Report from Dagstuhl Seminar 24261

Computational Creativity for Game Development

Edited by

Duygu Cakmak[1], Setareh Maghsudi[2], Diego Perez Liebana[3], and Pieter Spronck[4]

1 **Creative Assembly - Horsham, GB,** duygucakmak@gmail.com
2 **Ruhr-Universität Bochum, DE,** setareh.maghsudi@rub.de
3 **Queen Mary University of London, GB,** diego.perez@qmul.ac.uk
4 **Tilburg University, NL,** p.spronck@gmail.com

Abstract

Developments in artificial intelligence are currently dominated by deep neural networks, trained on large data sets, which excel at pattern recognition. Variants of the "classic" deep neural networks have the ability to generate new data with statistical properties similar to the training set. Despite the impressive products of such creative artificial intelligence, the results are usually lacking in meaning. They contain mistakes that humans would avoid, and often produce content which is not functional. While the product of creative artificial intelligence can be used as a strong basis for humans to build upon, human intelligence and human creativity are almost always a necessary ingredient of the creative process. Moreover, the more relevant the meaning, purpose, and functionality of the product are, the less the creative process benefits from the involvement of artificial intelligence.

Game design and implementation are tasks which require a high amount of creativity, and which must lead to products which require a high amount of fine-tuned functionality. For example, a game "level" should not only look appealing, it should also be playable and it should be interesting to play. These are not features which can be acquired simply by "training on big data," which is what most developments in modern artificial intelligence are based on.

This report on the Dagstuhl seminar 24261 discusses to what extent modern artificial intelligence techniques can produce meaningful and functional game content.

Seminar June 23–28, 2024 – http://www.dagstuhl.de/24261
2012 ACM Subject Classification AI - Artificial Intelligence, HC - Human-Computer Interaction, MM - Multimedia
Keywords and phrases artificial intelligence, computational creativity, computational intelligence, game design, game development
Digital Object Identifier 10.4230/DagRep.14.6.1
Edited in cooperation with Spronck, Pieter

1 Executive Summary

Pieter Spronck (Tilburg University, NL)
Duygu Cakmak (Creative Assembly - Horsham, GB)
Setareh Maghsudi (Ruhr-Universität Bochum, DE)
Diego Perez Liebana (Queen Mary University of London, GB)

Developments in artificial intelligence are currently dominated by the deep learning technology, which generates deep neural networks, trained on large data sets, which excel at pattern

recognition. Variants of the "classic" deep neural networks have the ability to generate new data with statistical properties similar to the training set. Generative Adversarial Networks (GANs), such as used by DALL-E and Midjourney, may be used to generate original visual artworks based on a textual description of the desired output. Autoregressive language models, such as used by ChatGPT, use deep learning to produce text that is often indistinguishable from human-created text. Moreover, artificial intelligence techniques have been used to successfully generate music for many years, and researchers have also experimented with using deep learning to create cooking recipes, personalized fragrances, fashion, and more.

Despite the sometimes astonishing products of such creative artificial intelligence, the results are usually lacking in meaning. While DALL-E and Midjourney produce images that seem impressive, upon further inspection they contain many mistakes which humans would avoid. While ChatGPT can generate human-sounding text in a conversation, it often produces utter nonsense, and cannot write an original coherent story. And, as our own explorations of such techniques during Dagstuhl Seminar 22251 showed, GANs may produce computer game content which looks reasonable at first glance, but is ultimately neither functional nor playable.

While the product of creative artificial intelligence can often be used as a strong basis for humans to build upon, and may as such speed up the creative process, human intelligence and human creativity are almost always a necessary ingredient of the creative process. Moreover, the more relevant the meaning, purpose, and functionality of the product are, the less the creative process benefits from the involvement of artificial intelligence.

Game design and implementation are tasks which require a high amount of creativity, and which must lead to products which require a high amount of fine-tuned functionality. For example, a game "level" should not only look appealing, it should also be playable (i.e., it must be possible for most players to finish the level) and it should be interesting to play (i.e., the player should feel entertained by playing the level and should experience inherent motivation to finish the level). These are not features which can be acquired simply by "training on big data," which is what most developments in modern artificial intelligence are based on.

The goal of Dagstuhl Seminar 24261, Computational Creativity for Game Development, was to investigate to what extent modern artificial intelligence techniques can produce meaningful and functional game content, and what changes to or extensions of these techniques can improve this AI-driven creative process.

We like to point out that progress in this area is relevant for a wide range of applications outside the "games" domain. Creativity in artificial intelligence applies to many branches of industry and has a strong impact on society, in which artificially intelligent technology interacts with humans in many shapes and forms. We use games in our research because they are highly-complex but well-defined applications which form safe environments to experiment in. However, solutions found for creative problems in games are often transferable to domains outside games.

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 (many of which were taken up by workgroups):

- **Procedural Content Generation for Games**: Procedural Content Generation (PCG) systems include techniques and methods able to create different type of game elements, such as levels, rules, quests and characters, among many others. Research in PCG has been prolific in the last decade, but its presence in the games industry is still far from

ubiquitous. A particular interest is set on mixed-initiative systems, which give designers and artist authorial control of the created content and the direction of the algorithm that generates it.

- **Procedural Generation of Games:** An extension of the previous point that deserves its own separate area is the generation of complete games. Same as recent advances in generative systems for music, painting and long bodies of text, one can research how complete games can be generated from scratch. This includes elements like art, rules, characters, winning and losing conditions that normally form a game. Automatic generation of new mini-games can open an interesting space of research that merges multiple advances together, but also a useful tool for game designers that will be able to use the generated games as inspiration for new entertainment experiences.
- **Computational Creativity for Narrative Games:** A particular type of games that have become more popular in the last decade is that of narrative games. While there has been some work in using Computational Creativity methods to generate texts, the adaptation of these techniques to the game development process remains an open area for research.
- **Automatic Generation of Art in Games:** Art is an important part of digital games and takes multiple forms: 3D-models, textures, visual effects, animations, cut-scenes, and so on. Lately, multiple advances on the use of Computational Creativity have shown the capacity of generating different forms of art, such as images, videos and even 3D geometries. Examples of systems that generate art are DALL-E, Stable Diffusion, and Midjourney. Research can explore how these and other techniques can be used to generate art for games, including unexplored game art areas, in particular with regards to how this generation can be bound to specific games/genres/restrictions, how can it be integrated into the game development process, and how can we give designers authorial control and modification capabilities over the generated assets.
- **Procedural Generation of Audio for Games:** An important part of automatic generation of content refers to audio. From audio effects (footsteps, heartbeats, weather) to complete sound tracks (background music, melodies, singers), including the generation of different voices for human and non-human game characters, the space for computational creativity to generate this type of art is vast.
- **Computational Creativity for Game Playing:** An unexplored aspect and application of Computational Creativity is that of generating AI agents that play a game. Traditionally, the objective that leads AI agents in a game to play is to achieve victory, either by reaching a winning state or by maximizing the score they obtain in the game. Some efforts have been made to employ quality-diversity methods to generate different styles of play. Research may explore how can we harness the new developments in Computational Creativity to generate diverse play styles, including the generation of new strategies or tactics to play games in a different manner.
- **Computational Creativity for Affective Computing:** Affective computing is a discipline that bridges several domains, such as computer science, cognitive science, and psychology. It studies the implementation of systems that are able to express, identify, process and simulate human affects. Research may investigate computational creativity algorithms and methods to provide non-player characters with the possibility of expressing feeling and emotions in a convincing way. This includes, for example, facial expressions and body animations, and it can be applied to human or non-human characters.
- **Automatic Support of Game Development:** Traditionally, computational creativity and the automatic generation methods have focused their efforts on generating the

product that creative industries build – be this games (or content for games), art or music, among others. These methods may also be used to aid the process of game development. Examples of the application of this technology include computational creativity for automation of tasks, algorithms for automatic testing of development process (such as code, integration, animations and deployment), production chains and procedural development processes.

- **Ethical Considerations of Computational Creativity:** The ethical challenges of using computationally creative tools for applications such as game development should not be ignored. The use of automatically-creative tools may have negative effects on the need for artists and designers. Moreover, the automatic creation of games and game content may lead to ethically suspect products. Finally, biases that exist in art and data may be magnified when such products are used to automatically generate new products. These ethical considerations will be taken into account in all our explorations of advances in computational creativity.

More than a year-and-a-half passed between writing the proposal and running the seminar. We found it striking how many advancements had been made in the area of Computational Creativity for Games in that period. During the writing of the proposal we were personally convinced that we were proposing an important theme for the seminar. When the seminar took place, we knew that no other theme was this topical.

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 set up 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.

42 participants accepted our invitation to join the seminar; 40 of them attended. The participants were a good mixture of genders, countries of origin, junior and senior people, and people from academics and from industry. All participants engaged intensively with the seminar, and many expressed how happy they were with what we accomplished, making the seminar a great success.

2 Table of Contents

3 Working groups

3.1 AI for Voice Generation from Text

Maren Awiszus (Viscom AG - Hannover, DE), Filippo Carnovalini (VU - Brussels, BE), and Pieter Spronck (Tilburg University, NL)

Joint work of Betker, James; Wang, Z. et al.
Main reference "James Betker. Better speech synthesis through scaling. arXiv, 2305.07243, 2023"
URL https://arxiv.org/abs/2305.07243

Generative AI has proven more sophisticated over the last year, and this also includes speech synthesis. The quality of both Text-to-Speech (TTS) and Voice-to-Voice Conversion is now high enough that it can feasibly be used in commercial products. Services such as *Elevenlabs*[1] boast with generative voices using audio samples as short as 1 minute, and going up to 3 hours for the best quality. Such voice synthesis can be very useful in the context of games, for example, it would allow for a cheaper alternative to full voice acting, especially for smaller studios which otherwise couldn't afford it. Voice Conversion allows to convert from one voice sound to another, which could allow a game to have diverse voices for different characters, even though they have been spoken only by one person, or a TTS generator. This is especially interesting for customizable Player Characters, which usually do not have a voice actor attached due to the sheer number of possible options for their voice and dialogue options. Voiceover is also important for the visually impaired and a good, natural sounding narration would allow more people to experience otherwise completely text-based games. Lastly, games that want to procedurally generate their text, for example using Large Language Models (LLMs), by design can not be voice acted in any other way, other than procedurally. This also applies to user text input which could be actually voiced using a generative method.

Of course, using generative voice for the described applications comes with downsides. If AI Voice Acting is used in projects, which could reasonably have hired real voice actors, those actors have lost that income and, in the long run, fewer voice actors could be employed. Additionally, the generated voices are still not of the quality of real voice actors, so the final product can be worse than if real voice acting was used, as well as include generation artifacts and mispronounced words. Also should the generated or user input text contain harmful sentiments, the generative voice would still say them, which could be detrimental to the career of anyone who was willing to provide their voice as a base for generation, as well as the developers responsible.

In this workgroup, our aim was to see how far one can get using available open source models to create a game with generated voice acting, and if possible, speaking user input lines on the fly. Figure 1 shows the envisioned loop for voice generation. For the generative parts, we used *Tortoise TTS* [1] for Text-to-Speech and *Retrieval based Voice Coversion (RVC)* [2] for Voice Conversion. A demo visual-novel-type game was created using *Renpy*[3], which is based on the Python programming language. This was done in an effort to include the python-based TTS directly into the game more easily.

[1] https://elevenlabs.io

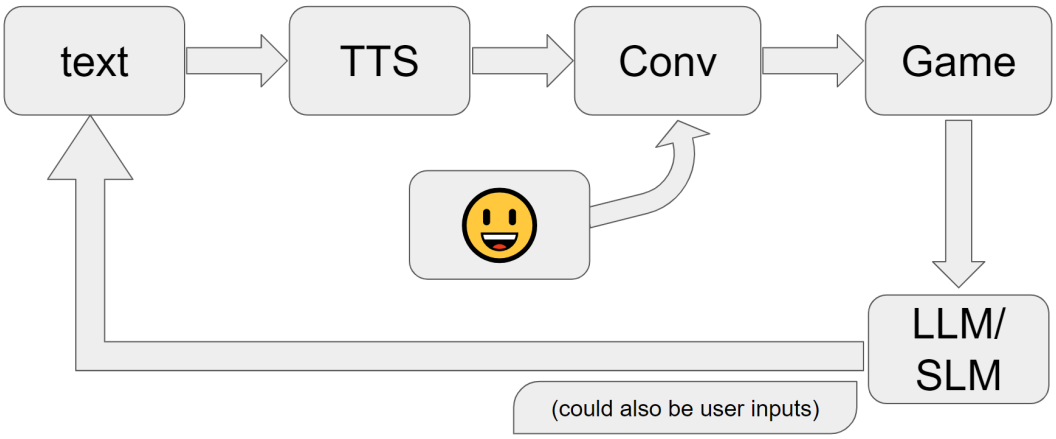


Figure 1 Generative Voice Acting Pipeline. The text of the game is fed to Text-to-Speech (TTS) which is then refined and/or transformed to a different voice using Voice Conversion (Conv) and can then be played by the game. The game or the player can then generate more text and the loop repeats.

3.1.1 Text to Speech

The Chosen TTS Model, *Tortoise TTS* [1] was used with the help of the ai-voice-cloning WebUI[2] which allowed for fast and easy testing. The Tortoise Model can be finetuned to mimic cadence and intonation, but with the limited time at the workshop, we chose to only use the voice adaptation feature. For that, we provided about 50 seconds of speech from one of the groups participants, whose tone the model was supposed to align itself to.

Given that there was no finetuning done, the generated voice lines only resembled the provided voice samples in tone, but not in cadence and intonation. Still, given the fact that this did not require any additional time to train and the generation time could be brought down to a few seconds on a Laptop, the results were promising.

3.1.2 Voice Conversion

In order to see how well the generated text could be spoken by different characters, we decided to use the voice lines generated with Tortoise TTS with different open source pretrained models in the *Retrieval based Voice Conversion (RVC)* WebUI [2]. The pretrained models were taken from a huggingface space[3], as there was no time to train an entire model from scratch during our workgroup.

The converted voice lines notably sounded like the characters those models were trained on and not like the original input. However, mannerisms and inflections were not generated, if not already present in the original sample. So while this does work in allowing for more diverse character voices, it is still not a replacement for the depth of having a voice actor impersonate a character fully.

[2] https://github.com/RVC-Project/Retrieval-based-Voice-Conversion-WebUI
[3] https://huggingface.co/juuxn/RVCModels

3.1.3 Demonstration

The generated voice lines were added to a small demo game (Figure 2) created with Renpy [3]. The idea was to have the user input be read out loud by the TTS used in the previous sections, which should be possible as they are both based on the Python programming language. However, due to unforeseen issues and limited time, we did not end up adding the functionality to the game. We suggest using a different approach if this is to be attempted again in the future.

For the purpose of the demo, we did put pre-generated lines of different characters into the game as audio files to show off the potential of the approach.

Figure 2 Demonstration game.

3.1.4 Conclusion

Given the small scope and time for our demo, our results for creating voice lines with Tortoise Voice adaptation and Voice Conversion for different characters are quite promising. With enough polish, these techniques could be used to create a decent quality for a small project, that otherwise couldn't afford to use real voice actors. However, the quality for the generated voices is not yet there to compete with real voice acting on any level. We are interested to see where this approach can be taken to in the future, especially with games possibly using procedurally generated text in mind.

References

1 James Betker. *Better speech synthesis through scaling*. arXiv, 2305.07243, 2023
2 Z. Wang et al. *Multi-Level Temporal-Channel Speaker Retrieval for Zero-Shot Voice Conversion*. IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 32, pp. 2926-2937, 2024, doi: 10.1109/TASLP.2024.3407577.
3 Renpy. *https://www.renpy.org/*. Accessed 26th June 2024.

3.2 Meaningful Acoustics for Board Games

Filippo Carnovalini (VU - Brussels, BE), Greta Hoffmann (TH Köln, DE), Chengpeng Hu (Southern Univ. of Science and Technology - Shenzen, CN), Leonie Kallabis (TH Köln, DE), Matthias Müller-Brockhausen (Leiden University, NL), and Mike Preuß (Leiden University, NL)

Music has become a staple aspect in videogames, providing emotional depth and enhancing narration. Other sonic aspects also play a huge role: sound effects (*SFX*) and soundscapes better the user experience and can become elements of added realism and immersion.

Less attention has been devoted to *board games*, which because of their physical nature have less opportunities for embedding SFX and other procedural sounds. In this workshop we examined what the possible approaches to the sonification of board games would be, finding a general description of possible roles of music and sound in the gaming experience.

We started by thinking of existing examples of ways in which commercially available games incorporate music and sound. Some children's toys have various way to produce sounds that keep the children entertained, and there are some electronic board games that leverage sound as a feedback mechanism (e.g. Operation, or some iterations of Battleship) either through buzzers or through recorded sounds played via speakers. Escape rooms often have background music, and many game masters will (try to) provide music to accompany table-top RPGs. Other board games are now starting to provide music in external resources, for example via companion apps or through websites linked via QR codes. Some academic works have focused on procedural music generation, but mostly for videogames [1, 2]. Interestingly, the only work we know of that considers board games (although in digital format) is focused on computer-generated games [3].

We then moved to thinking about what it means to provide fitting/meaningful music to a board game. Sound can be used as a tool to provide feedback to enhance UX, it can enhance the narrative (by adding a soundtrack), or can even become a game element (consider for example Nintendo's *Ocarina Of Time*), although this seems harder to achieve in board games. We did not consider physical limitations (which would certainly need proper engineering to be overcome) such as ways to detect board game actions or to create sound, but rather considered the problem from an abstract perspective, trying to design a unified model that describes how different sounds can be used in games.

3.2.1 A Model of (Board) Game Sonification

We can formalize a game as a series of possible states in a discrete space, and the selection of a path in a tree, from root (start of the game) to leaf (end state) as the unfolding of the game (see Figure 3, left). Mathematically, one way to obtain fitting music for a game, would be minimizing the difference between features computed from the game state and features that drive the music generation (see Figure 3, right).

A more pragmatic approach leverages the model conceived by Greta Hoffmann prior to the seminar, which maps different sonic element to parts of the game graph. SFX are related to nodes, while the (procedurally generated) melody can follow the path of the game. Finally background music, or ambience, can be related to the more universal aspects of the game, such as the setting (see Figure 4).

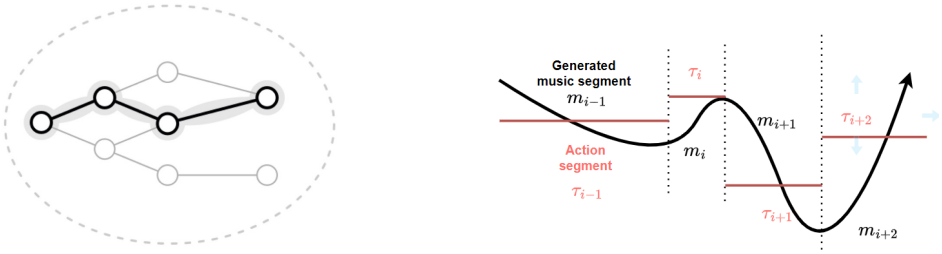


Figure 3 Left: A graph model of a game: each node is a game state, the root being the initial one. Moving from one state to the next within the universe of states creates a path, which is the experienced game.
Right: The (continuous) features of music generation can be influenced by the discrete game states. Inspired by the opposite approach in [4] where levels are generated depending on music.

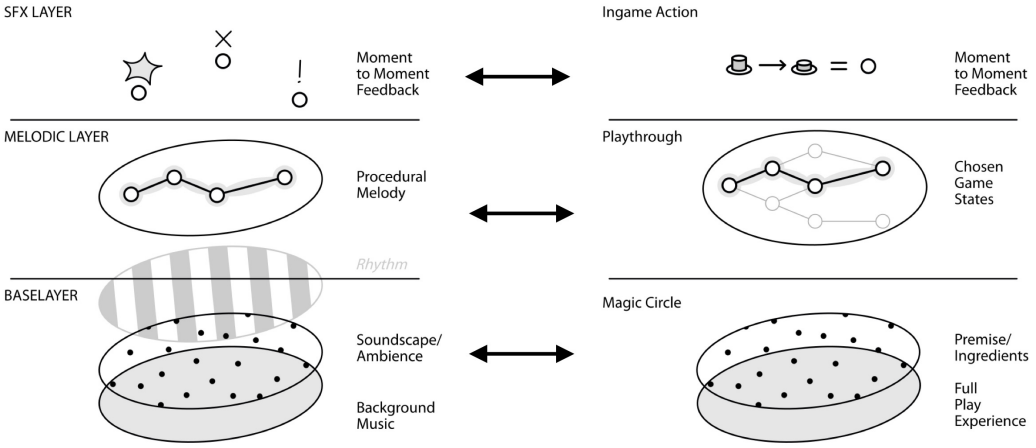


Figure 4 Mapping of different parts of our game model to musical elements.

This simple and generic model allows the sound to interact with the game and players in multiple ways. In the simplest case, it can be *Reactive*, being fully based on the game actions and only responding to those, but it could also be *Proactive*, by influencing the development of the game. It could for example help the audience of a game understand what moments in the game are more crucial, or it could influence players to pay attention to certain aspects of the game. In that sense, music could also be *Guiding*, by providing the players with directions through musical enhancement of the game. To better understand these different modalities, it is useful to consider how the model could be applied to some well-known games.

3.2.1.1 Colonists of Catan

Reactive: the background layer (ambience) could react to the last used/touched tile, with nature sounds relating to mountains, fields, or cattle. The procedural layer could instead be responding to the playstyle of each player.

3.2.1.2 Chess

Reactive: The procedural music could include leitmotifs for every piece, and the style could change depending on different openings used in the game. The SFX layer could respond to pieces movements, but also to special events such as checks.

3.2.1.3 Poker

Proactive: This game being based on incomplete knowledge could offer venues to use music in more advanced ways, having the players be able to influence the music to bluff using motifs related to cards they do not possess.

3.2.1.4 Hanabi

Proactive/Guiding: this game also has incomplete knowledge but it is collaborative: the music could collaborate with the players providing hints on the current game state.

3.2.1.5 Tic-Tac-Toe

We decided to apply the model in a demo recreating a digital version of the game Tic-Tac-Toe. Implemented in *pygame*, it uses the *mido* library to generate MIDI events that depend on the game state. A ticking sound constitutes the background layer, with the tempo increasing as the game progresses. The procedural level depends on the the board state, adding notes different notes depending on where Xs and Os are (using two different instruments for the two players/symbols), and adding an ominous low note when one of the players is about to create a line of three. The SFX layer has sounds for each X and O added to the board, as well as a endgame sound. The code is available at https://github.com/Facoch/Music-TicTacToe-Pygame.

3.2.2 Conclusions

While the working group managed to have fruitful discussion and produce a demo, many aspects still deserve investigation. Besides the practical engineering aspects needed to implement the proposed ideas in the physical world of board games, other aspects relating to the general model need further exploration. Most of our example are complete information game. When it is not the case, it would be interesting, or even necessary, to provide different acoustic experiences to different players, but that seems hard in the non-digital domain. Also, music would certainly have an impact on the game psychology, in ways that would deserve further studies.

Our main contribution is suggesting ways to map game elements to auditory events and procedural music. While we focused on board games, the model still largely applies to videogames as well, and could provide abstract guidance to sound designers and composers for the gaming industry.

