södertörn University – Huddinge, SE
- Diego Perez Liebana
Queen Mary University of London, GB
- Mike Preuß
Leiden University, NL
- Lisa Rombout
Tilburg University, NL
- Jacob Schrum
Southwestern University – Georgetown, US
- Pieter Spronck
Tilburg University, NL
- Tommy Thompson
AI and Games – London, GB
- Remco Veltkamp
Utrecht University, NL
- Vanessa Volz
modl.ai – Copenhagen, DK
- Jichen Zhu
IT University of Copenhagen, DK