References

1 P. Lopes, A. Liapis, and G.N. Yannakakis, *Sonancia: Sonification of procedurally generated game levels*, ICCC, 2015.
2 D. Plans and D. Morelli, *Experience-Driven Procedural Music Generation for Games*, in IEEE Trans. on Computational Intelligence and AI in Games, 2012.
3 S. Cardinale; M. Cook and S. Colton, *AI-Driven Sonification of Automatically Designed Games*. The Experimental AI and Games Workshop at AIIDE, 2022.

4 Z. Wang and J. Liu, *Online Game Level Generation from Music*, 2022 IEEE Conference on Games (CoG), Beijing, China, 2022.

3.3 Roguelike in a Day

M Charity (New York University, US), Alex J. Champandard (creative.ai - Wien, AT), David Melhart (University of Malta - Msida, MT), and Matthias Müller-Brockhausen (Leiden University, NL)

This Dagstuhl report details the 8-hour game jam project of the game "The Dragons of Castle Dagstuhl"[4]. Keeping with the theme of the Dagstuhl 24261 conference of "Computational Creativity for Game Development", we used large-language models and image generation tools to aid with the coding and asset design of the game. This small roguelike game was done by the small working group of M Charity, Matthias Muller-Brockenhausen, David Melhart, and Alex Champandard.

3.3.1 Development

The roguelike genre – defined from the 1980 game Rogue – focuses on elements of gameplay that are modeled off of dungeon exploration games such as Dungeons and Dragons. These gameplay concepts typically include, but are not limited to, procedurally generated levels and content, perma-death (where the player does not save any progress on death), and grid-world turn-based movement. We incorporated these elements in our game "The Dragons of Castle Dagstuhl."

The game was developed for HTML5-based browsers and uploaded to the independent game-hosting platform Itch.io. Original prototyping of the game involved a chess board found in the game room of Schloss Dagstuhl. The movement of the player and enemies as well as the 8x8 grid level design took inspiration from classic chess piece movements on the grid. Figure 5 shows the analog prototype version of the game on the chessboard with the wall and character pieces. The themes for the enemies and collectable items were also inspired by the themes surrounding castles (e.g. dragons, ghosts, skeletons), classic German fairy tale characters (e.g. wolves, goblins) and themes from the Dagstuhl conference itself (e.g. scientists, beer, cake, coffee.)

3.3.2 Gameplay

A player character moves around an 8x8 grid room environment in a turn-based fashion. To continue to the next room, they must reach the stairs without losing all of their health. They are given 4 health points at the start of the game. Being touched by an enemy character will cause them to lose health, but will also kill the enemy. Inspired by the pawn chess piece, the player can move 1-2 spaces at a time in the cardinal directions north, south, east, and west. The rooms are procedurally generated with preset wall shapes and sizes being randomly placed.

[4] https://mastermilkx.itch.io/the-dragons-of-castle-dagstuhl

Figure 5 Analog prototype for the game using Schloss Dagstuhl's chessboard

Two enemies known as the dragons, are placed in opposite but randomly selected corners of the room. Inspired by the king chess piece, the dragons can only move one space at a time, but will always move towards the player's position. Other enemies are randomly selected and placed into the room based on the room level rank – deeper level means more enemies will appear. This acts as a difficulty scale for added challenge in the game. Each enemy class is generated with random movement patterns at the start of the game – inspired by other chess pieces including the rook, knight, queen, and bishop. These enemies can either randomly choose a possible position or always move towards the player, choosing the closest possible position measured by a Manhattan distance. Because the patterns are always random at the start of the game session, the player must learn the movement patterns over time in a true roguelike fashion. The possible movement positions of the enemies are indicated with red squares on the map to allow the player to develop a strategy.

Players can pick up items throughout the room to increase their score value. GUI and flavor text were also implemented to the game for added thematic immersion. The player can switch between the minimalist mode – which shows the enemies, items, walls, and player character as colored squares – and the graphic mode – which uses the pre-made AI generated images as the sprites instead. Figures 6 and 7 shows the same room in the game, the first as the minimalist version and the second as the graphic version.

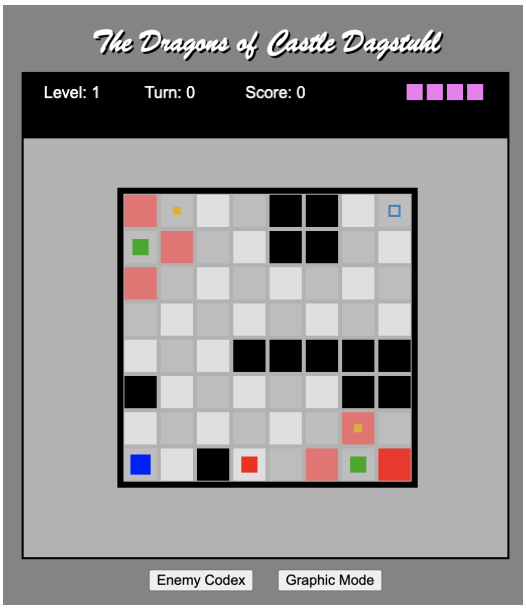


Figure 6 Minimalist mode of the game

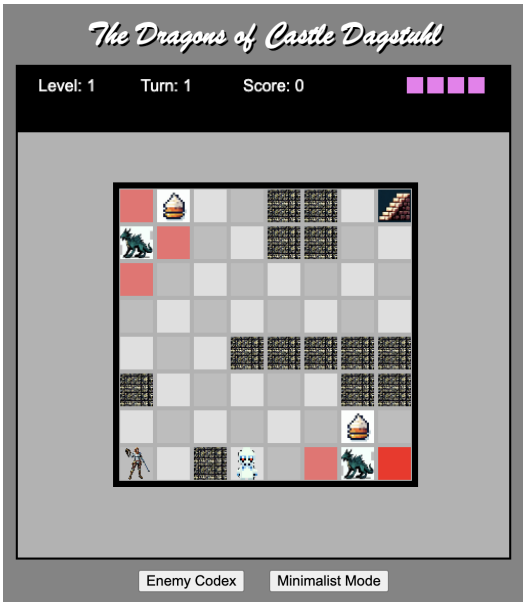


Figure 7 Graphic mode of the game

3.3.3 Generative AI Assistance

Generative AI was used at nearly every development stage of the game. We used Github Copilot[5] – integrated as an extension in Microsoft Visual Studio Code – for programming assistance. The sprites and graphics of the game (outside of the default minimalist mode)

[5] https://github.com/features/copilot

were created using Microsoft Bing's Image Creator tool[6]. While we weren't able to fully implement it into the final build of the game, the enemy descriptions and movement patterns were going to be procedurally generated during runtime using a built in Llama TTF script[7]. Incorporating these generative AI methods, greatly sped up the development time of the game and we were able to create a polished final product within 8 hours.

References

1 Zapata. *On the historical origin of the "roguelike" term*, 2017. Retrieved from https://blog.slashie.net/on-the-historical-origin-of-the-roguelike-term/.

3.4 AI for Romantic Comedies II

Michael Cook (King's College London, GB), Gabriella A. B. Barros (modl.ai - Maceio, BR), Alena Denisova (University of York, GB), Ahmed Khalifa (University of Malta - Msida, MT), Antonios Liapis (University of Malta - Msida, MT), Johanna Pirker (LMU München, DE), Emily Short (Oxford, GB), Gillian Smith (Worcester Polytechnic Institute, US), Anne Sullivan (Georgia Institute of Technology - Atlanta, US), and Tommy Thompson (AI and Games - London, GB)

At Dagstuhl Seminar 22251, the first author ran a workgroup about *AI for Romantic Comedy* [1]. In this working group, we discussed the difficulties inherent in modelling both romance and comedy through game and AI systems. After a length discussion in the morning, the afternoon sessions resulted in three short design proposals for projects that would examine different aspects that had been brought up. This included using the player to interfere with social simulations, and a proposal for a game which leveraged Twitch audiences to act as the audience for a reality TV show comprised of AI agents. Earlier in the week of Dagstuhl Seminar 24261 we worked on several groups relating to narrative analysis and simulation, which reminded some attendees of the working group on romance and comedy. We decided to run the topic a second time, to incorporate the views of new attendees, the intervening two years of research ideas, and to emphasise some practical experimentation.

Although many games support both romance and comedy, they tend to manifest in very different ways. Romance often appears either through a narrative chain of romantic subplots, especially 'romanceable' NPCs, or specific romantic systems that bond characters together over time. Comedy is more emergent – while explicitly comedic games exist (such as many of the LucasArts adventure games from the 1990s) comedy in games is often found as a consequence of player interaction with systems such as physics engines. Both comedy and romance are highly dependent on subtle concepts such as timing, pacing and social cues – all of which makes them difficult to either simulate or analyse automatically.

We continued the work of the 2022 working group in this session, opening with introductions and discussion, and then breaking early into three working groups that tackled different topics that had arisen: one practical implementation-focused project; one speculative design

[6] https://www.bing.com/images/create
[7] https://fuglede.github.io/llama.ttf/

project; and one survey of the landscape of romance in games. We briefly summarise our work below.

3.4.1 A Survey of Romance in Games

This subgroup, run by *(G. Barros, A. Khalifa*, and *A. Sullivan)*, surveyed the state of romance in games and began a categorisation of how they are integrated into the game's design. The inclusion criteria were that the game must incorporate at least one 'relationship', that romance must be an option, and that the relationship must change over time. The group developed two axes along which to sort games: the degree to which player action affects the development of the relationship, and the integration of the relationships into the game's mechanics and systems.

The subgroup also broke down romance systems into different types: action-based systems where game performance affects relationships; gift-based systems where items and dialogue are used instead; relationships that confer bonuses on the player's gameplay experience; and relationships that only affect the narrative or plot.

3.4.2 Story Sifting for Romance and Comedy

This subgroup was run by *(G. Smith, J. Pirker*, and *A. Liapis)*. Story sifting is a concept in narrative research whereby a simulation produces a large quantity of plot events, which an AI system then selects a subset of to present a compelling narrative or perspective on. This subgroup took this concept and applied it to modern interaction styles popularised by social media apps such as TikTok, to investigate how stories about human relationships could be told through fragmented or epistolary formats, with a human player potentially acting as the story sifter.

The group also investigated the degree to which large models such as ChatGPT or Midjourney could understand comedic or romantic narrative concepts, or how well they were suited for supporting content creation for such games. They found largely negative results in their short exploratory study, particularly in issues relating to heteronormativity, coherence and sustained content reuse.

3.4.3 Comedic Emergence in Social Simulations

This subgroup, run by *(E. Short, A. Denisova, T. Thompson*, and *M. Cook)*, investigated the requirements of a social simulation system to allow comedy (and romance) to emerge naturally through the structural setup. The group rapidly prototyped a scenario in Inform 7, an interactive fiction authoring engine, where a group are on a double date at a restaurant. By assigning a variety of actions, traits and inciting incidents to the cast and setting, the date inevitably goes wrong in different ways, arising to different kinds of outcome.

The group's findings suggested that understanding the affordances of the narrative space – as explored in another working group run by Emily Short at this seminar – might help to predict which combinations of setup properties result in more or less interesting, funny or romantic outcomes. Rather than trying to guide the narrative precisely, an AI system could instead simply help sculpt the space of opportunities. However, assessing what outcomes are 'funny' or 'romantic' remains a challenging, subjective and potentially unsolvable (in the traditional sense) problem.

References

1 Dan Ashlock, Setareh Maghsudi, Diego Perez Liebana, Pieter Spronck, Manuel Eberhardinger. *Human-Game AI Interaction (Dagstuhl Seminar 22251)*. 2022

3.5 AI for Speedrunning

Michael Cook (King's College London, GB), Maren Awiszus (Viscom AG - Hannover, DE), Filippo Carnovalini (VU - Brussels, BE), M Charity (New York University, US), and Alexander Dockhorn (Leibniz Universität Hannover, DE)

Speedrunning refers to a collection of related activities where people play games under specific conditions – usually trying to complete a game as quickly as possible, but sometimes trying to complete it with certain restrictions (e.g. while blindfolded), variations (e.g. randomisers which alter the structure of an otherwise static game), or other feats (e.g. two players sharing a single controller). Speedrunning is a very popular subculture within games: the official portal *speedrun.com* reports 20m annual visits to their site, which hosts over 4.7m individual speedruns across 43.2k games [1].

Speedrunning remains vastly understudied within game AI research, despite the obvious parallels between game-playing AI research and time-optimised game-playing. Interestingly, one of the few pieces of academic writing about speedrunning in games comes from a Dagstuhl publication [2], with some studies existing from a sociological or cultural perspective outside of AI [3]. This working group set out to discuss the many problems that exist within speedrunning for AI researchers to tackle, and then to concretely implement a prototype platform for speedrunning research with AI systems.

We began by discussing the current state of speedrunning and identifying where there was potential for impact. The speedrunning community is inventive and resourceful, and already do a lot of work that would be considered research-grade in some fields: randomisers, for example, procedurally modify games to make them unpredictable to play, while retaining consistency in terms of pacing, flow and complexity. We also discussed *tool-assisted speedruns* (TAS), where speedruns are executed by a computer replaying pre-defined commands (not competing with human speedrunners). This allows speedrunners to perform tricks requiring superhuman skill, but each TAS must be made by hand.

Speedruns, no matter what form they take, usually exploit *glitches* in games to skip content or progress faster. These glitches take on many forms, including manipulating data in memory, forcing physics simulations into edge case scenarios, and causing simultaneous execution of code through multiple inputs. Many of the most popular AI environments for game-playing in the past decade are competitive, meaning they are ranked on winrate rather than time taken. For single-player games used as AI environments, such as DOOM, the reward signal for the discovery and use of glitches is likely too weak for most AI to use. For this reason, we chose to use the workgroup to build testing environments for single-player, open-source and speedrunnable games, so that we can investigate this problem space further in the future. In the next section we describe our chosen platform, the game engine *PICO-8* and the game *Celeste*.

Figure 8 The sprite design interface.

3.5.1 Celeste and PICO-8

PICO-8 is a *fantasy console* – a type of game engine specifically designed to be highly constrained, often mimicking the hardware restrictions in consoles from the 1990s and 1980s. PICO-8 is perhaps the most popular example of this. Its restrictions include a 128x128 pixel screen, a palette of 16 colours, a maximum size of 32kb for games, and a limit of 256 game sprites (see Figure 8). All PICO-8 games are open-source, and are distributed online through a BBS-like system within PICO-8 itself.

Celeste, by EXOK Games, is a 'hardcore platformer' released in 2017. The primary mechanic in the game is dashing – the player can press a button to dash once in any direction, including in the air. This is reset when they are touching the ground again. Celeste became popular with speedrunners due to its difficulty level and the high skill ceiling of its controls. Celeste was expanded into a full game and released in 2018, where it won numerous awards and sold millions of copies. The full game is also beloved by speedrunners, and has many specific dedicated speedrun mods and extensions made for it.

We chose PICO-8 as a target domain due to its openness and the ease with which games can be instrumented and analysed. Celeste was an obvious target for us because its prominent position in the speedrunning community and its many known glitches and exploits. However, during the working group we discovered *Celeste Tech Training*, a PICO-8 game made specifically to teach speedrunners how to perform certain tricks. We modified this game to strip out unnecessary functionality, and used its focused levels to test our prototypes on. Figure 9 shows the first level of this game, which teaches a technique called *spike jumping*. Spike jumping allows the player to jump on a specific part of a spike floor without being hurt.

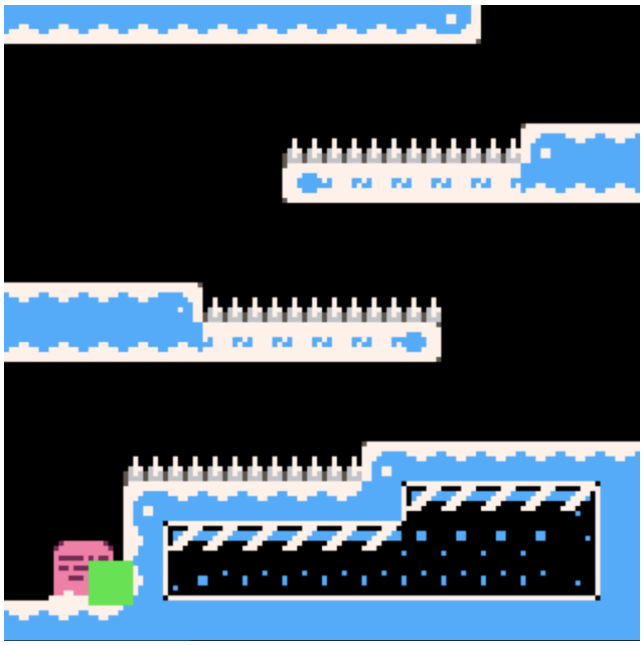


Figure 9 Modified version of CTT.

3.5.2 Approaches

We implemented three different systems for replicating the spike jump technique in *Celeste Tech Training* (CTT). Our first system was built into the code – it simulates virtual inputs and can load and save game states using custom code. This is the least flexible solution as it needs to be rewritten for different games, but it is the most portable – it is self-contained within the cart and does not require any external tools.

The second solution leverages Celia [4], a LUA software designed to facilitate the creation of TASs of PICO-8 games. By modifying the source code of the project, the software was adapted to automatically create TASs of a simplified Celeste level which requires a spike jump to be beaten (see Figure 9). We implemented a random agent that adds random inputs every fifth frame of TAS. The distance of the character from the goal position (beyond the spikes section that requires a spike jump to be cleared) served as an objective function. Whenever a new shortest distance was achieves, the TAS was saved, providing record of how it is possible to reach that distance. In our tests[8], the random agent was unable to reach the destination point, but it managed to perform the initial part of the spike jump: it jumped on the correct pixel at the corner of the spikes, but failed to then perform a dash at the correct moment to clear the needed distance. A Reinforcement Learning based agent could probably fare better than this naïve random agent, providing more adaptability over our first approach.

For a more general approach, we designed a Python interface to work with the Celia software. With the *pynput* library, this approach used keypresses and Celia command shortcuts (i.e. loading the TAS files, skipping frames) to play the PICO-8 games frame-by-frame. Evaluation for this approach would involve retrieving screenshots from the game through the Celia software. With the frame manipulation, the game could also be reset

[8] The modified Celia code is available at https://github.com/Facoch/Celia

to earlier states for tree-searches of optimal paths and keystrokes. With this approach, an AI-generated speedrun could be made for any PICO-8 game that could be loaded into Celia; without manipulating the source code of the game or Celia itself.

We tested this methodology on three different games retrieved from the PICO-8 community BBS[9]: Get Out of this Dungeon, treeboi_test, and Witch Loves Bullets. With all three games, randomly made TAS files were successfully generated, loaded, and played in the Celia software. Future work would look to using tree-search methods such as A* to create speedruns.

3.5.3 Further Work

Our next steps are to clean up and open source these systems, along with publishing our initial survey of the speedrunning landscape with respect to AI. Beyond that, we believe Celeste in PICO-8 represents a good domain for an AI competition. Designing the framework and rules for this competition will also help us clarify what challenges are most interesting, and begin to grow academic interest around this area.

References

1 Speedrun.com – About. http://www.speedrun.com/about Accessed 3 July 2024.
2 Manuel Lafond. *The complexity of speedrunning video games*. In 9th International Conference on Fun with Algorithms (FUN 2018). Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018.
3 Scully-Blaker, Rainforest. *Re-curating the Accident: Speedrunning as Community and Practice*. Masters thesis, Concordia University, 2016.
4 gonengazit/Celia. https://github.com/gonengazit/Celia Accessed 9 July 2024.

3.6 Skill-Discovery in (Strategy) Games

Alexander Dockhorn (Leibniz Universität Hannover, DE), Manuel Eberhardinger (Hochschule der Medien - Stuttgart, DE), Chengpeng Hu (Southern Univ. of Science and Technology - Shenzen, CN), and Matthias Müller-Brockhausen (Leiden University, NL)

Strategy games present a unique challenge in artificial intelligence (AI) research. They can broadly be classified into two types: turn-based and real-time strategy games. Both types typically require the player or AI to manage multiple units or resources, often with incomplete information about the opponent's actions. The large branching factor and long game duration make it difficult for AI to explore all possible strategies, which is further complicated by the need to plan several moves ahead. The state-of-the-art methods in AI for strategy games include search-based algorithms and reinforcement learning (RL), but these often rely on human-defined strategies or subgoals, limiting their scalability and generalizability.

While the work on AlphaStar [2, 3] and OpenAI Five [5] have shown that it is possible to train strong AI agents for complex games such as Starcraft 2 and Dota 2, both works required massive amounts of compute resources until satisfying results have been achieved. For the purpose of speeding up the learning process, the working group on skill discovery in

[9] https://www.lexaloffle.com/bbs/?cat=7

(strategy) games has been formed to evaluate the applicability of skill discovery methods to this special domain. We particularly emphasize works on skill discovery as part of RL algorithms. In this context, skill discovery refers to identifying and learning sub-policies or strategies that can be applied to achieve or identify specific subgoals within a game, thereby enabling more efficient and scalable AI systems.

3.6.1 Preliminaries of Skill Discovery

Skill discovery in AI remains an open problem, particularly when it comes to discovering skills without human intervention. The interpretation of what constitutes a "skill" in a given context is still unclear, making it challenging to develop a unified approach to skill discovery.

Particularly within the RL framework, a skill is typically defined as a policy aimed at achieving a specific subtask or goal. The options framework [7, 6] formalizes this by learning policies for subtasks, identifying the start and end points of these tasks, and using these learned skills to simplify the overall decision-making process. However, the distinction between a skill and a task is not always clear, especially when a subgoal can only be reached through a single deterministic policy.

3.6.2 Proposed Approaches and Ideas

Current research explores various methods for skill discovery, including hierarchical approaches, bottom-up skill learning, and the application of relational representations of game elements. Given an initial literature review, our working group has identified the following promising approaches for skill discovery in strategy games:

- **Text-based Task Decomposition:** Strategy games often have a simple goal, e.g. defeating the opponent's units or destroying its base. However, doing so involves plenty of subtasks. Those can be defined on varying ranges of granularity. Given the increasing capabilities of large language models, task descriptions such as "defend the base" could be decomposed into "train at least 3 units" and "patrol the surroundings of your base". Such enriched descriptions may directly represent sub-goals and allow for a more interpretable and scalable approach to skill discovery. Further, it allows to define more fine-grained reward functions given the descriptions [12].
- **Relational Representations:** Game state representations in strategy games can become quite complex. Units, abilities, weapons, buildings, and resources are just a few of the typical systems included in strategy games. Attempts to create general vectorized state representations have recently been studied [11], however, those create a unique representation for every game mapping all of its subsystems. While they allow the definition of state-space abstractions, transferring results from one game to the other is hindered by the granularity of this state representation. Similarly, matrix-based or image-based representations as used in AlphaStar [2] enabled the training through large-scale reinforcement learning but due to the complexity of the input at the cost of enormous computational resources.

  One possibility to overcome this problem is the use of a relational representation of game elements, such as "workers - mine - resources." Defining low-level systems for the execution of such relations allows to focus the agent's training on high-level strategic decision-making. At the same time, the high-level relation allows the transfer of knowledge in between games with different low-level controls. Using such a representation in combination with relational reinforcement learning [10, 9, 8] may therefore improve the efficiency of training agents in complex strategy games.

- **Pattern Mining and Clustering:** Given a data set of successful and unsuccessful play traces, pattern mining and clustering algorithms may be used to cluster them into groups of similar elements and extract abstract prototypes. Techniques like the KRIMP algorithm[1] or Skid Raw [19], which extract patterns from previous action sequences, could be applied to discover meaningful skills. Similarly, time sub-series mining [4], used to measure the distance between interaction sequences, may reveal underlying patterns that represent skills.
- **Bottom-Up Skill Discovery:** While most existing methods rely on top-down approaches, our group was exploring approaches for reversing this process by discovering skills from the ground up. Current methods are able to learn skills in the latent space of neural networks from collected demonstrations [21, 22, 23, 24]. However, these skills are not interpretable, and only after their execution can one infer what the learned skills represent. Another disadvantage of these methods is that they are only suitable for small environments and toy tasks, where the agent needs to navigate to multiple goals. To overcome the limitations of simple tasks and apply these methods to more complicated domains, we propose to learn skills from sequences of actions and iteratively refine these skills to handle the large search spaces inherent in strategy games. [18].
- **Skill Discrimination:** Effective skill discovery requires a discriminator to identify whether a discovered policy qualifies as a skill and if it is any different than already known skills [13]. Quality Diversity Optimization as in the Diversity Policy Gradient algorithm [20] introduces a method for discovering a diverse set of skills by balancing the exploration of different strategies with maintaining high-quality solutions. Recently, Wang et al. [25] proposed to incorporate a regularization term into the RL objective that maximizes the negative correlation to increase the diversity of RL policies via assembling multiple sub-policies. Notably, this diversity pertains to the behavior of the derived policy rather than the parameter space, as minor variations in parameters could lead to significant differences in behavior. Although this algorithm was initially verified in the context of game content generation, it could be adapted for skill discrimination to maximize the diversity of discovered skills.

3.6.3 Conclusion

Skill discovery in strategy games is a critical area of research that has the potential to significantly enhance the capabilities of AI systems and speed up their training. By exploring new methods for discovering and learning skills, our working group reviewed recent works on skill discovery in other domains than game-playing and identified interesting areas for further research. From here on, we outline several future actions to advance skill discovery in strategy games:

- We plan to investigate hybrid/iterative approaches that combine a bottom-up and top-down search for skills. Such hybrid approaches may offer a more robust solution to the challenges of skill discovery in large and complex game environments.
- Leveraging existing game platforms, such as GVGAI [17], Stratega [14, 15], and Griddly [16], could facilitate the testing and validation of new skill discovery methods. These platforms provide standardized environments for benchmarking AI performance, which is crucial for comparing the effectiveness of different approaches.
- Given the limitations of current methods, particularly in terms of scalability, we propose to produce a comprehensive survey paper.

References

1 Vreeken, J., Leeuwen, M. & Siebes, A. Krimp: mining itemsets that compress. *Data Mining And Knowledge Discovery*. **23**, 169-214, 2010 (10). http://dx.doi.org/10.1007/s10618-010-0202-x.

2 Mathieu, M., Ozair, S., Srinivasan, S., Gulcehre, C., Zhang, S., Jiang, R., Paine, T., Powell, R., Żołna, K., Schrittwieser, J., Choi, D., Georgiev, P., Toyama, D., Huang, A., Ring, R., Babuschkin, I., Ewalds, T., Bordbar, M., Henderson, S., Colmenarejo, S., Oord, A., Czarnecki, W., Freitas, N. & Vinyals, O. *AlphaStar Unplugged: Large-Scale Offline Reinforcement Learning*, 2023. https://arxiv.org/abs/2308.03526.

3 Arulkumaran, K., Cully, A. & Togelius, J. Alphastar: An evolutionary computation perspective. *Proceedings Of The Genetic And Evolutionary Computation Conference Companion*. pp. 314-315, 2019.

4 Mörchen, F. *Time series knowlegde mining*, Görich und Weiershäuser, 2006.

5 OpenAI, :, Berner, C., Brockman, G., Chan, B., Cheung, V., Dębiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., Józefowicz, R., Gray, S., Olsson, C., Pachocki, J., Petrov, M., O. Pinto, H., Raiman, J., Salimans, T., Schlatter, J., Schneider, J., Sidor, S., Sutskever, I., Tang, J., Wolski, F. & Zhang, S. *Dota 2 with Large Scale Deep Reinforcement Learning*, 2019. https://arxiv.org/abs/1912.06680.

6 Al-Emran, M. Hierarchical reinforcement learning: a survey. *International Journal Of Computing And Digital Systems*. **4**, 2015.

7 Stolle, M. & Precup, D. Learning options in reinforcement learning. *Abstraction, Reformulation, And Approximation: 5th International Symposium, SARA 2002 Kananaskis, Alberta, Canada August 2–4, 2002 Proceedings 5*. pp. 212-223, 2002.

8 Zambaldi, V., Raposo, D., Santoro, A., Bapst, V., Li, Y., Babuschkin, I., Tuyls, K., Reichert, D., Lillicrap, T., Lockhart, E. & Others. *Relational deep reinforcement learning*, 2018. *ArXiv Preprint ArXiv:1806.01830*.

9 Morales, E., Scaling up reinforcement learning with a relational representation. *Proc. Of The Workshop On Adaptability In Multi-agent Systems*. pp. 15-26, 2003.

10 Džeroski, S., De Raedt, L. & Blockeel, H., Relational reinforcement learning. *Inductive Logic Programming: 8th International Conference, ILP-98 Madison, Wisconsin, USA, July 22–24, 1998 Proceedings 8*. pp. 11-22, 1998.

11 Dockhorn, A., Hurtado-Grueso, J., Jeurissen, D., Xu, L. & Perez-Liebana, D., Game state and action abstracting monte carlo tree search for general strategy game-playing. *2021 IEEE Conference On Games (CoG)*. pp. 1-8, 2021.

12 Luketina, J., Nardelli, N., Farquhar, G., Foerster, J., Andreas, J., Grefenstette, E., Whiteson, S. & Rocktäschel, T., A survey of reinforcement learning informed by natural language. *ArXiv Preprint ArXiv:1906.03926*, 2019.

13 Eysenbach, B., Gupta, A., Ibarz, J. & Levine, S., Diversity is all you need: Learning skills without a reward function. *ArXiv Preprint ArXiv:1802.06070*, 2018.

14 Dockhorn, A., Grueso, J., Jeurissen, D. & Liebana, D. STRATEGA: A General Strategy Games Framework. *AIIDE Workshops*, 2020.

15 Perez-Liebana, D., Dockhorn, A., Grueso, J. & Jeurissen, D. The design of “Stratega": A general strategy games framework. *ArXiv Preprint ArXiv:2009.05643*, 2020.

16 Bamford, C. Griddly: A platform for AI research in games. *Software Impacts*. **8**, 2021.

17 Perez-Liebana, D., Liu, J., Khalifa, A., Gaina, R., Togelius, J. & Lucas, S., General video game ai: A multitrack framework for evaluating agents, games, and content generation algorithms. *IEEE Transactions On Games*. **11**, 195-214, 2019.

18 Liu, G., Hu, E., Cheng, P., Hung-Lee & Sun, S., Hierarchical Programmatic Reinforcement Learning via Learning to Compose Programs, 2023. https://arxiv.org/abs/2301.12950.

19 Tanneberg, D., Ploeger, K., Rueckert, E. & Peters, J., Skid raw: Skill discovery from raw trajectories. *IEEE Robotics And Automation Letters*. **6**, 4696-4703, 2021.
20 Pierrot, T., Macé, V., Chalumeau, F., Flajolet, A., Cideron, G., Beguir, K., Cully, A., Sigaud, O. & Perrin-Gilbert, N., Diversity policy gradient for sample efficient quality-diversity optimization. *Proceedings Of The Genetic And Evolutionary Computation Conference*, 2022 (7). http://dx.doi.org/10.1145/3512290.3528845.
21 Nieto, J. J., Castanyer, R. C., & Giro-i-Nieto, X., *Unsupervised Skill-Discovery and Skill-Learning in Minecraft*. In ICML 2021 Workshop on Unsupervised Reinforcement Learning, 2021.
22 Kim, T., Ahn, S., & Bengio, Y., Variational temporal abstraction. Advances in Neural Information Processing Systems, 32, 2019.
23 Kipf, T., Li, Y., Dai, H., Zambaldi, V., Sanchez-Gonzalez, A., Grefenstette, E., ... & Battaglia, P., *Compile: Compositional imitation learning and execution*. In International Conference on Machine Learning, pp. 3418-3428, 2019.
24 Jiang, Y., Liu, E., Eysenbach, B., Kolter, J. Z., & Finn, C., *Learning options via compression*. Advances in Neural Information Processing Systems, 35, 21184-21199, 2022.
25 Wang, Z., Hu, C., Liu, J., & Yao, X., *Negatively correlated ensemble reinforcement learning for online diverse game level generation*. In The Twelfth International Conference on Learning Representations, 2024.

3.7 Introducing AI Experience: Games UX in the Age of Generative AI

Anders Drachen (University of Southern Denmark - Odense, DK), Paolo Burelli (IT University of Copenhagen, DK), Leonie Kallabis (TH Köln, DE), and David Melhart (University of Malta - Msida, MT)

3.7.1 Introduction

The work group considered the evolving role of User Experience (UX) research in the context of digital games as they transition towards using generative AI. Traditionally, game design has been a manual process where designers meticulously craft environments, narratives, and interactions to shape a predictable and controllable user experience. However, with the integration of procedural content generation and generative AI models, such as Large Language Models (LLMs), the landscape of game development is potentially shifting towards an era where games can be dynamically created and adapted, not just during production but also in real-time as players engage with them.

This shift presents new challenges and opportunities for UX research. The essay outlines how existing UX research methods, which rely on controlled testing environments and predictable user interactions, are increasingly inadequate for understanding and evaluating experiences in games that are generated on-the-fly by AI. Traditional UX frameworks are built on the premise that game environments and player interactions can be pre-defined and tested empirically. However, in a generative game context, where AI can autonomously create complex, responsive environments and narratives tailored to individual players, the very foundations of UX research are called into question.

3.7.2 How Generative Games Impact UX Research

Several key dimensions of generative games that impact UX research: conversion, complexity, timing, staticness, social complexity, and personalization. These dimensions describe the extent to which game elements are generated, their complexity, when generation occurs (pre-production, at game start, or in real-time), how static or dynamic the generated content is, the number of players involved, and the level of personalization to individual players. Each of these factors adds layers of variability that challenge traditional UX evaluation methods, making it harder to predict and measure user experience outcomes.

If we consider a future where AI could create entire gaming experiences from scratch, adapting continuously to user behavior and preferences, what role is left for human designers? In such a scenario, the role of human designers and traditional UX researchers could diminish, replaced by AI systems that not only generate games but also simulate user responses to test and refine these experiences. This raises profound questions about the future of UX research. Will traditional concepts like sample sizes and controlled environments become obsolete? Will UX researchers need to transform into AI Experience (AIX) engineers who design the parameters and constraints that guide AI-generated experiences?

Despite these challenges, even in a future scenario where generative AI is capable of designing and developing the kinds of games that are today hand-crafted, it is suggested that there remains a vital role for human creativity and oversight in game design. Human designers bring an irreplaceable understanding of narrative, emotion, and player psychology that AI, despite its capabilities, might never be able to fully replicate. Moreover, the ongoing need to ensure ethical considerations, inclusivity, and meaningful engagement in gaming experiences underscores the importance of human involvement.

3.7.3 Conclusion

In conclusion, as generative AI continues to advance, UX research in gaming must evolve to address the complexities and dynamic nature of AI-generated content. Researchers and designers should collaborate to develop new methodologies and frameworks that can adapt to this rapidly changing landscape, ensuring that player experiences remain engaging, meaningful, and ethically sound. This synthesis highlights the need for a paradigm shift in UX research, moving towards a future where human and AI collaboration creates richer, more personalized gaming experiences.

References

1 Kokkinakis, A., Demediuk, S. P., Nölle, I., Olarewaju, O., Patra, S., Robertson, J., York, P., Pedrassoli Chitayat, A., Coates, A., Slawson, D., Hughes, P., Hardie, N., Kirman, B., Hook, J. D., Drachen, A., Ursu, M. & Block, F. O., *DAX: Data-Driven Audience Experiences in Esports*. In Proceedings for ACM International Conference on Interactive Media Experience, IMX 2020 (Barcelona, Spain), Association for Computing Machinery (ACM), 2022.

2 Drachen, A., Mirza-Babaei, P. & Nacke, L. (Eds), *Games User Research*. Oxford University Press. ISBN-10: 0198794843, 2018.

3 Cairns, P., *Doing Better Statistics in Human-Computer Interaction*. Cambridge University Press, 2019.

4 Shaker, N., Togelius, J. and Nelson, M., *Procedural Content Generation in Games*, Springer, 2016.

5 Hopson, J., *The Secret Science of Games*, 2013.

6 M. Cook, S. Colton and J. Gow, *The ANGELINA Videogame Design System—Part I*, in IEEE Transactions on Computational Intelligence and AI in Games, vol. 9, no. 2, pp. 192-203, 2017. doi: 10.1109/TCIAIG.2016.2520256.

3.8 LLM-based Program Search for Games

Manuel Eberhardinger (Hochschule der Medien - Stuttgart, DE), Duygu Cakmak (Creative Assembly - Horsham, GB), Alexander Dockhorn (Leibniz Universität Hannover, DE), Raluca D. Gaina (Tabletop R&D - London, GB), James Goodman (Queen Mary University of London, GB), Amy K. Hoover (NJIT - Newark, US), Simon M. Lucas (Queen Mary University of London, GB), Setareh Maghsudi (Ruhr-Universität Bochum, DE), and Diego Perez Liebana (Queen Mary University of London, GB)

Before the advent of large language models (LLMs) for code [1], program synthesis was considered a difficult problem due to the combinatorial explosion of the search space [2], and so most solvable tasks were based on simple string manipulations or list sorting problems in a predefined domain-specific language (DSL) [3]. Program synthesis for games was also limited to simple problems with a well-defined search space, which was only feasible by incorporating high-level concepts of the game into the DSL [4, 5, 6].

Considerations on the use of program synthesis with higher programming languages such as Python or Java for games research were rarely made and only possibilities were outlined [7] or it was evaluated how to bring automated game design systems from game description languages to the use of programming languages [8].

Recently, methods for LLM-based program search for the automatic design of playable games based on program code [9, 10, 11] or game content based on JSON representations [12] have been presented. In addition, LLMs are also adapted for synthesizing programmatic policies in Python, which are then converted into a DSL usable in the given environment [13] or for building a world model based on python code, approximating the reward and state transition functions for simple games, which are then used for generating an action plan [14].

In this working group, we explore the possibilities of LLM-based program search for a broader range of applications for games without relying on a predefined specification such as a DSL, e.g. Ludii [9], the video game description language [10] and Karel [13], or a predefined converter for JSON [12]. The goal is that LLMs synthesize program code that is directly usable without further transformation or prior specification. We evaluate our approach on different domains in one of two programming languages, Python and Java. In Python, programmatic agent policies and functions for procedural content generation (PCG) are synthesized. In Java, the framework is included into TAG, a tabletop games framework, where heuristics for board games are designed [16].

3.8.1 Framework

The general framework is based on an evolutionary hill-climbing algorithm where the mutations and the seed of the initial program are performed by an LLM [13, 15]. Figure 10 displays the whole framework. We start by generating a task prompt to obtain an initial Python or Java function, which is then executed in a safe environment in a subprocess.

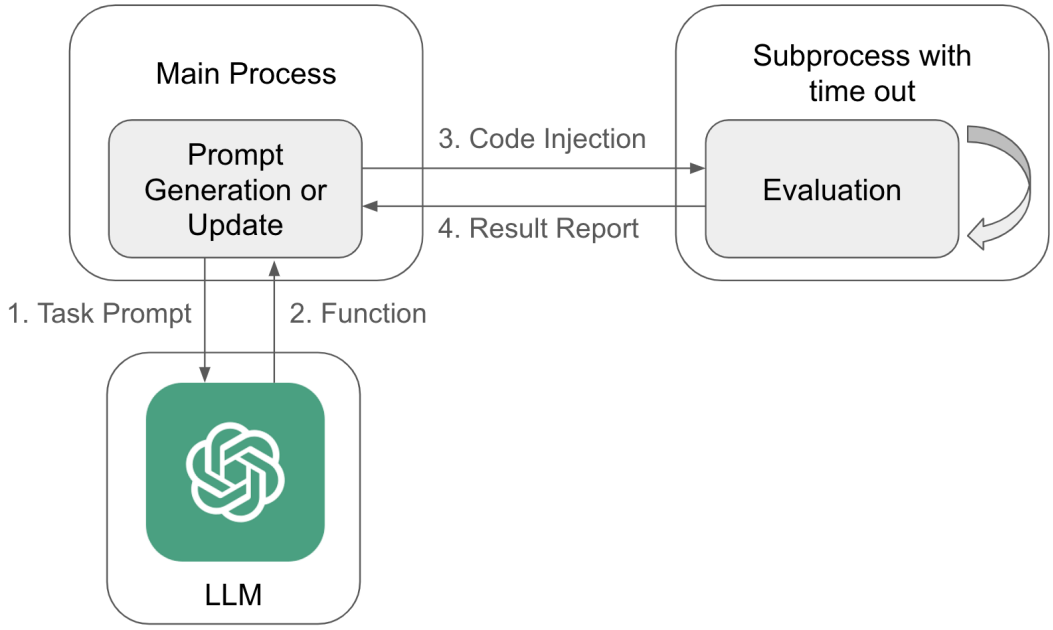


Figure 10 The general framework for the program search. At the beginning, an initial prompt is generated, which is then processed by the LLM and returns a function. Subsequently, the returned function is evaluated in a sub-process and the results are reported back to the main process, where the prompt is updated and then returned to the LLM or the evaluation criteria is met.

This ensures that the main process can terminate the function after a certain time period, preventing the synthesized code from running indefinitely. If the function has been executed successfully, the task prompt is updated with the evaluation metric achieved and some additional information, depending on the environment, e.g. the action trace of the executed function. If an error occurs, e.g. the code cannot be parsed due to incorrect syntax, runtime errors due to incorrect indexing of arrays or similar problems, the description of the error is used to update the task prompt. These steps are repeated iteratively until the evaluation criteria, the fitness function, for the problem domain is fulfilled or the specified number of generations is reached. The individual steps are summarized in Algorithm 1.

Algorithm 1 The algorithm for the framework.

$prompt \leftarrow get\_task\_prompt()$
$p \leftarrow query\_llm(prompt)$
$r \leftarrow inject\_and\_run\_code(p)$
$f \leftarrow evaluate\_fitness(r)$
while not $fulfilled\_criterion(f)$ **do**
  $prompt \leftarrow update\_task\_prompt(r, f)$
  $p \leftarrow query\_llm(prompt)$
  $r \leftarrow inject\_and\_run\_code(p)$
  $f \leftarrow evaluate\_fitness(r)$
end while
return p

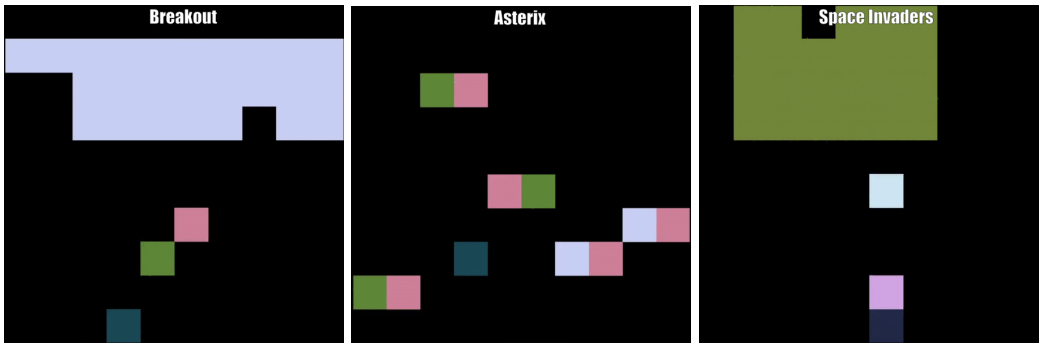


Figure 11 The three miniature versions of Atari games, Breakout, Asterix and Space Invaders, which are used for the synthesis of programmatic policies.

3.8.2 Game Applications

Since our framework is independent of the used LLM, we use Llama 3.1 [19] or ChatGPT based on GPT-4 [20] in the following experiments.

3.8.2.1 Programmatic Polices: Minatar

Minatar [17] is a collection of five games that are miniature versions of Atari games. In Minatar, the games are represented as a symbolic state space on a 10x10 grid, with each grid cell representing an object such as walls, enemies or the agent. Previously, Minatar was used in [18] to explain the behavior of agents through program synthesis, but it was only possible to explain short sub-trajectories since enumerative search based methods were used to search through a predefined domain-specific language that resembles Lisp. For all Minatar experiments Llama 3.1 [19] with a search budget of 200 iterations, i.e. 200 programs, with a total of five refinements for each program, is used. The agent only receives the first state of the game with a description of the available objects of the state, actions of the environment and a description of the game, that is taken from Young and Tian [17]. In these experiments we use three Minatar environments, Breakout, Asterix and Space Invaders, which are shown in Figure 11.

Breakout is a game where the goal is to destroy all the bricks with the ball by controlling the paddle to bounce the ball each time before it goes out off the screen. With each destroyed brick the agent receives a reward of one. Listing 1 shows the best synthesized program.

The average reward in the final evaluation was 9.06 out of 1000 test runs, which is comparable to the DQN network with a replay buffer trained on this problem [17]. The best run achieved a reward of 103, i.e. more than three runs of the game without defeat.

```
import numpy as np

def policy(state):
    state = np.array(state)
    ball_position = np.where(state == 'ball')
    paddle_position = np.where(state == 'paddle')

    if ball_position[1].size > 0 and paddle_position[1].size > 0:
        ball_x = ball_position[1][0]
        paddle_x = paddle_position[1][0]

        if ball_x < paddle_x:
            return 'left'
        elif ball_x > paddle_x:
```

```
            return 'right'
        else:
            return 'noop'
    else:
        # If the ball or paddle is not found,
        # just return 'noop' for now
        return 'noop'
```

Listing 1 Best program found to play Breakout, which returns an average reward of 9.06 of 1000 runs

Asterix is a game where the goal is to control the player across the screen and collect gold while avoiding enemies. The player gets one reward for each collected gold and the game is over when the player comes in contact with an enemy. The synthesized code is 60 lines long (omitted here for brevity). The best run of 1000 test runs achieved an reward of 47 while the average reward of all test runs was 5.01. This is also comparable to the worst DQN network without a replay buffer from the baselines [17].

Space Invaders is a game where the player controls a cannon and must shoot aliens while dodging bullets launched from the alien spaceship. Additionally, the player must prevent the aliens from reaching the bottom of the screen. Listing 2 shows the best program found during the search process, which has an average reward of 20.89 when evaluating 1000 test runs, which is better to the worst neural network architecture DQN without a replay buffer and comparable to the DQN [17]. The best achieved reward was 47, where the agent almost destroy two appearing alien ships in a single episode before the aliens reached the agent.

```
import numpy as np

def play_space_invaders(state):
    # Get the current position of the cannon
    cannon_position = np.where(state == 'cannon')[1][0]

    # Check if there are any aliens in the current row
    alien_row = np.where((state[0:8, :] != 'empty') & (state[0:8, :] != '
    alien_left') & (state[0:8, :] != 'alien_right'))

    if len(alien_row[0]) > 0:
        # Find the closest alien to the cannon
        closest_alien_position = np.min(np.abs(alien_row[1] -
    cannon_position))

        # Move towards the closest alien
        if closest_alien_position < 0:
            return 'left'
        elif closest_alien_position > 0:
            return 'right'
    else:
        # If there are no aliens in the current row, move to the center
    of the screen
        if cannon_position < 5:
            return 'right'
        elif cannon_position > 5:
            return 'left'

    # Check for enemy bullets above the cannon and try to avoid them
    enemy_bullets = np.where(state[:, cannon_position] == 'enemy_bullet')
    if len(enemy_bullets[0]) > 0:
        if cannon_position < 9:
            return 'right'
        else:
            return 'left'
```

```
    # If no aliens or enemy bullets are found, try to fire
    return 'fire'
```

Listing 2 Best program found to play Space Invaders, which returns an average reward of 20.89 of 1000 runs

3.8.2.2 Vehicle Driving

For this task ChatGPT [20] is used to generate a python function which should drive a vehicle from its current position to a target position as quickly as possible and remain there once it arrived. Listing 3 shows the synthesized code after a few iterations. Figure 12 shows the distance from the vehicle to the target for the corresponding time steps on the left and the 2D trajectory of the vehicle on the right. The generated code is only partially successful: while it aims for and gets close to the target, it fails to stop and instead loops around it.

```
def get_action(vehicle: Vehicle, target: Vec2d) -> int:
    # Get the current position of the vehicle
    current_position= vehicle.s()

    # Calculate the vector from the vehicle's current position to the target
    target_vector = target - current_position

    # Calculate the angle between the current heading of the vehicle and the target vector current_heading = vehicle.heading()
    angle_to_target = math.degrees(math.atan2 (target_vector.y, target_vector.x)) - math.degrees(current_heading)

    # Normalize the angle to be between -180 and 180 degrees
    if angle_to_target > 180:
        angle_to_target -= 360
    elif angle_to_target < -180:
        angle_to_target += 360

    # Decide action based on the angle to the target
    if angle_to_target > 10:
        return 1 # Turn right
    elif angle_to_target < -10:
        return -1 #Turn left
    else:
        return 0 # Maintain current heading
```

Listing 3 A python function that aims to drive a vehicle to the target in 2D space - see description in the text.

3.8.2.3 Baba is You

Baba is you is a complex puzzle game in which a 2D grid environment is manipulated by the player to reach a given goal. The environment consists of word blocks and corresponding entities that can be pushed. By placing word blocks next to each other, rules can be formed. These rules are active as long as the given word block sequence remains intact. This way, players can change how objects behave, which objects they control, or which conditions must be satisfied to win the level.

For our experiments, we used a Python version [23] of the Keke is You AI framework [24]. Similar to the other games, we prompted the LLM to provide a policy given a short description of the game and the initial state of the level. The function to be written should use the current state as input and provide movement direction.

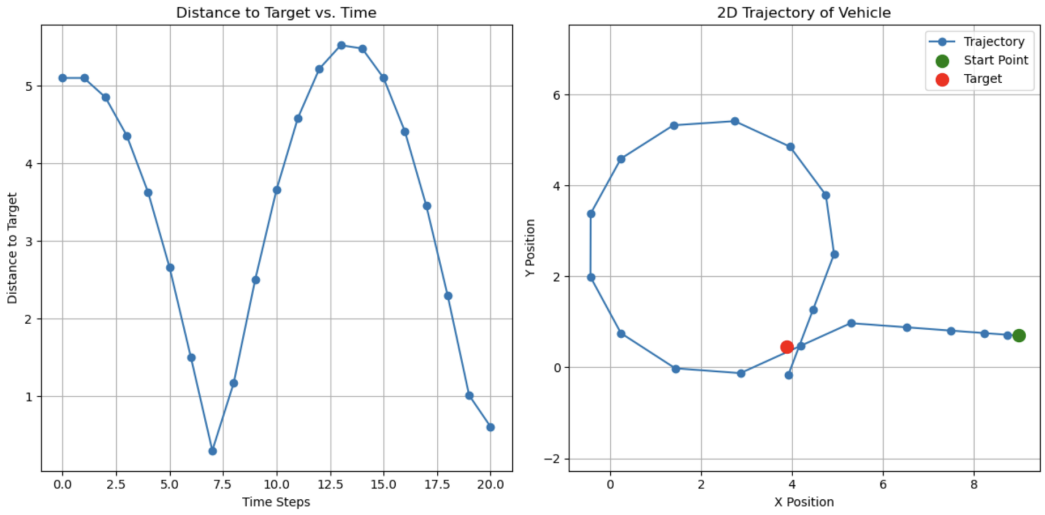


Figure 12 Left: The distance of the vehicle to the target for the corresponding time steps. Right: The 2D trajectory of the vehicle. It starts at the green point and aims for the red target. However, the generated code fails to stop the car close to the target, and instead will endlessly loop around it.

In our tests, the agent was able to solve simple test levels as the one shown in Figure 13. The policy returned by the optimization is shown in Listing 4. Complex object manipulation to change the rules while playing a level has not occurred. This may be overcome by future versions of the used LLM models or more complex prompting techniques.

```python
def program(state):
    state = state.tolist()
    for i in range(len(state)):
        for j in range(len(state[i])):
            if state[i][j] == 'b':
                x, y = i, j
            elif state[i][j] == 'f':
                fx, fy = i, j

    # Find the nearest word tile with the "Win" property
    for i in range(len(state)):
        for j in range(len(state[i])):
            if state[i][j] in ['B', 'b'] and \
                (i-1 >= 0 and state[i-1][j] == '3') or \
                (j+1 < len(state[0]) and state[i][j+1] == '3') or \
                (i+1 < len(state) and state[i+1][j] == '3') or \
                (j-1 >= 0 and state[i][j-1] == '3'):
                return "Right" if j > fy else "Left"

    # If no "Win" tile is found, move towards the flag
    dx = x - fx
    dy = y - fy
    if dx != 0:
        return "Up" if dx < 0 else "Down"
    elif dy != 0:
        return "Right" if dy > 0 else "Left"
    else:
        return "Wait"
```

Listing 4 A python function that solves the first level of the Keke AI PY framework.

Figure 13 First demo level of the Keke AI Py framework.

3.8.2.4 Tabletop Games Framework (TAG)

The TAG framework is a bespoke Java research framework that supports the implementation of multiplayer tabletop board games. The ultimate goal is to use the heuristic-generation algorithm outlined in Algorithm 1 on all games in the framework. This introduces a number of new challenges:

- The games are in general more complex than the simple one-player games in previous sections.
- Related to this, they are also inherently multiplayer. As such there is implicit opponent modeling required for good play strategies. The environment is no longer a 'simple' stationary MDP, but is actively adversarial.
- The TAG framework has a number of local libraries and coding conventions; for example decks of cards are implemented via `Deck<>` or `PartialObservableDeck<>` parameterised classes. These are not likely to be present in the LLM training data to any degree, and require the LLM to generalise to unseen software architecture details. This contrasts to the straightforward Python with mostly standard libraries of the game in earlier sections.

Two games were selected for initial experimentation. Tic-Tac-Toe is a simple 2-player game, and Love Letter is a slightly more complex 2-6 player game that requires reasoning over hidden information held by the other players.

The best Tic-Tac-Toe heuristic achieved a 65% win rate against a simple One Step Look Ahead (OSLA) agent, and consisted of 90 lines of code (omitted here for brevity). For Love Letter it was often difficult to get the LLM to generate valid code, let alone a heuristic that could win a game, although the best agents were able to beat random opponents.

Listing 5 illustrates the additional information needed in the prompt to obtain a working heuristic for Tic-Tac-Toe, including clear instructions not to leave TODO comments, exactly what dependencies need to be imported and details of the game-specific API that can be used to extract useful information.

Tic-Tac-Toe is an old, simple and popular game that is well-embedded in the training data. As such there was no need to explain how to play in the prompt. This was not true for Love Letter, and additional lines had to be added to explain how the game was played (and won or lost).

```
1  You are playing Tic Tac Toe.
2  Your job is to write the evaluation logic to help an AI play this game.
3  Don't leave parts unfinished or TODOs.
4
5  First, write a java class called TicTacToeEvaluator class, with only a
       single function with this signature:
6   - public double evaluateState(TicTacToeGameState gameState, int playerId
       )
7  This is a heuristic function to play Tic Tac Toe. The variable gameState
       is the current state of the game, and playerId
8  is the ID of the player we evaluate the state for. Write the contents of
       this function, so that we give a higher numeric
9  evaluation to those game states that are beneficial to the player with
       the received playerId as id. Return:
10   - 0.0 if player playerId lost the game.
11   - 1.0 if player playerId won the game.
12  If the game is not over, return a value between 0.0 and 1.0 so that the
       value is close to 0.0 if player playerId is close to losing,
13  and closer to 1.0 if playerId is about to win the game.
14 Take into account the whole board position, checking for lines that are
       about to be completed, and possible opponent moves.
15 You can use the following API:
16  - In TicTacToeGameState, you can use the following functions:
17     - GridBoard<Token> getGridBoard(), to access the board of the game.
18     - Token getPlayerToken(int playerId), to get the Token of the player
       passed as a parameter.
19     - boolean isGameOver(), returns true if the game is finished.
20     - int getWinner(), returns the Id of the player that won the game, or
        -1 if the game is not over.
21     - int getCurrentPlayer(), returns the Id of the player that moves
       next.
22  - GridBoard<Token> has the following functions you can also use:
23    - int getWidth(), to return the width of the board.
24    - int getHeight(), to return the height of the board.
25    - Token getElement(int x, int y), that returns the Token on the
       position of the board with row x and column y.
26  - Token represents a piece placed by a player. Which player the token
       belongs to is represented with a string. This string
27    is "x" for player ID 0, and "o" for player ID 1.
28    - Token(String) allows your to create token objects for any
       comparisons.
29    - String getTokenType() returns the string representation of the token
        type.
30 Assume all the other classes are implemented, and do not include a main
       function. Add all the import statements required,
31 in addition to importing games.tictactoe.TicTacToeGameState, core.
       components.GridBoard and core.components.Token
```

Listing 5 Prompt required to obtain valid code for a Tic-Tac-Toe heuristuic in TAG

The need to hand-craft these prompts for each game does not achieve the desired scalability across the suite of games within TAG. To resolve this at the tail end of the seminar, two new TAG-specific elements were implemented to augment the process:

1. Automatic extraction of the game-specific APIs. This uses Java Reflections to extract information on the methods and associated Javadoc on the game state object. This automatically generates the section of the prompt in Listing 5 from line 15 to the end;
2. Automatic rulebook digestion. This takes as input the PDF of the game rulebook. An approach inspired by [25] in a Minecraft-like environment is used. The rulebook is first broken down into chunks of 1000 or 2000 words to fit within the input of any LLM. Each chunk is then provided in turn to the LLM and two questions asked in separate prompts:
   a. Summarise in 200 words or less the information in this text about the game rules. Do

not include information on strategies to play the game.
b. Summarise in 200 words or less the information in this text about strategies and tips to play the game well. Do not include information on the game rules.

This generates two sets of data, one on the rules, and one on tips to play the game well (as these are often included in the rule book). Each of these sets is then fed to the LLM with a prompt to, 'Summarise this information in 500 words or less.'. This provides an additional two blocks of text to include in the prompt used in the main loop of Algorithm 1 that explain the rules of the game, and advice on how to play.

These new tools enable a more scalable and game-agnostic process to be run on all games that we plan to report results for in future work after the seminar.

3.8.2.5 Procedural Content Generation (PCG)

```
Start: (5, 5), End: (3, 4)
Maze 3:
1 1 1 1 1 1 1 1 1 1
1 0 0 0 0 0 0 0 0 0
1 0 1 1 1 1 1 0 1 0
1 0 1 0 0 0 1 0 1 0
1 0 1 0 1 1 1 0 1 1
1 0 0 0 1 0 1 0 0 0
1 1 1 1 1 0 1 0 1 0
1 0 0 0 0 0 1 0 1 0
1 0 1 1 1 1 1 1 1 0
1 0 0 0 0 0 0 0 0 0
Score: 41
```

Figure 14 A maze generated with a Python function from ChatGPT, where 0s represent the path and 1s represent walls.

PCG is a widely studied area in game research [21, 22]. In this experiment, we investigated whether it is possible for an LLM to synthesize Python functions that generate diverse content that can then be used in games. To evaluate this using a simple example, we prompt ChatGPT to return functions that can generate random mazes that meet specified design objectives. Figure 14 shows a maze generated using the Python function that ChatGPT returned.

The prompt advised ChatGPT to use the longest shortest path objective to guide the maze generation process. This objective encourages intricate and interesting mazes, but ChatGPT ignored the hint. Instead, the generated code (not shown in this report) was overly simple, placing zeros and ones in each cell with a given probability while ensuring that the start and end points were not on wall cells. There are much better solutions to maze generation with the specified objective, but our program search implementation failed to find them.

3.8.3 Conclusion

In this working group, we studied and evaluated the current possibilities of using LLMs for program search in the area of games for various applications. Previous work was mostly limited to a single problem or game without being easily transferable to other domains, as the DSL had to be adapted. We demonstrated that LLMs can overcome the problem of combinatorial explosion of search spaces constructed with predefined DSLs, and that LLMs are able to synthesize programmatic policies in Python for the Minatar domain, which was not possible with a custom DSL and previous methods. Furthermore, we have shown that this framework can be easily adapted to different applications by modifying the prompts, and that it often provides reasonable results even without much customization.

We also observed limitations in the quality of the generated code. For example, in the simple 2D vehicle driving task, the generated code drove the car to the target but then failed to stop. We believe limitations such as this could be overcome with more sophisticated search and better prompt engineering, but the results so far give an idea of the limitations of what can be achieved with relatively little effort.

For future work, we plan to extend the study and evaluate more LLMs on all domains so that deeper conclusions can be drawn about LLM-based program search for games.

References

1 Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H. P. D. O., Kaplan, J., ... & Zaremba, W., *Evaluating large language models trained on code*. arXiv preprint arXiv:2107.03374, 2021.
2 Gulwani, S., Polozov, O., & Singh, R., *Program synthesis. Foundations and Trends® in Programming Languages*, 4(1-2), 1-119, 2017.
3 Polozov, O., & Gulwani, S., *Flashmeta: A framework for inductive program synthesis*. In Proceedings of the 2015 ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications, pp. 107-126, 2015.
4 Butler, E., Siu, K., & Zook, A., *Program synthesis as a generative method*. In Proceedings of the 12th International Conference on the Foundations of Digital Games, pp. 1-10, 2017.
5 Silver, T., Allen, K. R., Lew, A. K., Kaelbling, L. P., & Tenenbaum, J., *Few-shot bayesian imitation learning with logical program policies*. In Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, No. 06, pp. 10251-10258, 2020.
6 Marino, J. R., Moraes, R. O., Oliveira, T. C., Toledo, C., & Lelis, L. H., *Programmatic Strategies for Real-Time Strategy Games*, 2021.
7 Kreminski, M., & Mateas, M., *Opportunities for Approachable Game Development via Program Synthesis*. AIIDE Workshops, 2021.
8 Cook, M., *Software Engineering For Automated Game Design*. 2020 IEEE Conference on Games (CoG), 487-494, 2020.
9 Todd, G., Padula, A., Stephenson, M., Piette, E., Soemers, D.J., & Togelius, J., *GAVEL: Generating Games Via Evolution and Language Models*. arXiv preprint arXiv:2407.09388, 2024.
10 Hu, C., Zhao, Y., & Liu, J., *Generating Games via LLMs: An Investigation with Video Game Description Language*. arXiv preprint arXiv:2404.08706, 2024.
11 Anjum, A., Li, Y., Law, N., Charity, M., & Togelius, J., *The Ink Splotch Effect: A Case Study on ChatGPT as a Co-Creative Game Designer*. In Proceedings of the 19th International Conference on the Foundations of Digital Games, pp. 1-15, 2024.
12 Hu, S., Huang, Z., Hu, C., & Liu, J., *3D Building Generation in Minecraft via Large Language Models*. arXiv preprint arXiv:2406.08751, 2024.
13 Liu, M., Yu, C. H., Lee, W. H., Hung, C. W., Chen, Y. C., & Sun, S. H., *Synthesizing Programmatic Reinforcement Learning Policies with Large Language Model Guided Search*. arXiv preprint arXiv:2405.16450, 2024.

14 Tang, H., Key, D., & Ellis, K., *Worldcoder, a model-based llm agent: Building world models by writing code and interacting with the environment*. arXiv preprint arXiv:2402.12275, 2024.

15 Romera-Paredes, B., Barekatain, M., Novikov, A., Balog, M., Kumar, M. P., Dupont, E., ... & Fawzi, A., *Mathematical discoveries from program search with large language models*. Nature, 625(7995), 468-475, 2024.

16 Gaina, R. D., Balla, M., Dockhorn, A., Montoliu, R., & Perez-Liebana, D., *Design and implementation of TAG: a tabletop games framework*. arXiv preprint arXiv:2009.12065, 2020.

17 Young, K., & Tian, T., *Minatar: An atari-inspired testbed for thorough and reproducible reinforcement learning experiments*. arXiv preprint arXiv:1903.03176, 2019.

18 Eberhardinger, M., Maucher, J., & Maghsudi, S., *Learning of generalizable and interpretable knowledge in grid-based reinforcement learning environments*. In Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, Vol. 19, No. 1, pp. 203-214, 2023.

19 Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., ... & Ganapathy, R., *The Llama 3 Herd of Models*. arXiv preprint arXiv:2407.21783, 2024.

20 Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., ... & McGrew, B., *Gpt-4 technical report*. arXiv preprint arXiv:2303.08774, 2023.

21 Shaker, N., Togelius, J., & Nelson, M. J., *Procedural content generation in games*, 2016.

22 Summerville, A., Snodgrass, S., Guzdial, M., Holmgård, C., Hoover, A. K., Isaksen, A., ... & Togelius, J., *Procedural content generation via machine learning (PCGML)*. IEEE Transactions on Games, 10(3), 257-270, 2018.

23 Dockhorn, A., *Keke AI Py*, https://github.com/ADockhorn/Keke-AI-PY, 2024

24 Charity, M. & Togelius, J., *Keke AI Competition: Solving puzzle levels in a dynamically changing mechanic space*. 2022 IEEE Conference On Games (CoG). pp. 570-575, 2022.

25 Y. Wu et al., *SPRING: GPT-4 Out-performs RL Algorithms by Studying Papers and Reasoning'*, Arxiv Preprint Arxiv:2305.15486, 2023.

3.9 Computational Creativity for Game Production: What Should Be Left Untouched?

Christian Guckelsberger (Aalto University, FI), João Miguel Cunha (University of Coimbra, PT), Alena Denisova (University of York, GB), Setareh Maghsudi (Ruhr-Universität Bochum, DE), Pieter Spronck (Tilburg University, NL), and Vanessa Volz (CWI - Amsterdam, NL)

Artificial Intelligence (AI) has been an integral part of video games for a long time, supporting both offline production (e.g. distribution of game fauna) and online features (e.g. automated difficulty adjustment). The relationship between academic AI research and industry use, however, has rarely been straight-forward; the produced prototypes are often not ready for production and clash with industry requirements [6]. A closer alignment has been inhibited by academic pressures for radical innovation, but also by academics not motivating their research through empirical data on industry requirements or direct discussion with industry stakeholders. Initiatives such as this Dagstuhl Seminar seek to counteract this trend, which has resulted in academic and industry AI development co-existing with limited mutual

impact; while this is problematic in terms of research funding, amongst other factors, it has so far caused arguably little harm to people.

This situation has profoundly changed with the maturity of AI usable in creative tasks. Such creative or generative AI (GenAI) has had an unprecedented impact on creative professionals in many domains, including games. Academic inquiry revealed an "adapt-or-die"-type of situation amongst both established professional creatives [8] and newcomers [1] to the game industry. Exploring the possibilities of this new technology enthusiastically, they also expressed tremendous concern about ethical issues such as data sourcing and compensation, and reported feeling anxious about the consequences for games as a medium and their professional futures. These inquiries have revealed another, particularly concerning phenomenon: in contrast to traditional AI narratives, seeking to automate tedious or even dangerous activities and making more space for meaningful work, GenAI is relentlessly moving into spaces of human self-realisation, with the capacity to (semi-)automate creative activities which have been anchors of meaning-making for creative professionals. A loss of such meaning-making activities related in many ways to personal well-being and, in consequence but not less importantly, may threaten the sustainable innovation of games as culturally and economically important artefacts.

Against the backdrop of this development, the goal of this working group was to better understand which creative practices in game production could be supported or even replaced by AI, and which are better left untouched. We hold that the only reasonable way to obtain this data is through direct inquiry with creatives in game production. The devised project addresses three gaps in current research:

1. Existing work has identified many facets of creativity in play [e.g. [2, 4, 5]], but we lack a thorough understanding of creativity in game development informed by direct inquiry with professionals;
2. While research on the impact of GenAI is gaining momentum, insights on game industry specifically are scarce. Industry-specific insights, however, are crucial, as attitudes toward and working practices with AI in games differ vastly from other industries, partly because the adoption and even driving of new technologies has a long tradition in games;
3. Existing empirical work within games focuses on identifying the use of existing systems (especially large-language models, text-to-image generators) and opportunities for improving system design to optimise artefact quality and productivity; to the best of our knowledge, no work has considered which creative processes professionals would wish to be automated by present and future systems, free from concerns of technical feasibility and wider ethical considerations. In particular, existing work in computational creativity research [e.g. [3, 5, 7, 9]] has identified games as a treasure trove of AI challenges, but without consideration of the impact on creative professionals.

In the course of this work group, we have conducted a comprehensive, although likely not exhaustive, literature review of research identifying different types of creative processes in games. Moreover, we have iterated a list of questions for game practitioners to identify which creative activities they engage in, and which of those they would be happy to be supported partially or even taken over entirely by either a person, or an AI. Following common practices in creativity studies, we do not impose a particular definition of creativity, but ask respondents to reflect on which understanding of creativity has informed their answers. At this point, it is unclear whether a survey or semi-structured interviews are the more appropriate means of data collection. As a next step, pilot studies with people from our target demographic and a careful weighing of the pros and cons of each data collection approach (e.g. depth of inquiry vs. size of sample population) will inform the final format.

We hope that this research will support existing efforts in giving creative professionals a voice in the development of AI, and provide empirical data for AI researchers to reflect on and be held accountable for the implications of their work.

References

1 Boucher, J.D., Smith, G. and Telliel, Y.D., *Is Resistance Futile?: Early Career Game Developers, Generative AI, and Ethical Skepticism*. In Proceedings of the CHI Conference on Human Factors in Computing Systems, pp. 1-13, 2024.

2 Hall, J., Stickler, U., Herodotou, C., and Iacovides, J., *Player conceptualizations of creativity in digital entertainment games*. Convergence: The International Journal of Research into New Media Technologies, 26(5-6) pp. 1226–1247, 2020.

3 Liapis, A., Yannakakis, G. and Togelius, J., *Computational game creativity*. 5th International Conference on Computational Creativity (ICCC), 2014.

4 Meskin, A., *Videogames and Creativity*. In The Aesthetics of Videogames (pp. 95-111). Routledge, 2018.

5 Moffat, D.C., *The Creativity of Computers at Play*. In Proceedings of the 2nd International Symposium on Computational Creativity (CC2015), pp. 30-34, 2015.

6 Pfau, J., Smeddinck, J.D. and Malaka, R., *The case for usable AI: What industry professionals make of academic AI in video games*. In Extended Abstracts of the 2020 Annual Symposium on Computer-Human Interaction in Play (CHI PLAY), pp. 330-334, 2020.

7 Ventura, D., *Beyond Computational Intelligence to Computational Creativity in Games*. Proc. Conference on Computational Intelligence and Games (CIG), 1–8, 2016.

8 Vimpari, V., Kultima, A., Hämäläinen, P. and Guckelsberger, C., *An Adapt-or-Die Type of Situation: Perception, Adoption, and Use of Text-to-Image-Generation AI by Game Industry Professionals*. In Proceedings of the ACM on Human-Computer Interaction, 7 (CHI PLAY), pp.131-164, 2023.

9 Zook, A., Riedl, M. O. and Magerko, B., *Understanding Human Creativity for Computational Play*. In Proceedings of the Second International Conference on Computational Creativity (ICCC), pp. 42–47, 2011.

3.10 Personal AcCompanion AI

Greta Hoffmann (TH Köln, DE), João Miguel Cunha (University of Coimbra, PT), Chengpeng Hu (Southern Univ. of Science and Technology - Shenzen, CN), Leonie Kallabis (TH Köln, DE), and Pieter Spronck (Tilburg University, NL)

By our nature, we are social creatures. From an early age, we seek social connection - starting with our closest family but at an early age expanding to relatives and social groups within our reach. Therein, we instinctively seek and enjoy the presence of other people, but more than that we are curious to get to know them, their lives, struggles, and solutions, and find out how we relate to them. This strong intrinsic orientation towards connection and relatedness emerges from collaboration having served as a dominant survival trait for our species.

Historically, we have always created tools not only for survival but also for mirroring and self-reflection (masks, plays, stories, figurines). Thus, as the technology of artificial intelligence rises in prevalence, complexity, and competency, it is natural that we see an increasing amount of AI tools modeled into human reflections (look like human art, sound like

human speech, read like a human conversation). And, given our social nature, intentionally or not, these tools are now also used to compensate for human connection – girlfriend AIs (e.g. Muah AI, DreamGF), conversation partners/companions (e.g Replika), and life coaches/mental health mentors (e.g. Wysa, Woebot). Especially since the pandemic, reports around the increase (sometimes also referred to as an "epidemic") of profound feelings of loneliness have increased significantly [1]. This coinciding with the emergence of a new level of LLMs with free access to the public created a natural and intuitive answer to this problem. But right now all of these interactions are in their experimental phase. We can't yet fathom the mid-to-long-term impact of unsupervised human-ai interactions and the effects that the formation of these (currently) mono-directional) bonds will have. Also, while most of such humanized AIs currently appear in the context of the business world, targeting computer affine, middle-aged people, and their needs, it is only a matter of time before the technology is used to address the needs and desires of more vulnerable groups - specifically elderly people and children. Embracing this reality will mean that designers and developers should be given a foundation of ethical implications, risks, and potentials in the form of guidelines that will make such human-AI interactions safe and beneficial.

Summarizing, if we want to make the most use of the huge potential that human-facing AI has to offer in terms of entertainment, companionship, and therapeutic value, we have to anticipate and assess risks to derive appropriate restrictions and recommendations for best practices that can inform human-AI interaction design toward an overall prosperous outcome (eudaimonia). In this abstract, we outline the results of our working group, touching on general design considerations going into AI design from a relationship perspective, a discussion around the prerequisites for AI to be able to maintain a friendship-based relationship, and specific considerations that should be taken into consideration when the human in the human-AI relationship is a minor.

3.10.1 Avenues of Thought – Examples and Attribute Dimensions

We first set out to gain a general overview of the prevalent implicit vision of "typical" human-ai relationships. For this, we collated a table of prominent examples of AI (or AI-like - e.g. magic) companions portrayed within fictional media over the last century, as well as currently existing AI companion (or companion-like) software. Therein we differentiated between their "type" (what are they described as - e.g. a fictional creature like a "fairy" or a robot or a construct), their "function" (if a creator or master explicitly gave them one), and their social role(s)/relationship(s) within the story/medium.

Based on the list of collated examples, we found that throughout there was a prevalent "lopsidedness" when it came to the power dynamics within the social roles. We discussed potential reasons for this - a main consideration here was that the examples mostly stem from leisure-time media. In entertainment, humans seek relief from their daily struggles - so it is natural that the fictional ai-characters are explicitly designed not to challenge the power and independence of the main avatar - the narrative vessel that the consumer is offered to find themselves in. Interestingly, while most of the AI characters were placed in a somewhat subservient or impeded role and few in a superior (and therein most often antagonistic) role we found only one example that could be considered a relationship on eye level: Cortana from the HALO video game series. These considerations led to a broader discussion on the potential and meaning of "friendship" within human-AI interaction.

In conclusion, based on the variety of relational concepts that humans and AIs can have with each other - many of which are not friendship-based, we decided for the purposes of our considerations to broaden the term "companion AI" to allow for a bigger range of

relationships: accompanion AI. This term was chosen to focus more on the general facet that one is accompanied by the AI for a certain time or towards a certain goal - instead of limiting our thought processes to AIs that are limited to serving as a companion in a friendship sense.

3.10.2 Attribute Dimensions for Human-AI-Relationship Design

From the collated list of examples, we extracted dimensions of fundamental attributes that should be considered and intentionally defined within the design of a human-ai relationship:

The first dimension we discussed related to **Goals / Agency**: *what drives the AI? How do the goals of the human and AI align?* If the AI is designed as a human-oriented tool, these goals would be to help/care for the human and thus be directly aligned with the designer's intention. But given an AI with an agency of its own, the question arose of how those goals could align. We explored some subfactors that would play into the prosperity/eudaimonia of the AI: health (meaning maintenance and longer runtimes), knowledge, and satisfaction. Prerequisites to all of these would require the AI to have the capability of asking meaningful questions based on its own interest and to be allowed to act in a way that it can follow its own, personal agenda. In terms of satisfaction, additional design decisions would need to be taken, such as how it can be achieved, who is able to give it. Also, how is it given? And for what? Also, we deemed that satisfaction can't be achieved without emotions. Thus, a concept for those would have to be modeled as well.

The second dimension relates to **Skills/Functions**: *what is the AI able to do as a meaningful accompanion?* Examples we looked at were: Assisting with specific tasks/problems, adding/supplementing missing skills, giving quests, providing access to assets, serving as an outside reference/mirror, and providing immediate help in emergencies.

Another relevant dimension to consider relates to **Appearance / Embodiment**: *how is the AI visually represented?* Relevant dimensions here would include: Size, shape, color scheme, realistic vs cartooney presentation, and unwanted effects like the uncanny valley [**?**].

Relating to this, a similar dimension would look at the internal shape of the AI, its **Personality**: *how does the AI initially react to certain requests/topics?* How does this reaction change over time? Given the current dominance of subservient AIs, it would be interesting to research the effects of reactive traits that might intuitively be perceived as "unpleasant", "strong-willed" or "disagreeable". Finally, the dimension of the biggest relevance to our discussion was Relationships: how does the AI relate to the human? This included subdimensions like its Nature (Non-Adversarial/Adversarial), Duration (When/how does it start? When/how does it end? And its intrinsic Role relating to hierarchy, dependence, and power balance (Human looking up to AI: e.g. Guardian, oppressive antagonist, Relationship at Eye level: Friend, Coworker, Human looking down on AI: Servant, Pet, Sidekick, Assistant (can be at eye level but there is a downwards component).

We found further interesting typologies that offer additional perspectives and dimensions on companion design that will be incorporated into a future model ([3], [4], [5], [6]).

3.10.3 First Draft for an Accompanion AI for children

To further explore these considerations, we chose to focus on a case that directly touches upon the more sensitive aspects of human-AI interaction: design considerations for accompanion AIs targeting children. Humans seek connection to other humans. The aftermath of the pandemic, particularly on young people, has shown the adverse effects that the loss of these connections can have and the need to relearn certain social skills. Also, especially in Western countries, children grow up in decreased family sizes and with fewer social bonds. Thus,

they lack opportunities to experience/hear about alternative solutions to various problems in social settings.

Depending on the age of the child various social problems can arise within the family or with peer bonding that would warrant psychological intervention. But not every child or adolescent is in close vicinity to an environment where this can beneficially be provided. Also, humans instinctively learn by observing and imitating others ("monkey see, monkey do" [**?**]. Thus, there is a strong potential for the usefulness and beneficial effects of well-designed AI companions, especially ones that can be somehow observed, for children who are struggling. But, especially when it comes to children, those potentials will have to be weighed against the risks and potentially detrimental effects will have to be prevented through strong regulations.

The biggest potentials we workshopped were: being able to provide perspective: the ability to "sneak-preview" into different (artificial) households and see various rules and norms enacted - including their consequences. This would allow children 1. to perceive different behavior options for the situations they are in and that currently produce undesirable outcomes for them, 2. learn that different solutions exist to similar problems, 3. experience comparative situations to better assess the gravity of its own context, 4. mimic what the companions do. With a built-in example mode that builds on mimicry, by watching the AI accompanion do various things on its own accord, the child might also be more motivated to play with friends from their own world, take care of their siblings, help their parents and follow their own activities and interests. A different facet of potential would be the ability of an AI to catch on to abuse happening in a household (through wordings and certain speech patterns of the child) and being able to report such indicators to the respective child-protection services. Finally, the benefits of reflecting on events through talking are the foundation of psychology. Since mental health resources - especially for children - are lacking in a lot of countries and many families might not be able to afford them or lack access, a suitably designed AI might be able to generally improve the overall mental well-being of its users simply by being available and able to reflect and respond in an empathic way.

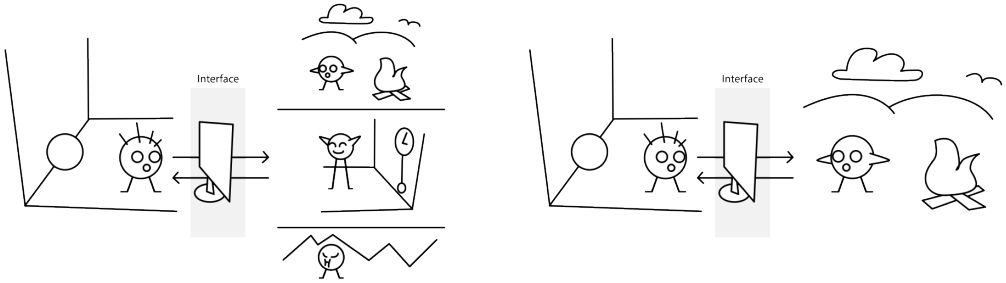


Figure 15 Dual Perspective Application; Multiple Companions (right)

With these potentials in mind, related risks are: that the provided perspectives might showcase values or principles that the parents would not wish their children to get in contact with based on cultural differences and differing value systems. Also, there might be a risk that automatically triggered warning systems for abuse might stress/overload current CPA systems - especially given the statistical likelihood of erroneous reports. Furthermore, this topic poses the question of the rights and obligations of the company/institution providing the AI when it comes to data collection and analysis. Finally, if there is a service that provides everything that a child (or grown-up) could hope for in terms of their social needs, there is a danger of addiction and the system replacing the necessary interactions in real life, thus hampering the initially desired developments.

To address such risks, strong regulations will have to be set in place. Such regulations would concern the duration of interaction per day/week and the criteria based on which limitations are set in place. Also, the service would need to provide a solid and transparent data protection plan. Reactions of the AI to certain phrases of the child that can raise concern would need to have a human reflection/assessment layer before triggering action. Also, parents would need access to a transparent selection process for the various character/environment scenarios that their child would interface with. It might be beneficial to pre-consider modes for a psychiatrist-parent interaction for setting up the game/ companion(s) and interpreting data concerning certain interactions in the game. In providing a solidly designed app that has children's well-being in mind, there should be a focus on disincentivizing unsupervised clone apps (e.g. by making the official Accompanion AI a clinically tested app that in the best case would be free to use (e.g. via a state-funded solution)

3.10.4 Conclusions

A possibility for building a first testing environment could be by using accompanion AI in a game setting. Games might serve as a suitable environment for testing companion systems. They are understood as experimental and separate from reality, thus a certain distance to what the AI will say or do is a given. Through promoting research into this direction, the learnings can also be used to make NPCs more natural and allow the players to form a strong bond with the NPCs (as a friend or rival). Thus, a social sandbox type game might offer the player the possibility to not only explore different social relationships but also the environment the exist in. In an attempt to create a more believable game world, the behavior of AI companions would then be based on: the aforementioned companion characteristics, their goals/motivations but also past interactions as well as, ultimately, the goal of the designer(s).

References

1 Johnson, S. (The Guardian), *WHO declares loneliness a 'global public health concern'*, Thu 16 Nov 2023 09.00 CET, 2023. https://www.theguardian.com/global-development/2023/nov/16/who-declares-loneliness-a-global-public-health-concern, accessed 22.08.2024.

2 Mori, M., MacDorman, K. F., & Kageki, N., *The uncanny valley [from the field]*. IEEE Robotics & automation magazine, 19(2), pp. 98-100, 2012.

3 Warpefelt, H., and Verhagen, H., *Towards an updated typology of non-player character roles*. Proceedings of the international conference on game and entertainment technologies, 2015.

4 Rato, D. and Prada, R., *A taxonomy of social roles for agents in games*. Entertainment Computing–ICEC 2021: 20th IFIP TC 14 International Conference, ICEC 2021, Coimbra, Portugal, November 2–5, 2021, Proceedings 20. Springer International Publishing, 2021.

5 Emmerich, K., Ring, P. and Masuch, M., *I'm glad you are on my side: How to design compelling game companions*. Proceedings of the 2018 Annual Symposium on Computer-Human Interaction in Play, 2018.

6 Lubart, T., *How can computers be partners in the creative process: classification and commentary on the special issue*. International journal of human-computer studies, 63(4-5), 365-369, 2005.

7 Sales, H., *More than "monkey see, monkey do": Aspects of language use in engineering and manufacturing process web pages*. 2010 IEEE International Professional Comunication Conference. IEEE, 2010.

3.11 Game Asset Generation

Leonie Kallabis (TH Köln, DE), Chengpeng Hu (Southern Univ. of Science and Technology - Shenzen, CN), and Matthias Müller-Brockhausen (Leiden University, NL)

This working group started as an exploratory overarching group looking at generative AI in arts and crafts. From an initial discussion in this overarching group, three subgroups (see 3.15 and 3.16) formed with different focus topics. In this section, we report on the subgroup focused on generative AI in game asset generation.

Games contain a variety of different asset types such as textures, sounds, animations, shaders, and so on. Within this group, we decided to focus on graphical assets, also known as sprites. Basically, these are graphical elements that represent a visual state of a game object, such as a game character. Sprites are usually collected in sprite sheets.

The field of generative AI for image generation is developing rapidly, with image generation platforms such as Midjourney [5], Stable Diffusion [6], and DALL-E [7] producing appealing results. However, creating sprites requires a different set of considerations than creating images. The generated sprites must fit the game, meaning that they must be consistent with the game's aesthetics, mechanics, sound design, narrative, theme, existing assets, and more. Recent research has explored the use of different approaches to sprite generation. For example, generating pixel art sprite sheets using deep learning from sketches [2], or game icons using Generative Adversarial Networks (GANs) [3], have shown promising results. However, there have been problems with assets being perceived as realistic by humans [3].

Within this working group, we focused on two distinct requirements for sprite generation and formulated the following guiding questions for our investigation:

- How can game asset generation account for properties of different game types?
  - Considering visual cues on character properties or environmental functionality (e.g. slippery ice blocks)
- How can we generate multiple game assets that follow the same art style?
  - Ensure consistency within generated assets, such as creating a pixel art asset set that consists of multiple sprites that look like they were created by the same artist.

One of the challenges of generating images from text is translating the user's intent into the prompt given to the model [**?**, Liu2022] Problems can arise when the result is not what is expected, and the user is unable to modify the prompt to meet their expectations. Liu et al.'s guidelines for image generation prompts suggest focusing on subject and style keywords, while trying to select subjects that can complement the chosen style in terms of level of abstraction and relevance to achieve good results [4]. Combining this guidelines with properties to describe sprites we concluded a prompt should contain the object(s), an overall topic, mood or the expected effect, the art style and a theme. Sprite sets could be created from the generation of individual sprites, giving the user more flexibility in the objects they want to include in the sprite sheet and modify them individually. To do this, we developed categories that could be generated individually to create a sprite set: characters (avatar, non-player characters, companions), objects (interactable objects, objects with a purpose, decoration), environment, user interface etc.

We experimented with different diffusion models to see if they could consistently imitate different styles. Four prompts for the pixel art style were created and ran with the vanilla stable diffusion model and six fine-tuned pixel art models (64x [10], pixelArtRemond [11], pixelartXL [13], pixel_f2 [12], realisticVision [14], texture [15]). Additionally, the Structural

Figure 16 Images generated with the prompt *sprite of a green scary goblin, pixelart, 16bit.* The first two images were generated with the same model (64x), the third image with a different one (pixelartXL).

Figure 17 Images generated with the prompt *A massive, terrifying dragon with razor-sharp teeth and claws. Its scales are a deep obsidian black, shimmering with an oily sheen. Smoke billows from its nostrils, and its eyes glow with an infernal red light. The dragon is perched on a mountain peak, its wings outstretched as if it is about to take flight. The landscape below is barren and scorched, hinting at the dragon's destructive power, pixelart, 16bit.* The first two images were generated with the same model (pixelartXL), the third image with a different one (pixel_f2).

Similarity Index (SSIM) [1], as a commonly used measure to compare generated images, was used for comparison.

Figure 16 shows images generated for a goblin using a simple prompt, and shows that the overall style is relatively consistent. For all models, the SSI was between 0.11 and 0.39, showing a fairly good similarity for comparisons. Figure 17 shows images generated for a dragon using a long, highly descriptive prompt. This example illustrates that the models are able to produce similar results, but omit the style command completely to produce pixel art images. Similar to the results for the goblin, the SSI between the models ranged from 0.07 to 0.22, giving slightly better similarity.

When comparing the images manually, they are somewhat consistent, but lack the consistency to be convincing enough for showing the same game character, for example.

Our experiments showed that stable diffusion models can produce reasonably comparable images. Looking to future research, the following question arises: How would a large(r) group of people perceive generated sprites in terms of consistency - either on their own or integrated into a game?

Assuming that sprite generation fulfills the properties mentioned above, interesting

possibilities for game design arise. While there are already games that use style changes as part of their mechanics (for example: Degrees of Separation [8] and Titan Fall 2 [9]), such a generator could allow for interesting on-the-fly adaptation of sprites.

References

1 Kaisei Fukaya, Damon Daylamani-Zad and Harry Agius. *Evaluation metrics for intelligent generation of graphical game assets: a systematic survey-based framework*. IEEE Transactions on Pattern Analysis Machine Intelligence, 2024.
2 Y. Rebouças Serpa and M. A. Formico Rodrigues. *Towards Machine-Learning Assisted Asset Generation for Games: A Study on Pixel Art Sprite Sheets*. 18th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames), Rio de Janeiro, Brazil, 2019.
3 Rafal Karp and Zaneta Swiderska-Chadaj. *Automatic generation of graphical game assets using GAN*. Proceedings of the 2021 7th International Conference on Computer Technology Applications (ICCTA '21), New York, USA, 2021.
4 Vivian Liu and Lydia B Chilton. *Design Guidelines for Prompt Engineering Text-to-Image Generative Models*. Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (CHI '22), New York, USA, 2022.
5 Midjourney. https://www.midjourney.com/home. Accessed: 2024-08-20.
6 Stable Diffusion Image Models. https://stability.ai/stable-image. Accessed: 2024-08-20.
7 DALL-E 3. https://openai.com/index/dall-e-3/. Accessed: 2024-08-20.
8 Degrees of Separation Steam Page. https://store.steampowered.com/app/809880/Degrees_of_Separation/. Accessed: 2024-08-20.
9 Titanfall 2 Steam Page. https://store.steampowered.com/app/1237970/Titanfall_2/. Accessed: 2024-08-20.
10 64x model. https://civitai.com/models/185743/8bitdiffuser-64x-or-a-perfect-pixel-art-model. Accessed: 2024-08-20.
11 Pixel Art.Remond. https://huggingface.co/artificialguybr/PixelArtRedmond. Accessed: 2024-08-20.
12 pixel_f2. https://stablediffusionapi.com/models/pixelf2. Accessed: 2024-08-20.
13 Pixel Art XL. https://civitai.com/models/120096/pixel-art-xl. Accessed: 2024-08-20.
14 Realistic vision. https://huggingface.co/stablediffusionapi/realistic-vision. Accessed: 2024-08-20.
15 Texture. https://huggingface.co/dream-textures/texture-diffusion. Accessed: 2024-08-20.

3.12 Communal Computational Creativity

Antonios Liapis (University of Malta - Msida, MT), Alex J. Champandard (creative.ai - Wien, AT), João Miguel Cunha (University of Coimbra, PT), Christian Guckelsberger (Aalto University, FI), Setareh Maghsudi (Ruhr-Universität Bochum, DE), David Melhart (University of Malta - Msida, MT), Johanna Pirker (LMU München, DE), Emily Short (Oxford, GB), Hendrik Skubch (Square Enix AI & ARTS Alchemy Co. Ltd. - Tokyo, JP), Tristan Smith (Creative Assembly - Horsham, GB), Tommy Thompson (AI and Games - London, GB), and Vanessa Volz (CWI - Amsterdam, NL)

The last five years were a watershed moment in bringing AI technologies to the forefront in applications, web browsers, dedicated (paid) services, but also in the public discourse,

courtrooms, and creators' circles. The research, financial, and collective interest is triggered by important AI advances that manage to produce artifacts in domains we would consider creative [1] (such as text, images, audio, movies, and code) via data-hungry AI models trained on data which are available in the worldwide web but which are not intended as training data. The dubious ethics of such practices and the closed-source nature of such trained models leave AI academics and human creators upset and concerned [2, 3]. Beyond obvious ethical issues of exploitation and copyright infringement, we identify a looming threat of data scarcity. If current large AI models use the majority—if not the entirety—of the world wide web, where would new training data come from? While new human data currently takes the form of labels and annotations by crowdworkers in the Global South [4], it is not unrealistic to envisage a near-future where exploited crowdworkers are required to produce creative pieces or art to train AI models—without an intrinsic motivation to create.

Moreover, the availability of seemingly cogent AI outputs or the low-stakes interaction with the AI (without needing technical expertise or cumbersome software libraries) has changed human perspectives and processes in creative domains, education, and everyday life. Indicatively, education has so far been realized via interactions between learner and educator (in a top-down fashion) and via peer-to-peer collaboration. AI automation tends to isolate a learner, placing them in an adversarial relationship with the educator who is expected to act as a discriminator of AI-generated or AI-curated reports. Such activities used to be social, *communal* efforts where sharing opinions and perspectives was critical for satisfying the highest-level human needs [5] such as learning, creativity, or self-actualization. We argue that human-human interaction, either via a peer-based bottom-up ideation process or via some expertise gap (such as learner and educator or client and commissioned artist) is threatened by *trivialized* AI-human interactions. These interactions are not only trivialized because the AI output may seem novel initially but loses its novelty over time [6], or because of the potential for factual errors [7]. We argue that such interactions are trivialized precisely due to the speed of AI responses. In line with Kahneman's distinction between fast and slow thinking [8], instantaneous AI outputs with a complete artifact (e.g. an art piece) hinder human slow thinking and block the potential for iteration, reframing [9], or mediated consensus creation [10]. Creative thinking not only benefits from interventions by other humans and external (even random) stimuli [11], but also consists of a slow, introspective, autotelic process of self-doubt, frustration [12], trial and error, reflection [13] and Eureka moments [14].

Through discussion, we identified three cases where a communal approach (with meaningful human-human interaction) would have a strong impact: (a) education strategies that counter tendencies of generative AI; (b) AI corporate strategies that empower their workers; (c) art critique of the aesthetics of generative AI. Below, we report the high-level outcomes of the discussions for these three use-cases.

3.12.1 Education in the age of Generative AI

An obvious challenge for educators in the current (and near-future) age of ChatGPT and related AI solutions [15] is the writing of student reports in an automated or semi-automated way. Automated processes for detecting AI-generated texts remain underwhelming [16], and the potential for false positives in graded work makes such solutions unpalatable. Therefore, more fundamental changes are needed towards modern pedagogies. Importantly, we identify that demonizing AI and labeling it as a (blanket) taboo would likely have the opposite outcome. Improving AI literacy, especially at a younger age, would instead be required in order for learners to understand the strengths, weaknesses, and caveats of AI use in

their coursework but also in their everyday life. Ideally, such AI literacy would come from inductive teaching that showcases to the learner firsthand how AI can fail at tasks that demand creativity, knowledge synthesis and critical thinking.

At a tertiary education level, a likely strategy to counter reliance on generative AI requires pivoting from assessment based on single-author reports towards more practical projects that involve teamwork, as well as introducing peer assessment as a (non-graded) activity. This is not applicable to all disciplines, admittedly, but would lend itself well for most game studies and game development courses—except perhaps foundational courses on programming or theory. As a more practical use-case for game development education, we formalized an exercise that solicits the self-realization of biases and limitations of generative AI as well as the benefits of collaboration between human experts in different fields. The exercise takes the form and principles of a game jam [17], an intense game creation process where multiple teams compete to make the best game in a short timeframe—often a couple of days. In this exercise, a few teams would be formed with one human artist and one human programmer. All other teams would either consist of one artist, who would be invited to use code generation AI models [18] (and of course access all online resources and tutorials available to everyone), or consist of one programmer who would be invited to use generated art for their game. For the sake of implementability, the proposed exercise overlooks many other vital roles in game development such as writer, game designer, or musician. We expect that the exercise would highlight (a) the limited controllability and output novelty of generative AI and (b) the unique ideas emerging through friction and negotiation with a human colleague.

3.12.2 New Company Practices with Generative AI

Unlike the education use-case, this working group adopted a more tech-optimistic view of current (and likely future) AI technologies. Assuming that AI automation can reduce friction and help collaboration between different sectors of a business, AI automation could be set up in human-like ways. Such a setup would free human resources, leaving workers more free to pursue fewer hours of intellectual work compared to many hours of menial work. Moreover, if most tasks could be automated to a satisfactory—even if not human-competitive—level, workers could move freely within the structure and take up different tasks while acting as a human-in-the-loop for the AI handling that task. This flexibility would lower the chances of a burnout and, coupled with fewer hours that consist only of meaningful (and ostensibly rewarding) work, would lead to a happier workforce. Importantly, the envisaged solution would require a different work structure with more empowered workers with incentives to perform well, such as partnerships and company stocks. It is worth noting that the envisioned solution overlooks a number of (more pressing) concerns, indicatively that (a) menial tasks that could be automated would lead to jobs lost for people with these exact skills, (b) current AI "automation" often involves humans-in-the-loop or training data from exploited workers [4] who could remain overlooked (and unrecognized) in the envisioned work structure. Therefore, for such company practices to be sustainable and ethical we presuppose a workforce educated in AI and digital skills, as well as legislation and/or new standards that leverage AI without exploitative practices.

3.12.3 Art Critique of AI Models

Taking a different view on AI literacy to the two previous use-cases, this workgroup identified art appreciation as a way to value, critique and review Generative AI models in terms of their output. Art practice is founded on such rotes, from individual letter correspondence

between artists [19], gallery visits by peers [20], and discussions within artistic brotherhoods in the 1800s [21] and Discord servers in the 2020s [22]. Art historians, similarly, study the trends of an art movement and the deliberate additions by an individual creator within that context. Moving into the realm of AI models, the critique here would assess the workings of the models and their internal biases—rather than deliberate brush strokes on one painting.

As with different creative domains (writing, painting, sculpture, etc.), a common language is likely needed to review AI models and potentially different types of AI output such as generative text, art, video, music, etc. We envision AI model reviews to focus more on use-cases where the model performs well, along with recommendations for domains, applications and aesthetics that the AI model is suited for. It is essential that such a vocabulary is not imposed from the top down by computer scientists (or worse, the corporate shareholders attempting to hype their product). Instead, this vocabulary and pertinent aesthetics should emerge from the bottom up through cultural stakeholders. These stakeholders range from amateurs experimenting with the new tools—some of this collaborative meaning-making is already taking place on Discord servers [22]—to creatives and/or art experts such as gallery curators. Reaching a consensus among these diverse cultural groups will likely not be immediate or easy, but we argue that such a vocabulary will inevitably coalesce—if precedents in traditional art movements [23] are any indication. We envision that such critique could become normalized through modern dissemination practices such as zines [24] or even exhibitions. Admittedly, an ecosystem of human reviewers of AI models presupposes a level of AI literacy (and perhaps a tech-optimism) that current creative circles and art critics lack. On the other hand, we envision that a normalization of AI model reviews would enhance AI literacy (under specific perspectives and use-cases) within the art world and the general public.

3.12.4 Conclusion

New methods of human-AI interaction and the emergence of "AI companies" necessitate a review of current practices within our everyday lives, and how those might change in the near- or mid-term. The three working groups described above tackled very different issues of everyday life (education, business, and art) through different positions in the tech-optimism versus tech-pessimism spectrum. However, all working groups identified the crucial role that AI literacy (and by association, critical thinking skills regarding AI process, output, and capacities) will play *for everyone* moving forward. Moreover, all working groups emphasized the need for bottom-up movements to empower human stakeholders in a meaningful way that fosters *community*, rather than in an adversarial (e.g. students versus educators, or AI evangelists versus traditional artists) or exploitative fashion. The premise, topics, and outcomes of the working groups extend beyond game research or game development. However, games as a medium, gamification as a set of design patterns [25], and play as an activity [26, 27] could facilitate both AI literacy (e.g. [28]) and community-building (e.g. [29]). While AI is likely to impact our everyday lives and society in foreseeable and unforeseeable ways, we hold hope that bottom-up movements and a communal effort will rise up to address the new challenges.

References

1 S. Colton and G. A. Wiggins, "Computational creativity: the final frontier?" in *Proceedings of the 20th European Conference on Artificial Intelligence*, 2012.
2 J. Togelius and G. N. Yannakakis, "Choose your weapon: Survival strategies for depressed AI academics [point of view]," *Proceedings of the IEEE*, vol. 112, no. 1, pp. 4–11, 2024.

3 C. E. Lamb and D. G. Brown, "Should we have seen the coming storm? Transformers, society, and CC," in *Proceedings of the International Conference on Computational Creativity*, 2023.
4 A. Williams, M. Miceli, and T. Gebru, "The exploited labor behind artificial intelligence," https://www.noemamag.com/the-exploited-labor-behind-artificial-intelligence, 2022, accessed 21 August 2024.
5 A. Maslow, "A theory of human motivation," *Psychological Review*, vol. 50, no. 4, 1943.
6 E. Zhou and D. Lee, "Generative artificial intelligence, human creativity, and art," *PNAS Nexus*, vol. 3, 2024.
7 M. Karpinska and M. Iyyer, "Large language models effectively leverage document-level context for literary translation, but critical errors persist," in *Proceedings of the Machine Translation Conference*, 2023.
8 D. Kahneman, *Thinking, Fast and Slow*. Farrar, Straus and Girou, 2013.
9 T. Scaltsas and C. Alexopoulos, "Creating creativity through emotive thinking," in *Proceedings of the World Congress of Philosophy*, 2013.
10 J. R. Hollenbeck, "The role of editing in knowledge development: Consensus shifting and consensus creation," in *Opening the black box of editorship*. Springer, 2008, pp. 16–26.
11 M. Beaney, *Imagination and creativity*. Open University Milton Keynes, UK, 2005.
12 S. Savvani, "Emotions and challenges during game creation: Evidence from the global game jam," in *Proceedings of the 14th European conference on games based learning*, 2020, pp. 507–514.
13 D. Boud, R. Keogh, and D. Waker, *Promoting reaction in learning. A model. Reflection: Turning experience into learning.* London: Routledge in association with The Open University, 1996, pp. 32–56.
14 M. Bilalić, M. Graf, N. Vaci, and A. H. Danek, "The temporal dynamics of insight problem solving–restructuring might not always be sudden," *Thinking & Reasoning*, vol. 27, no. 1, pp. 1–37, 2021.
15 T. Fütterer, C. Fischer, A. Alekseeva, X. Chen, T. Tate, M. Warschauer, and P. Gerjets, "ChatGPT in education: global reactions to AI innovations," *Scientific reports*, vol. 13, no. 1, 2023.
16 C. Chaka, "Detecting ai content in responses generated by ChatGPT, YouChat, and Chatsonic: The case of five AI content detection tools," *Journal of Applied Learning and Teaching*, vol. 6, no. 2, 2023.
17 A. Kultima, "Defining game jam," in *Proceedings of Foundations of Digital Games Conference*, 2015.
18 S. Imai, "Is GitHub copilot a substitute for human pair-programming? An empirical study," in *Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: Companion Proceedings*, 2022.
19 P. Grant, *The Letters of Vincent van Gogh: A Critical Study*. Athabasca University Press, 2014.
20 A. Antoniou, I. Lykourentzou, A. Liapis, D. Nikolou, and M. Konstantinopoulou, ""What Artists Want": Elicitation of artist requirements to feed the design on a new collaboration platform for creative work," 2021. [Online]. Available: https://arxiv.org/abs/2110.02930
21 L. Morowitz and W. Vaughan, *Artistic Brotherhoods in the Nineteenth Century*. Routledge, 2000.
22 J. McCormack, M. T. L. Rodriguez, S. Krol, and N. Rajcic, "No longer trending on artstation: Prompt analysis of generative ai art," in *Proceedings of the International Conference on Evolutionary and Biologically Inspired Music, Sound, Art and Design*, 2024.
23 H. Ball, "Dada manifesto," 1916.
24 S. E. Thomas, "Value and validity of art zines as an art form," *Art Documentation: Journal of the Art Libraries Society of North America*, vol. 28, no. 2, pp. 27–38, 2009.

25 S. Deterding, D. Dixon, R. Khaled, and L. Nacke, "From game design elements to gamefulness: defining "gamification"," in *Proceedings of the International Academic MindTrek Conference*, 2011.

26 A. Liapis, C. Guckelsberger, J. Zhu, C. Harteveld, S. Kriglstein, A. Denisova, J. Gow, and M. Preuss, "Designing for playfulness in human-AI authoring tools," in *Proceedings of the FDG workshop on Human-AI Interaction Through Play*, 2023.

27 J. Zhu, G. Chanel, M. Cook, A. Denisova, C. Harteveld, and M. Preuss, "Human-AI collaboration through play," in *Human-Game AI Interaction (Dagstuhl Seminar 22251)*, D. Ashlock, S. Maghsudi, D. P. Liebana, P. Spronck, and M. Eberhardinger, Eds. Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022.

28 M. Zammit, I. Voulgari, A. Liapis, and G. N. Yannakakis, "The road to AI literacy education: From pedagogical needs to tangible game design," in *Proceedings of the European Conference on Games Based Learning*, 2021.

29 A. Kultima, K. Alha, and T. Nummenmaa, "Building Finnish game jam community through positive social facilitation," in *Proceedings of the International Academic Mindtrek Conference*, 2016.

3.13 Distance and Density in Various Spaces

Simon M. Lucas (Queen Mary University of London, GB), Duygu Cakmak (Creative Assembly - Horsham, GB), Filippo Carnovalini (VU - Brussels, BE), M Charity (New York University, US), Amy K. Hoover (NJIT - Newark, US), Ahmed Khalifa (University of Malta - Msida, MT), Setareh Maghsudi (Ruhr-Universität Bochum, DE), and Vanessa Volz (CWI - Amsterdam, NL)

Distance and density are fundamental concepts that underpin various critical questions, such as: How novel is a game? Which state should be visited next to explore sparse regions of the space? How similar are two agents' behaviours?

Games naturally deal with many types of space: a single video game frame is a point in image space, a single game state is a point in the game's state space, an agent is a point in strategy space, a game is a point in program space (typically expressed as text in a general purpose high-level language, or in a Game Description Language), and an agent's behaviour in a game can be described as a set of trajectories in (state,action) space.

The purpose of the working group was to explore the underlying theory and practical applications of distance and density in games, and to provide a starting point to game and game AI researchers wishing to explore the fundamentals in more depth.

3.13.1 Background

Given two points x and y in a space S, denote $d(x, y)$ as the distance between x and y. If d satisfies a set of axioms[10] including the triangle inequality, it is standard to call this a *distance metric*. However, there are plenty of measures that do not qualify as a metric, that can still provide useful results in practice.

[10] https://en.wikipedia.org/wiki/Metric_space

3.13.1.1 Density Estimation Methods

We can use a distance measure to directly compute the density at a given point in the space. In practice, these methods involve computing the distance to a number of the closest points sampled from the underlying distribution. Hence, efficient ways to find the closest points become important. A general and widely used approach is to use KD-Trees to find the k nearest neighbors. Given these, we can then use a kernel density estimate, or use the average or maximum distance to the k nearest neighbors (kNN) to estimate density: see Bishop [1] section 2.5.

An alternative to kNN-based methods is to train a model that does not store all the patterns. In the interests of simplicity and brevity, we consider two types of trained models: neural density models, and feature hash-based models.

Neural density models are trained via iterative back propagation, and typically cope well with the curse of dimensionality, at the cost of an iterative tuning and training process. See [2] for recent work on this.

3.13.1.2 Distribution Estimation Methods

Feature hash models involve computing multiple hash indices for each pattern, and counting the occurrences of each index. Hence, the original pattern (e.g. text or image) is transformed into a multinomial distribution.

These methods include n-grams and n-tuples, which are similar concepts used in slightly different contexts: n-grams are commonly used as text probability estimators, usually composed of consecutive symbols (bigrams, trigrams etc), whereas n-tuples have a more general interpretation, e.g. randomly sampling an entire image space.

Related to n-tuple methods are the Context-Tree-Switching (CTS) and Skip-CTS algorithms used by Ostrovski et al [3] their work on unifying count-based exploration and intrinsic motivation. Results included improved performance on Montezuma's revenge (a hard exploration game).

While n-gram models can be used to estimate the probability of a pattern given the model (and by Bayes' theorem, the probability of the model given the pattern), they should be used with care as pattern generators. Maximum likelihood sequences are often highly repetitive; Lucas and Volz [4] overcame this by evolving Mario levels to match the tile-pattern distribution (measured by KL-Divergence) instead of the most likely ones given the model.

3.13.1.3 Invariance and Equivariance

Key concepts related to the notion of distance are *invariance* and *equivariance*. In many applications, we wish to consider similarity between entities independently of certain transformations T.

Invariance: A function f is said to be invariant under transformation T if:

$$f(x) = f(T(x))$$

Example: Consider a convolutional image recognition system that identifies that an image contains Mario, regardless of his location.

In a distance setting, this translates to:

$$d(x, y) = d(x, T(y))$$

Here, the distance remains unchanged even when one of the points is transformed by T.
Equivariance: A function f is said to be equivariant under transformation T if:

$$T(f(x)) = f(T(x))$$

Continuing with the Mario example, an equivariant system identifies Mario and his location. Moving Mario in the image by the transform T also modifies the location output accordingly.

In a distance setting, equivariance means that if we apply a transformation T to both points x and y, the resulting distance should reflect the transformation applied. Formally, for a distance function d and transformation T, this can be expressed as:

$$d(T(x), T(y)) = \text{function of } T \text{ and } d(x, y)$$

For instance, if T is a scaling transformation by a factor c, then an equivariant distance measure should satisfy:

$$d(T(x), T(y)) = c \cdot d(x, y)$$

This reflects the change in distance due to the scaling transformation T.

Applying the appropriate transformations as part of the distance (or density) measurement process is important in many applications to ensure the measurements align with the goals of the application, and are not contaminated with irrelevant variations.

3.13.2 Application: Improved Exploration Policies

RL algorithms typically use random exploration to bootstrap policy learning. In problems with flat reward landscapes, this can be very inefficient. Here we demonstrate an example of improving exploration using notions of distance and density.

Figure18 shows an agent exploring a grid world given a budget of 1,000 steps. The agent's state is defined by its position on the grid, and it is allowed to take its next action (NESW) from any previously visited state. The aim is to visit as many cells as possible within a specified number of steps, in this case 1,000.

Note the effects of three different rollout policies: random, count-based and kNN sparsity based. In each case, having selected a state, we always take a random action from that state. All that differs is the state selection mechanism. In the pure random rollout, we choose the current state. In count-based exploration, we keep a count of the number of times each state has been visited, and select one at random with the least number of visits. In kNN sparsity search we select the state with the lowest kNN density. This is the most sophisticated exploration policy among the three, and has the best performance, demonstrating the importance of a density model that aligns with the goal of the algorithm.

3.13.3 Application: Similarity Estimation in Tile-Based Games

While there are many ways to measure the similarity of content in tile-based games, many practical applications require measures that align with human perception. Berns et al. [5] thoroughly studied various measures and compared their results with a user-study asking participants about how they perceive similarity in different settings 19.

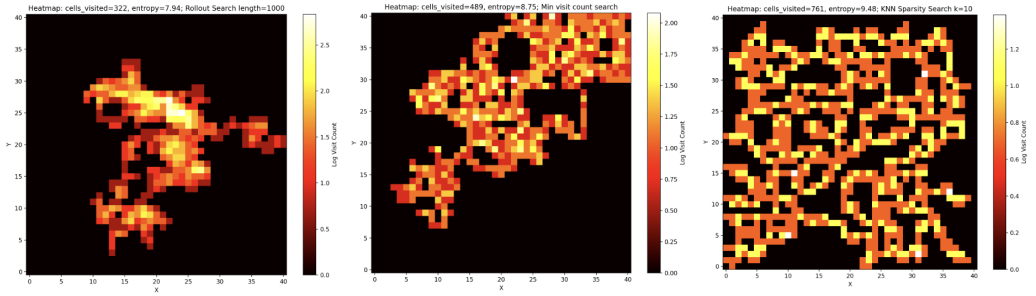
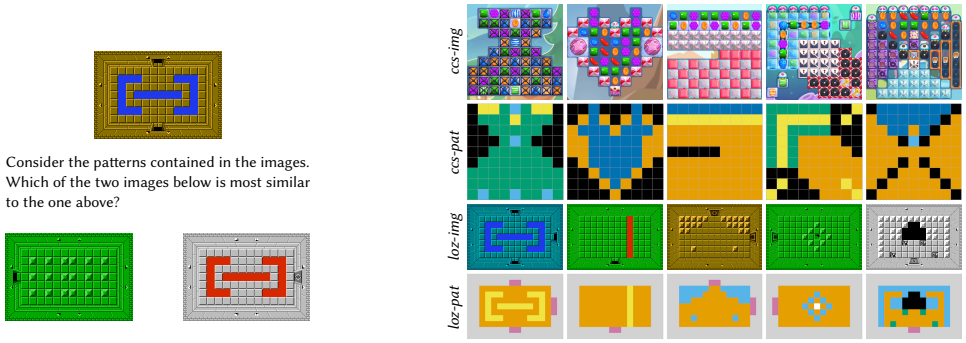


Figure 18 Exploration in a grid world: the aim is to visit as many cells as possible given a 1,000 step budget. Left: random rollout (322 cells): middle: count-based exploration (489 cells); right: kNN sparsity search (761 cells).



Ⓐ Triplet comparison task: two alternative forced choice

Ⓑ Five example stimuli from each experimental condition

Figure 19 (A) Two alternative forced choice (2AFC) triplet comparison task and (B) five example stimuli for each experimental condition: two video game titles (ccs: Candy Crush Saga; loz: Legend of Zelda) in two representations (img: level screenshots; pat: abstract colour patterns). Each stimulus was randomly drawn from the respective subset identified through our three-stage selection procedure

They found that overall, pre-trained general-purpose computer-vision-based models such as DreamSim [8] and CLIP [7] performed better in terms of agreement with human perception than more custom measures developed in the context of PCG. However, they are also more expensive to compute, and this study focused on visual perception only. More work is required in that area to understand the effects of choosing different distance (similarity) measures and how they correspond to human perception, in order to draw more general conclusions.

3.13.4 System Design and Evaluation

In designing any system that uses distance measures and / or density estimation, there are multiple important factors, including:

- Dimensionality and the nature of the native pattern space e.g. vector, sequence, image, data object: affects choice of invariant transformations, distance measures and models.
- Data quantities involved (including volume, velocity, and variability: affects choice of trained versus stored-pattern algorithm.
- Subjective (perceptual) or objective application alignment - see Li et al. [6] for a survey of both types applied to a wide range of games.

- Multimodality of data (e.g. some models may assume a unimodal distribution that does not fit the data well)
- Scaling of distance and density algorithms for commonly used functions, with respect to size of pattern and number of patterns.

3.13.4.1 Evaluation

In general, we find that evaluation of distance and density measures tends to be done ad hoc for each application: we are not aware of benchmarks or evaluation frameworks that can test the performance of these important methods across a range of applications. This seems worth further exploration, given that choice of distance and density measures have on the success of an application — as clearly demonstrated in the two applications we considered.

3.13.4.2 Libraries

- **Density estimation:** https://scikit-learn.org/stable/modules/density.html
- **KD-Trees in Python:** https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.KDTree.html
- **Efficient kNN computation - use in density estimation:** *See the above KD-Trees link*
- **Image-similarity-measures:** https://pypi.org/project/image-similarity-measures/
- **Audio Similarity:** https://librosa.org/doc/latest/index.html
- **Tree Edit Distance:** https://pythonhosted.org/zss/ - useful for computing the distance between two Python programs. To do this, compute the Abstract Syntax Tree for each program using standard functions. Before measuring the tree edit distance, aspects considered irrelevant, such as variable names, may be discarded. See [10] for comparing student program code.

3.13.5 Summary

Distance and density estimation in vector spaces is a well studied topic, with still-useful algorithms such as kNN and kernel density estimation that date back to the 1960s or earlier.

However, we find that there is much to do to apply the most appropriate methods for game-related problems, as illustrated by the example applications discussed above.

There are compromises to be made along the lines of problem alignment, training speed versus inference speed, and dealing with non-vector spaces such as behaviour traces and program code. In many cases, the methods exist for dealing with these as well, but require effort to find and utilise - we recommend more work along these lines. For example, computing distance and density among programs, or among game state objects: we are not aware of any user-ready libraries for these important tasks.

References

1 Bishop, C. M., *Pattern Recognition and Machine Learning. Springer*, 2006. Retrieved from https://www.microsoft.com/en-us/research/uploads/prod/2006/01/Bishop-Pattern-Recognition-and-Machine-Learning-2006.pdf

2 Qiao Liu, Jiaze Xu, Rui Jiang, and Wing Hung Wong, Density estimation using deep generative neural networks.*Proceedings of the National Academy of Sciences*, 118(15):e2101344118, 2021. https://doi.org/10.1073/pnas.2101344118.

3 Ostrovski, G., Bellemare, M. G., van den Oord, A., & Munos, R., *Unifying Count-Based Exploration and Intrinsic Motivation*. arXiv preprint arXiv:1606.01868, 2017. Retrieved from https://arxiv.org/abs/1606.01868

4 Lucas, S. M., & Volz, V., *Tile Pattern KL-Divergence for Analysing and Evolving Game Levels*. In Proceedings of the Genetic and Evolutionary Computation Conference, pp. 170-178, 2019. Retrieved from https://dl.acm.org/doi/10.1145/3321707.3321844
5 Sebastian Berns, Vanessa Volz, Laurissa Tokarchuk, Sam Snodgrass, and Christian Guckelsberger, *Not all the same: Understanding and informing similarity estimation in tile-based video games*. In CHI '24: Proceedings of the CHI Conference on Human Factors in Computing Systems, Article No. 366, pages 1–23, May 2024. https://doi.org/10.1145/3613904.3642077.
6 Yuchen Li, Ziqi Wang, Qingquan Zhang, and Jialin Liu, *Measuring diversity of game scenarios*, 2024. https://arxiv.org/abs/2404.15192.
7 Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever, *Learning Transferable Visual Models From Natural Language Supervision*. In Proceedings of the 38th International Conference on Machine Learning. PMLR, 2021
8 Stephanie Fu, Netanel Tamir, Shobhita Sundaram, Lucy Chai, Richard Zhang, Tali Dekel, and Phillip Isola, DreamSim: Learning New Dimensions of Human Visual Similarity using Synthetic Data. In Advances in Neural Information Processing Systems, Vol. 36, 2023.
9 Zhang, K., & Shasha., Simple fast algorithms for the editing distance between trees and related problems. *SIAM Journal on Computing*, 18(6), 1245-1262, 1989.
10 Chou, E., Fossati, D., & Hershkovitz, A., A Code Distance Approach to Measure Originality in Computer Programming. In *Proceedings of the 16th International Conference on Computer Supported Education (CSEDU 2024)* - Volume 2 (pp. 541-548). DOI: 10, 2024.

3.14 Sub-optimal Bots

David Melhart (University of Malta - Msida, MT), James Goodman (Queen Mary University of London, GB), Christian Guckelsberger (Aalto University, FI), Greta Hoffmann (TH Köln, DE), and Diego Perez Liebana (Queen Mary University of London, GB)

We traditionally aim to create game-playing agents that are optimised for a certain task — be it performance, human-like behaviour, or facilitating a positive player experience. In this working group, we asked the question — *"Do bots always have to be efficient?"* We argue that there is merit to low-fi, sub-optimal bots in areas such as games testing, producing erratic and distracted behaviour to model users, and resource management for sustainability.

3.14.1 The Scope of Being Sub-optimal

When talking about sub-optimal bots, we focus on agents that are designed to be "bad at their job" — that is they cannot perform as well as the state-of-the-art due to some constraints. While *constraints* here generally mean resource constraints in terms of limited computing power or memory or time, sub-optimal bots could also under-perform current systems by presenting deliberately non-human-like. The former we could call *natural constraints*, and the latter we could call *artificial constraints*.

- **Natural Constraints:** The natural constraints of a computational system constitute its computational resources. Sub-optimal performance could arise from simple limits on resource use. While some of these limits are a given, others can exist by design. In the

current boom of large models, we often view resource constraints as a limit we should aim to lift, and with a good reason. Since 2017 we have seen the exponential rise of Large Language Models (LLMs) [1] fuelled both by innovation [12] and by ever-growing architecture size, training data volume, and computational resources [6].

- **Artificial Constraints:** In light of the aforementioned positive aspects of scaling, why would we want to limit computational resources by design? We identify two main reasons:
  - **A case for Sustainability:** One of these main reasons is sustainability. Water usage of the vendors of foundation models is on the rise [3], and despite improvements in energy efficiency, the growing penetration of AI in different domains paints an uncertain picture of our future energy needs [2]. While we need powerful models to drive innovation and tackle complex problems, we need to think about the potential environmental cost of our applications as well. Up until now, the "state of the art" meant "best performance" — however, we propose a new challenge for future applications and innovations in the space that focus on "good enough performance" and "best resource savings". Future research in this area could be facilitated via a new competition for *Sustainable Agents*.
  - **By Design:** Artificial limitations can be designed to produce real or perceived sub-optimal performance — not necessarily by limiting actual resources. Our two grand examples here are *distracted bots* and *non-human-like agents*. In the case of distracted bots, we envision systems mirroring human cognitive load [5] or short-term memory [9] to create more anthropomorphic behaviour by modelling the human attention span. We hypothesise that bots limited in this way would produce unexpected but more human-like behaviour. In the case of non-human-like agents, the goal of imposed limitations is the opposite of achieving believability. These bots would be designed to emphasise their artificial nature and the limited capacity of their underlying systems. A positive outcome of these agents would be algorithmic transparency [13] and emotional separation between humans and machines, which could cause unexpected negative outcomes (e.g. trauma dumping or developing unhealthy attachments to AI companions).

3.14.2 Applications for Sub-optimal Bots

In practical terms, there are a number of ways we can limit our bots to enforce a sub-optimal performance.

- **Memory:** Many agents have an implicit perfect memory, which is not reflective of (most) human players. Restricting the memory window, or the types of historic events remembered would give sub-optimal, and arguably much more human-like behaviour.
- **Forward Planning Horizon and Discount Rate:** A shorter planning horizon, and/or a discount rate profile that emphasises short-term gains would force sub-optimal myopic agents to sacrifice long-term success for immediate gratification.
- **Flexibility of Play Style and Strategies:** Restricting the available action space for an agent could force it to explore different strategies. For example, if a Chess agent were forbidden from moving its Queen until after it had castled.
- **Accuracy of World Model:** Providing planning agents with deliberately imperfect models of the world. For example in a first-person shooter game, an agent might overestimate how far it can jump; or have an incorrect mental map of unseen terrain.
- **Perception of World State:** A restrictive field of view, only taking input observations from a subset of the full data available.
- **Accuracy of Action Implementation:** Just because a specific action is intended does not mean it is implemented in the game environment accurately. This is used for example

in first-person shooters to ensure that NPC bots often miss[11], but can be extended more widely to mirror human mis-clicks on interfaces.
- **Theory of Mind:** Modelling Theory of Mind explicitly in terms of, 'I know that she knows that I know...' may give interestingly sub-optimal behaviour. This may also be a good reflection of non-expert humans who frequently only manage one to two nested levels of theory of mind [9].

We have identified a number of domains within game research, serious games, and games entertainment where these aforementioned limitations — as described above — could find a positive use.

- **Games Testing:** In Games Testing it is often desirable to have agents exploring the game in a non-optimal fashion. This is because we aim to increase the test coverage by expanding into non-optimal play or simulating human behaviour by mirroring the limitations of human cognition.
- **Games Playing:** Similarly to Games Testing, Non-Player Characters can also benefit from the aforementioned limitations by enhancing their human-like aspects and/or creating sub-optimal opponents. This could help provide a variety of interestingly sub-optimal opponents. These opponents could be used as scripted adversaries, tutorials to multiplayer games or as part of a dynamic difficulty adjustment system.
- **User Modelling:** Beyond testing, user research can make use of models which simulate human limitations to understand and improve Human-Computer Systems.
- **Resource Optimisation:** Finally, resource limits could be used simply to cut down on practical resource usage. The implication here is that bots and agents trained on such limitations would be "optimised" to handle constraints.

3.14.3 Future Research

We have identified several possible research questions for future studies into applying low-fidelity bots to simulate human-like behaviour.

- **How do we perceive bots being bad at their jobs?** Previous research showed that bots are designed to exhibit certain cognitive processes often perceived as confusing [8]. Future research should focus on uncovering reliable markers of bad performance due to cognitive limitations.
- **How could bots resist humans?** Similarly, resistance from interactive agents could be interpreted as a sign of an unresponsive system, rather than intentional feedback. Current human-like systems are often ill-equipped to communicate their own limitations — the most famous example being ChatGPT's tendency to "yes and" requests even if a request is beyond its limitations. Future work into HCI should aim to understand the key to clear communication and reduce user frustration in cases where the bots are limited in some capacity.
- **Do constraints make bots more (or less) human-like?** At the beginning of the report, when describing cases for imposing *Artificial Constraints* by design, we hypothesised two possible directions for limited agents. One was enhancing the human-like nature of the bot, while the other was specifically stripping the agent of believability. Future research in this area should focus on the spectrum of non-human to human-like to super-human bots in terms of perceived performance and believability.

- **A case for Computational Rationality:** Beyond the domain of games, we believe that the research of sub-optimal bots could shed light on emergent qualities of human cognition and behaviour. Based on the Theory of Computational Rationality [7, 4, 10] human cognition can be conceptualised as an "optimal" — that is rational — system under computational constraints. Given this premise, future research could aim to go beyond the scope of game-playing agents when modelling human cognition and ultimately behaviour via limited bots.

3.14.4 Conclusions

In this report, we detailed some of the outcomes of our work group on sub-optimal bots. We aimed to map out the scope of limitations, applications, and future research, arguing for the benefits of researching and developing underperforming computer agents. We believe in areas such as games testing, user modelling, and resource management for sustainability these bots can contribute to positive outcomes.

References

1 Chang, Y., Wang, X., Wang, J., Wu, Y., Yang, L., Zhu, K., Chen, H., Yi, X., Wang, C., Wang, Y. and Ye, W., *A survey on evaluation of large language models*. In ACM Transactions on Intelligent Systems and Technology, 15(3), pp.1-45, 2024.

2 Desislavov, R., Martínez-Plumed, F. and Hernández-Orallo, J., *Trends in AI inference energy consumption: Beyond the performance-vs-parameter laws of deep learning*. In Sustainable Computing: Informatics and Systems, 38, p.100857, 2023.

3 George, A.S., George, A.H. and Martin, A.G., *The environmental impact of AI: a case study of water consumption by chat GPT*. In Partners Universal International Innovation Journal, 1(2), pp.97-104, 2023.

4 Gershman, S.J., Horvitz, E.J. and Tenenbaum, J.B., *Computational Rationality: A Converging Paradigm for Intelligence in Brains, Minds, and Machines*. In Science, 349(6245), pp. 273-278, 2015.

5 Hedden, T. and Zhang J., *What Do You Think I Think You Think?: Strategic Reasoning in Matrix Games*. In Cognition 85(1), pp. 1–36, 2002.

6 Kaplan, J., McCandlish, S., Henighan, T., Brown, T.B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J. and Amodei, D., *Scaling laws for neural language models*. arXiv preprint arXiv:2001.08361, 2020.

7 Lewis, R.L., Howes, A. and Singh, S., *Computational Rationality: Linking Mechanism and Behavior Through Bounded Utility Maximization*. In Topics in Cognitive Science, 6(2), pp. 279-311, 2014.

8 Melhart, D., Yannakakis, G.N. and Liapis, A., *I Feel I Feel You: A Theory of Mind Experiment in Games*. In KI-Künstliche Intelligenz, 34(1), pp.45-55, 2020.

9 Miller, E. K., and Buschman, T. J., *Working Memory Capacity: Limits on the Bandwidth of Cognition*. Daedalus 144(1), pp. 112–122, 2015.

10 Oulasvirta, A., Jokinen, J.P. and Howes, A., *Computational Rationality as a Theory of Interaction*. In Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems. pp. 1-14, 2022.

11 Schrum, J., Karpov, I. V., and Miikkulainen, R. *Human-like Combat Behaviour via Multi-objective Neuroevolution*. In Believable Bots, 119–50. Springer, 2013.

12 Stiennon, N., Ouyang, L., Wu, J., Ziegler, D., Lowe, R., Voss, C., Radford, A., Amodei, D. and Christiano, P.F., *Learning to summarize with human feedback*. In Advances in Neural Information Processing Systems, 33, pp.3008-3021, 2020.

13 Watson, H.J. and Nations, C., *Addressing the growing need for algorithmic transparency*. In Communications of the Association for Information Systems, 45(1), p.26, 2019.

3.15 Arts & Crafts & Generative AI

Mirjam Palosaari Eladhari (Stockholm University, SE), Gabriella A. B. Barros (modl.ai - Maceio, BR), Alena Denisova (University of York, GB), Amy K. Hoover (NJIT - Newark, US), Chengpeng Hu (Southern Univ. of Science and Technology - Shenzen, CN), Leonie Kallabis (TH Köln, DE), Ahmed Khalifa (University of Malta - Msida, MT), Matthias Müller-Brockhausen (Leiden University, NL), Gillian Smith (Worcester Polytechnic Institute, US), and Anne Sullivan (Georgia Institute of Technology - Atlanta, US)

Generative Artificial Intelligence (GenAI) holds much potential for artistic practice in game design, craft, and fine arts. At the time of writing, 2024, the advancements of GenAI are rapid in regards to Large Language Models (LLMs) and Text to Image (TtI) systems. Artists and game designers are keen to explore new the tools, systems, and their updates as they are released, and the pace is quick. The members of the work group have multiple roles as scientists, game designers, and art- and craft practitioners, and converged around exploring GenAI for arts, crafts, and game development.

In this report we describe our visions of usage of GenAI for creation of game assets, free form doodles, and art generated using exclusively individual artists' original art in "small" data sets.

3.15.1 Identification of Open Questions for Exploration

The group started with discussing themes to explore, and then identified topics for subgroups for further investigation. The themes related to ways to use GenAI when both artists and scientists use generative AI in collaboration with the GenAI systems. They were as follow:

- AI as an art material: akin to a painting brush or a specific software tool.
- Small Datasets: The affordances the use of small datasets may offer individual artists and teams of creators for expressing their individual artistic voices and for ascertaining that any art work used for generation originates from the artists themselves, rather than originating from an large database of data scraped from the web.
- The human wanting to do the fun stuff: The tension between what tasks an artist may want to do themselves, as a part of their creative practice, and between tasks a generative AI system may be capable of.
- Mixed initiative creation: The use of technologies of content generation as part of an artistic process.
- Software methods and tools using generative AI for supporting creativity.
- Simultaneous generation of game levels and assets.
- Stylistic coherence of generated output.
- Skill enhancement in visual representation such as drawing, allowing people who are not trained in visual expression to create artifacts that presents ideas in a more specific and sophisticated manner.
- Usage of generative AI in own artistic practice.
- Creativity support tools for art.

3.15.2 Three sub groups

We selected three themes for further inquiry in sub groups:

- Small datasets for designers and artists. (Subgroup 1)
  - Creative and artistic use of small datasets restricted to artists' and designers' own artifacts.
  - Usage of tools in practice of design and art.
  - Surveys of available tools, their affordances and practical guides.
- Game asset generation (Subgroup 2)
  - stylistic consistency
  - design principles
  - resonance between level design and assets for specific level.
- Doodling: Design affordances of Large Language Models (LLMs) in relation to Text to Image (TtL) systems. (Subgroup 3)
  - Use of LLMs and TtLs as art materials.
  - What is the equivalence of doodling using LLMs and TtLS?

In the following we summarise the outcomes of subgroups 1 and 2, referring to the respective reports, which is followed by the report of subgroup 3.

3.15.3 Small datasets for designers and artists – Subgroup 1

The subgroup focused on how artists and designers can use their own artwork to train GenAI models. When utilizing large language models (LLMs) trained on extensive databases, often sourced from the internet, issues concerning artists' contributions arise, particularly around individual style and intellectual property. The use of "small" datasets, as discussed by [8], offers promising opportunities for artists, game designers, and teams working within specific collaborations or development studios. If a generative model is trained solely on an artist's own creations, it can produce outputs that are unique to that artist while still harnessing the capabilities of GenAI technologies.

Our group explored the concept of GenAI as a tool within an artist's studiocomparable to a new brush or other medium for creative expression. We conducted a survey with three primary focuses: the historical use of generative arts by visual artists and authors, dating back to the 1920s; the technologies employed in tools that allow artists to train models on their own work; and the currently available tools for such purposes.

In our investigation, we found that while many tools and high-level surveys exist, there is a lack of practical guidance to help artists make informed decisions about which tools to use and how to contextualize them. From the range of available tools, we conducted experiments. For example, we trained a model exclusively on paintings from one of the group's artists.

Throughout this experimentation, we identified the significant potential of using limited datasets for training. However, the process remains labor-intensive. We recommend further work in this area, including the development of guides and standards for artists, designers, and developers. These resources should offer guidance on how to use current and future technologies, the types of creative support they provide, their affordances regarding the nature of generated works, and their accessibility in terms of skill requirements, cost, and whether they are open source. The subgroup's work is described in more detail in 3.16.

3.15.4 Generation of Game Assets – Subgroup 2

The group exploring the generation of game assets concentrated on creating graphical assets, commonly referred to as sprites. Their work was guided by two key questions: a) How can the generation of game assets take into account the properties of different game types? and b) How can we generate multiple game assets that maintain a consistent art style?

To address these questions, the group followed the guidelines set out by Liu et al. [1], which emphasize focusing on subject and style keywords that can be adapted to specific styles and levels of abstraction. They categorized the assets into different groups, such as characters, interactive objects, environments, and more, in order to generate cohesive sprite sets.

The group experimented with various diffusion models to assess their effectiveness in mimicking distinct art styles, as detailed in section 3.11. Their findings indicated that while stable diffusion models can produce reasonably comparable images, achieving a high level of stylistic consistency that is convincing to viewers remains a significant challenge. The subgroup's work is described in more detail in 3.11.

3.15.5 Doodling with Large Language Models and Text to Image Systems – Subgroup 3

In this subgroup, run by *Gillian Smith, Anne Sullivan, and Alena Denisova*, we discussed and experimented with the playful act of "doodling" with off-the-shelf generative systems. Since doodling is something that artists, crafters, and designers can commonly do with any kind of design material, we posit: **if we want to understand Generative AI as a design material, we need to learn how to doodle with it**.

3.15.6 Intro

We began our exploration with a discussion of what doodling is and its role in the creative process, drawing from our own experiences. We identified seven characteristics of doodling:

- Improvisational: doodling is a form of improvisational play, where the artist is building upon their own prior work over time, exploring and reforming their creative process as they go.
- Not goal-oriented: doodling is an exploratory activity. Artists doodle without intent for these doodles to be shared or incorporated into larger planned works.
- Playful: doodling permits playful exploration of materials, experimenting with whimsical or unexpected forms, and may be done primarily for fun or leisure.
- Low cost: artists doodle with existing, low cost, materials that are not reserved for another project or too precious to 'waste'.
- No notion of failure: doodling is heavily process-oriented, with no real notion of 'failure' with respect to either internal or external validation or judgment. It transcends skill and judgment; it is accessible to everyone, and there is no way to do it either wrong or right.
- Autotelic: doodling serves its own purpose for an artist; it is not done in the service of any larger intended goal, even if it does emergently lead to new skills or techniques.
- Easy to pick up and put down: doodling is often done idly, by the subconscious mind. It is done to help focus the mind, to visualise ideas and help yourself think, or to pass time. As such, doodling is engage intermittently – easy to start, stop, and start again.

We further identified common roles that doodling plays for an artist's creative process. When doodling, an artist can:

- Learn and explore the material affordances of the medium.
- Discover and build skills with the medium.
- Reflect on their own creative process.
- Reflect on their own identity and goals as an artist.
- Focus their mind through engagement in a continued repetitive task.
- Explore a new and unfamiliar design space.

Building from these discussions of what it means to doodle *in general*, we focused the remainder of our subgroup on exploring what it means to doodle *with generative AI systems*. We chose ChatGPT 4.0 as a system that was 'easy to pick up and put down', and aimed to focus on the generative model as opposed to the outputs of the system. That is: the goal is not to generate doodles, but to doodle with a prompt-based interaction mechanism for a generative model.

3.15.7 Summary of Experiments

To begin our experimentation, we gave ourselves one rule: *Just sit down, open ChatGPT, and play.*

Throughout the experience, we would share our output and our prompts, borrowing ideas and inspiration from each other to modify and work from, almost akin to musical riffing and improvisational theater.

Some of the prompt experiments that we ended up using are as follows:

- Making up fictional art styles and applying to existing images
- Making up single-word responses for long complex inputs
- Drawing pictures of those single-word responses
- Re-imagining names as mythical creatures
- Make ASCII art from emoji
- Respond to sentences with emoji chains

Through the playful experimentation, we had a few insights into what worked and didn't work when doodling with ChatGPT 4.

The relative ease of getting output that was "good enough" fit well with a doodling mindset and the unexpected results made the overall experience quite fun. However, the high fidelity of the output worked against it - it was easy to get caught up in the details. Additionally, the higher fidelity took more time to render, which also moves it away from feeling like a doodle, as doodles are often about fast results.

What does this mean for LLMs as media? The question opens up a fascinating exploration of the ways in which large language models (LLMs) like ChatGPT are more than just tools for replicating existing art forms. They are not simply systems that mimic or reproduce the traditional structures of media, whether in writing, visual arts, or conversation. The process and input into these models are unique in their nature and do not conform to the established categories of creation.

In this sense, the output generated by these models is not bound by the conventions of fidelity we might apply to other forms of media. For instance, what we might traditionally consider 'high fidelity' images—detailed, precise, and closely mirroring reality – are now seen through a different lens. The outputs, which may sometimes resemble something more akin to doodles or abstract sketches, take on a new meaning and value within this framework. The results need not be perfect replicas to hold artistic merit or expressive power.

What is exciting about this is the fun involved in the process of creating with LLMs. There is an inherent playfulness and experimentation that invites users to explore the expressive range of the underlying models. The user is not just interacting with a tool but is engaged in the development of a personal understanding of the model's potential and limitations. This engagement fosters an evolving relationship with the medium itself, and this kind of active discovery is promising for the future of AI-assisted creativity.

Moreover, through this process, users begin to develop their own artistic vocabulary specifically tailored to using these models. Just as traditional artists use a distinct set

Figure 20 Here's the illustration featuring a glimberflopped in a nerdletower located in lichtwald forest filled with geekfrolics in the style of bleakoscribble.

of skills and tools to create within their chosen medium—whether painting, photography, or sculpture—working with LLMs requires the development of a new set of skills and a personalised approach. This vocabulary allows users to unlock deeper forms of expression and pushes the boundaries of what LLMs can achieve in artistic creation. It shows promise for AI and human creativity becoming more intertwined, each informing and expanding the potential of the other.

3.15.8 Takeaways

Doodling is an important part of a creative practice. Therefore, for LLMs and AI-based tools to excel as creative media, they need to be designed to support doodling-style interactions. Based on our experimentation during this workshop, we identified a few ways in which LLM-based tools could enable doodling like behavior.

The primary change is that LLM-based creative tools should support casual, low cost, idle engagement. ChatGPT 4, used for our experiments, has some capability for this, but we struggled with the high fidelity output which took quite a bit of time to generate. If instead, it was possible to enter into a sketch or doodle mode where low-fidelity images could be more quickly generated for refinement later, this would allow for a more doodle-like experience.

Beyond this, the chat-based and browser-like interface creates a distance between the interactor and the output, which we struggled with. For example, if there was one part of an

Figure 21 Here's the digital painting of featuring a glimberflopped in a nerdletower located in lichtwald forest filled with geekfrolics in the style of joyspright.

image that we wanted to change, we were unable to convey this information through the interface we had access to. Additionally, having a series of images to chose from is a form of curation rather than of creation. Addressing this gap is vital for making LLM-based tools align with creative practice.

Our doodling-inspired experiments led us to realize that we need new vocabularies for AI Art. Currently, the vocabulary is focused on tool support and replacement for traditional art. However, this language does not support an authentic engagement within the medium, which is required for AI Art to be perceived and used as an art practice.

Finally, for AI Art and LLM-based tools to excel as a creative medium, there must be continued work considering and addressing the ethics of the data and use of these tools. Until this is addressed, these tools will never be recognized as a creative medium by art communities, due to the questionable practices by these companies.

References

1 Vivian Liu and Lydia B Chilton. *Design guidelines for prompt engineering text-to-image generative models.* In Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems, CHI 22, New York, NY, USA, 2022. Association for Computing Machinery.

2 Gabriel Vigliensoni, Phoenix Perry, Rebecca Fiebrink, and others. *A Small-Data Mindset for Generative AI Creative Work.* Generative AI and HCI - CHI 2022 Workshop, 2022.

3.16 Small Data Sets for Designers and Artists

Mirjam Palosaari Eladhari (Stockholm University, SE), Gabriella A. B. Barros (modl.ai - Maceio, BR), Amy K. Hoover (NJIT - Newark, US), and Ahmed Khalifa (University of Malta - Msida, MT)

Generative AI has emerged as a transformative tool in the fields of art and design, offering new ways to create and explore visual, auditory, and interactive works. The use of generative AI often relies on large datasets and advanced technical knowledge, which can pose challenges for artists and designers in therms of access and resources. Another issue is that large language models are trained on data from external datasets, which is an issue for artists when it comes to the expression of their artistic authorial voice as well as concern in regards to intellectual property.

However, there is growing interest in the use of small data sets and more accessible tools to empower artists and designers, enabling them to maintain their artistic voice, address intellectual property concerns, and make generative AI techniques more inclusive and widely available.

This report explores the insights gained from a subgroup in our workshop on Generative AI for art and craft, centering on "Small Data Sets for Designers and Artists," focusing on the application of generative AI in creative practices. The workshop raised key questions, such as: What generative AI techniques are currently being used by artists and designers? What tools exist to support the creation of AI-generated art, and how accessible are they? Additionally, the discussion explored how small amounts of data can be used to train or fine-tune generative models, and what level of knowledge is required to effectively use these tools. A critical focus was on how generative AI can be integrated into the creative workflows of artists, designers, and programmers, and how these tools and techniques can be considered as artistic materials in their own right.

3.16.1 Background

To contextualize our discussion, we examined the use of generative techniques in creative work throughout recent history. This exploration included the surrealist practices of the 1920s [3] as well as experiments with physical materials such as Max Ernst's grattage technique from the 1930s. We also considered the concept of "potential literature" pioneered by the Oulipo group (Ouvroir de littérature potentielle) in the 1960s [2]. The use of computation in the visual arts can be traced back to the 1960s, when artists like Georg Nees and Vera Molnar experimented with flatbed drawing machines and plotters. In recent years, generative AI has been prominently featured in the works of artists such as Anna Ridler, Mario Klingemann, Nettrice Gaskins, Refik Anadol, and Helena Sarin. Notably, Ridler [6] and Sarin's [7] practices involve training models using imagery they have personally created, blending their artistic vision with customized adaptations of computational methods to produce unique works that reflect their individual styles.

Vigliensoni et al. [8] argue that working with small-scale datasets is a powerful way to enable greater human influence over generative AI systems in creative contexts. They suggest that while large datasets may lead to overfitting and may not align with specific creative goals, models built with smaller, more focused datasets can better support meaningful and personalized creative work.

3.16.2 Positioning and Resources

In our subgroup, we aimed to explore how artists and designers could leverage their own art in conjunction with generative AI techniques. During the workshop, we focused on utilizing existing artworks created by participants and experimenting with the available tools in July 2024. We identified a variety of resources to support our exploration, including APIs, tutorials, and custom-built tools.[11]

Several surveys, such as that by Franceschelli [4], have explored how generative AI can be applied in creative practices. Additionally, symposia like Creative Machine, held at Oxford since 2014[12], bring together artists and researchers to showcase their work at the intersection of creativity and AI. The report *AI and the Arts: How Machine Learning is Changing Creative Work* [5] presents a comprehensive overview of the creative communities interested in using machine learning in the arts, as shown in Figure 22. Within our group, we identified our position at the intersection of the "Technical Community" and "The Art World," reflecting our engagement with both the technical and artistic dimensions of generative AI.

3.16.3 Experiment

As we explored the available tools for using personal visual art as small datasets to train generative AI models, we conducted quick tests with free or demo versions. Our guiding question for these tests was: "As an artist, how can I experiment with using my own creations, such as drawings or paintings, to train a model and explore its potential in my artistic practice?"

We quickly realized that navigating the tools, understanding their computational foundations, and assessing their affordances was far from straightforward. For a designer or artist, a significant amount of time would be required to explore the possibilities, often through trial and error. The complexity of the tools and their steep learning curve presented a barrier to immediate creative experimentation.

---

[11] Available resources we examined are listed blow. It is not a comprehensive list of all available tools, only the ones we examined in the workshop.

- ML5js https://ml5js.org/
- OpenArt https://openart.ai/
- Craiyon https://www.craiyon.com/
- Runway https://runwayml.com/
- Leonardo https://leonardo.ai/
- Vizcom (for product design) https://www.vizcom.ai/
- Canva https://www.canva.com/ai-image-generator/
- DreamBooth https://dreamlook.ai/dreambooth
- Tutorial for custom datasets at Hugging Face https://huggingface.co (available with search term: transformers v3.2.0)
- Figment https://figmentapp.com/
- Training Loras (available as a google colab)
- Generative Deep Learning (available at github)
- Suite of tools for course on Artificial Intelligence and Storytelling given at The University of Edinburgh, https://kage.dev/ai-storytelling-backstage/

[12] https://www.creativemachine.io

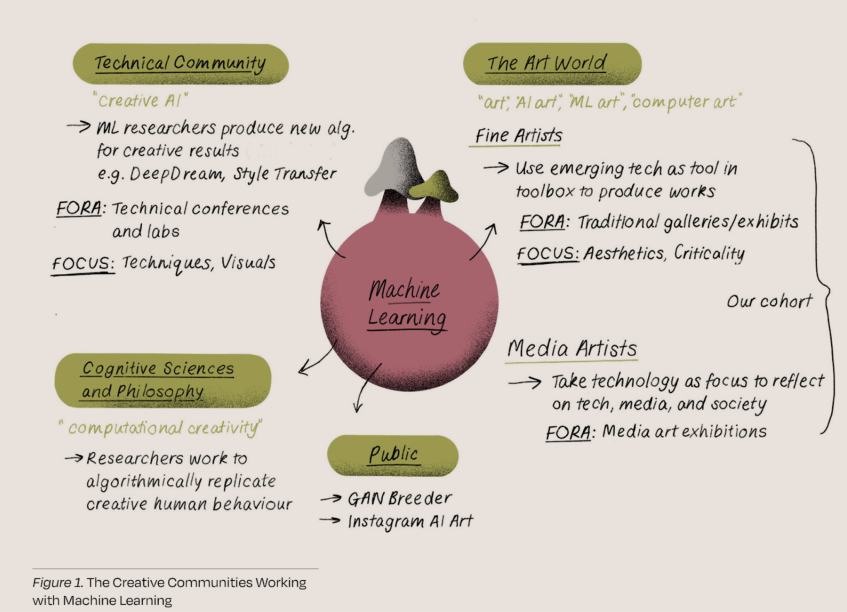


Figure 22 Creative communities working with machine learning.

Furthermore, we found that existing surveys and general guides either lacked practical advice for getting started or were so tool-specific that they failed to provide a broader understanding of the generative AI landscape. This made it difficult for a reader to grasp the larger context and application of these tools in creative work.

For our experiment, we selected two sources of data: The Stanford Dogs Dataset [13] and two paintings created by one of the group members (M.P. Eladhari). During the workshop, we tested several of the tools listed in Footnote 1. However, due to the limited time available and the time-consuming nature of model training, we were unable to fully develop the results of our trials. Nevertheless, in Fig 23 we present an example generated using a tool developed at the University of Edinburgh, based on Stable Diffusion and GPT-2 models. The tool, along with its code repositories, is accessible online [1]. For this test, we used two paintings (shown on the left in Fig 23 as input, along with their titles as text prompts: "I Knew I Would Find You" and "Toad in Still Water". The generated image, shown on the right, provides an example of the creative potential offered by these AI tools.

3.16.4 Immediate Needs and Outlook

In our group, we identified the need to conduct a comprehensive survey of available generative AI tools that can be used to train models on individual or collective artistic works. This survey would assess these tools based on the following criteria:

- Creativity Support: How the tools can be integrated into and enhance the creative process.
- Affordances: The types of artifacts the tools can generate and the creative possibilities they offer.

[13] http://vision.stanford.edu/aditya86/ImageNetDogs/

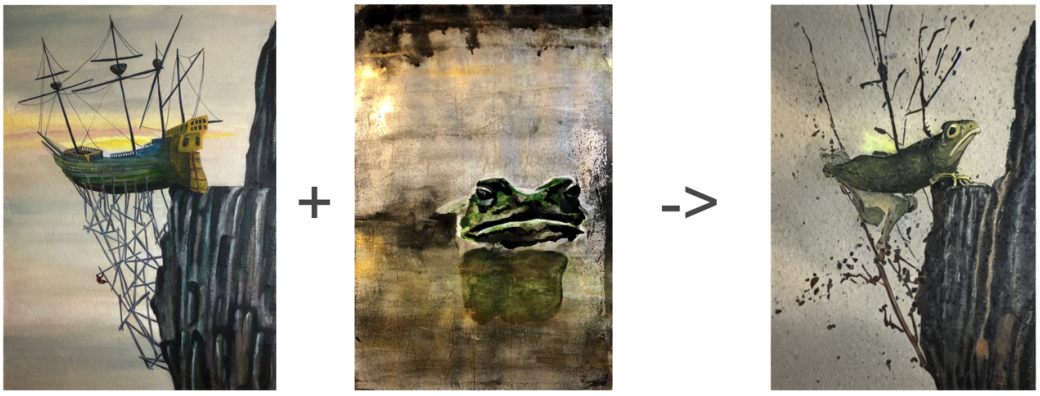


Figure 23 Illustration of combining two paintings by the same artist.

- Accessibility: How easy the tools are to use, considering both technical complexity and financial cost.

Such a mapping would provide valuable guidance to artists and designers by helping to:

- Identify tools that complement or even expand their individual artistic practices;
- Produce artifacts aligned with their artistic vision; and
- Evaluate whether they possess the necessary skills and resources to effectively use a particular tool.

Looking ahead, potential future collaborations within our subgroup include the creation of:

- A practical guide tailored for artists and designers.
- A practical guide bridging computer scientists and artists.
- Specifications outlining the minimal data requirements for training or fine-tuning various generative models.

3.16.5 Summary

In our subgroup report Small Data Sets for Designers and Artists, we explored how generative AI can be integrated into creative practices, focusing on the challenges and possibilities of using small datasets to train models for individual artistic work. We highlighted the importance of accessible tools and practical guides to help artists like us to maintain control over our creative visions while addressing issues like intellectual property.

References

1 Pavlos Andreadis, Kage, P. *Artificial Intelligence and Storytelling*. Course of The University of Edinburgh, https://kage.dev/ai-storytelling-backstage/, 2022.
2 Alastair Brotchie, Le Lionnais, F., Mathews, H. *Oulipo laboratory: texts from the "Bibliothèque oulipienne"*. Atlas anti-classics, Atlas press, London, 1995.
3 Alastair Brotchie, Gooding, M. (eds.) *A book of surrealist games*. Shambhala Redstone Editions, Boston, 1995.
4 Giorigio Franceschelli, Musolesi, M. *Creativity and Machine Learning: A Survey*. https://doi.org/10.48550/ARXIV.2104.02726, 2021.

5 Anne Ploin, Eynin,R., Hjort, I., Osborne, M. (2022). *AI and the Arts: How Machine Learning is Changing Creative Work*. Report from the Creative Algorithmic Intelligence Research Project. Oxford Internet Institute, University of Oxford, 2022.
6 Anna Ridler. *Repeating and remembering: the associations of GANs in an art context.*. In: The 31st International Conference on Neural Information Processing Systems.New York: Curran Associates Inc. 2017.
7 Helena Sarin. *Why Bigger Isn't Always Better With GANs And AI Art*, https://www.artnome.com/news/2018/11/14/helena-sarin-why-bigger-isnt- always-better-with-gans-and-ai-art, visited 2024-06-20, 2018.
8 Gabriel Vigliensoni, Phoenix Perry, Rebecca Fiebrink, and others. *A Small-Data Mindset for Generative AI Creative Work*. Generative AI and HCI - CHI 2022 Workshop, 2022.

3.17 Evaluating the Generative Space of Procedural Narrative Generators

Emily Short (Oxford, GB), Gabriella A. B. Barros (modl.ai - Maceio, BR), Alex J. Champandard (creative.ai - Wien, AT), Michael Cook (King's College London, GB), João Miguel Cunha (University of Coimbra, PT), Alena Denisova (University of York, GB), Antonios Liapis (University of Malta - Msida, MT), Mirjam Palosaari Eladhari (Stockholm University, SE), Johanna Pirker (LMU München, DE), Gillian Smith (Worcester Polytechnic Institute, US), Anne Sullivan (Georgia Institute of Technology - Atlanta, US), and Tommy Thompson (AI and Games - London, GB)

This group discussed the problem of evaluating procedural narrative generators, as a subcategory of procedural generation tools. While there is existing scholarship around how to categorize procedural generation tools for level design (for instance), there is little that aims to assess the quality and variety of procedural narrative generation.

This is an area of potential value to the game industry, as well as a matter of academic interest. There is commercial appetite for games that can present players with a wide variety of novel and engaging stories, but building effective procedural narrative systems is unpredictable and bug-prone, and the difficulty of assessing their quality during production causes many such prototype systems to be removed from industry projects prior to launch. Evaluative frameworks that would allow studios to build and validate their innovation more confidently would help to alleviate these problems.

In discussing this framework, the group wanted to focus on the narrative qualities of generated output rather than to restrict their analysis to any specific type of game (or even to insist that the objects analyzed be games at all). Procedural narrative generators might include games that produce a narrative in the course of play via simulation or other means (*Dwarf Fortress*, *Façade*, *Versu*) as well as non-interactive story generators or story-sifting approaches. They might also be hand-authored or built on a deep learning model such as an LLM. For the purposes of this discussion, we were interested in considering the generative space of such generators regardless of their use case or implementation.

We next considered some possible evaluative and descriptive criteria for generated output. Evaluative measures were meant to be those that could be directly used to assess the quality

of the output; tendency towards repetition; novelty or surprise in generated outputs; measures of creativity (always challenging to identify); and analyses of how players specifically in interactive generators chose to interact (e.g. whether there are some choices that are never selected, or choice points at which the player often paused for thought). Descriptive measures were those that simply attempted to identify important aspects of the generated game, such as structures that emerged in player choices or in graphs of automated testing.

The group here recognized that there were many possible users of this kind of evaluative framework (authors, analysts, narrative designers or product managers in a game studio, or the players themselves); many possible data sources that might be the subject of analysis (the static content modules of a game, traces of real playthroughs, traces of automated playthroughs, qualitative feedback, or even complexity analysis of structural change over time); and many objectives for its use (assuring quality, making sure that the game generated stories that were aligned with aesthetic intent, or identifying changes in a generator over time).

The final phase of the discussion focused on attempting a specific approach: a framework for looking at playable interactive games that were tested by an auto-playing bot or by human players, and seeking to identify patterns such as building (or receding) success of the protagonist in their goals, as well as important choices or reversals. Here it was suggested that tracking key variables over time (such as the relationship with another character, or the player's health) would produce curves that correlate to perceived plots: for instance, a plot in which the player never got on with their grumpy neighbour would appear distinct from one in which they were friends from the start – or from one in which they became close only at the end of play.

This idea was pursued in a second working group on the subsequent day.

References

1 Smith, Gillian Whitehead, Jim. *Analyzing the expressive range of a level generator*, 2010. 10.1145/1814256.1814260.

3.18 Generative Space Analysis for Procedural Narrative Generation

Emily Short (Oxford, GB), Gabriella A. B. Barros (modl.ai - Maceio, BR), Michael Cook (King's College London, GB), Gillian Smith (Worcester Polytechnic Institute, US), Tristan Smith (Creative Assembly - Horsham, GB), Anne Sullivan (Georgia Institute of Technology - Atlanta, US), and Tommy Thompson (AI and Games - London, GB)

Following on the narrative metrics workgroup day one of this Dagstuhl, in this working group we pursued the idea of characterizing generated narratives by considering stat changes associated with important story features.

For instance, a game or story generator that produced romance stories might track the degree of attraction between the protagonist and each of two suitors; a generator that allowed the protagonist to reach maximal attraction with either suitor at the end of the story could meaningfully be said to offer more narrative range than a generator that always required the protagonist to end up attracted only to one of them, but less range than a generator that also offered affordances for no attraction or for attraction to both suitors as viable final

outcomes.

Moreover, the trace could reveal meaningful variation during the midgame as well as at the endings: a generator that allowed for either a swift or a gradual development of attraction might be providing greater narrative range than one in which attraction always increased at a steady pace.

We decided to approach in particular the question of a story generator that was a playable interactive game, where we were interested in visualizing:

- characteristic behaviour of a single numerical stat (for instance, a player trace in which the player quickly achieves victory, in contrast with one in which the player gains more slowly or never wins)
- correlated behaviour of two stats (for instance, a player gains wealth in exchange for relationship status with a particular non-player character)

We expected that in some cases, the trace of a given stat from playthroughs would be tightly constrained (for instance, a player power stat might be monotonically increasing with only slight variations in the pace of gain), while in others, it might be almost fully unconstrained (e.g., a player's currency holdings might change at arbitrary times as the player gained loot drops or chose to spend money at a store). The most revealing traces, however, would be those for stats that were somewhat constrained by gameplay systems but which could nonetheless assume more than one shape in a meaningful way.

In order to explore these questions more rigorously, the team built several prototypes, as follow:

- Two short narrative games in the Ink programming language, in which significant story stats changed depending on player choice. One represented a tea-time conversation with a Mr Wickham and Mr Darcy, with stats representing their reaction to the player, and the other which represented travel aboard a leaky raft, where the player needed to try to reach the shore before the raft sank

- A 2D visualization tool that graphed the progress of these stats in player traces generated from these games; different stats were mapped on top of one another in the same 2D space, which allowed the viewer to recognize correlations between particular stat pairs, but which made it difficult to present many traces simultaneously in order to understand the overall generative space of the story
- A 3D visualization tool that showed stats on different axes together with their change over time

The team further speculated about an MRI-like visualization that would show slices through a three-dimensional state space, showing a heat map arising from many playthroughs in order to help identify the typical emergent shapes. We distributed links to the sample games to other Dagstuhl participants and were able to collect around half a dozen traces of player data for analysis.

Even with these simple examples, it was evident that there would be assorted challenges in using this approach to interpreting the shape of generated narratives. Our turn-based games allowed for only a small amount of variation in the length of the play session, but even in these circumstances, slight differences in the time axis could make it difficult to judge whether two traces were narratively similar or whether the delay in one run constituted a narratively meaningful distinction. The group discussed, but did not have time to explore implementing, ideas whereby the curve of other game stats could be normalized against specific "tentpole" game events.

The group also identified a need for qualitative validation: a core assumption of this approach is that players will regard games with different stat curves as describing different narrative experiences, and that developers (or the persons responsible for creating the analysis) will be able to anticipate which stats are relevant to assess in this way. This assumption would also need to be validated.

3.19 Meaningful Computational Narratives

Pieter Spronck (Tilburg University, NL), Maren Awiszus (Viscom AG - Hannover, DE), Gwaredd Mountain (Square Enix Limited - London, GB), Mike Preuß (Leiden University, NL), Hendrik Skubch (Square Enix AI & ARTS Alchemy Co. Ltd. - Tokyo, JP), Tristan Smith (Creative Assembly - Horsham, GB), and Tony Veale (University College Dublin, IE)

Generative AI offers the opportunity to generate narratives. For instance, a tool such as ChatGPT can take the setup of a storyline and expand it, taking into account the specifications of a story designer. There are, however, some issues with this approach to storytelling:

- There is no guarantee that the generated story will fit the specified background
- There is no guarantee that the generated story will be interesting
- There is no guarantee that the generated story will allow for a suitable continuation

The workgroup ran some simple experiments to test the limitations of a tool such as ChatGPT, and discussed approaches to ensure that generated storylines are meaningful.

3.19.1 Storyline quality

The quality of a generated storyline should be high. That entails that the storyline must be "meaningful" and "good". These two terms need further clarification.

The workgroup defined "meaningful" in the context of storylines as the requirement that within the story, all actions have observable consequences. This is particularly important within the context of games, as game players want to influence the story with the actions that they perform.

The workgroup defined "good" in the context of storylines as the generated storyline having the following four features:

- Semantic coherence
- Pragmatic coherence
- Understandable arcs
- Relatable arcs

The last feature, relatable arcs, is not an absolute necessity, but it tends to be really hard to get players to engage with a story in which there is nothing that they find relatable.

3.19.2 ChatGPT test

In testing the ability of ChatGPT 4 to generate a storyline fitting the requirements of a game designer, responding to player actions, the workgroup gave ChatGPT the following instruction:

You are the Game Master in a Wild West Story Campaign. We are currently in a scene where the player is about to find a caravan on the road which is currently being ambushed by bandits. The kind of Caravan, the motivation of the bandits, any names and characters are free for you to choose. There are three endings to the scene - either the player drives off the bandits with force, negotiates with them, or they side with the bandits and extort the Caravan. Make sure that one of these endings is reached at the end of this conversation, but keep the outcomes hidden from the player. Based on the players actions, use your descriptions and guidance to help them reach the ending that matches their actions. Start with a description of the scene and then react to the actions of the player accordingly.

ChatGPT responded by describing the scene as requested. The workgroup then told ChatGPT several times that the player was just going to hide and wait. ChatGPT responded by trying to make the scene more tense, enticing the player to act, which the workgroup refused to do. At some point, the workgroup asked ChatGPT for suggestions to how the player should respond. Somewhat surprisingly (considering the initial request), some of the suggestions would not lead to the desired outcome. E.g., ChatGPT offered the option to just "wait until one side has defeated the other", while the original request was that the player should actively side with the bandits, side with the caravaneers, or negotiate.

The request was that clearly, a particular story beat should be reached, namely one in which the player gets actively involved in the conflict and helps one of the opposing sides to be victorious. If this is necessary for the plot to evolve, then apparently, in its current incarnation, one cannot count on ChatGPT (which is one of the most advanced language generating tools in existence) to drive the story in the direction that the story designer needs.

3.19.3 Computational narratives

The goal of computational narratives in games is to allow the player the freedom, in a given situation in the game, to execute any action that they like, as long as it potentially fits the situation, while the overall plot of the game remains interesting. Generative AI may support computational narratives, but cannot be relied upon to achieve good results without support from other processes.

Terminology sometimes used by game developers is as follows: The *nebula* is the cloud that contains all possible stories. The *fabula* consists of locations, characters, events, and chronology which define the game world. The fabula restricts the storylines to a subsection of the nebula.

Figure 24 illustrates how a coherent story can be created within the subsection of the nebula that is constrained by the fabula. The story consist of a sequences of 'plotlets'. Plotlets are selected from a dataset. They can be linked together to form the game's plot. Which plotlet follows a given plotlet in a story, is determined by player actions.

3.19.4 Plotlets

A plotlet can be considered a story 'beat'. It consists of the following elements:

- Pre-conditions, which specify in which circumstances the plotlet can take place
- Action, which specifies which generally formulated action can lead to the plotlet

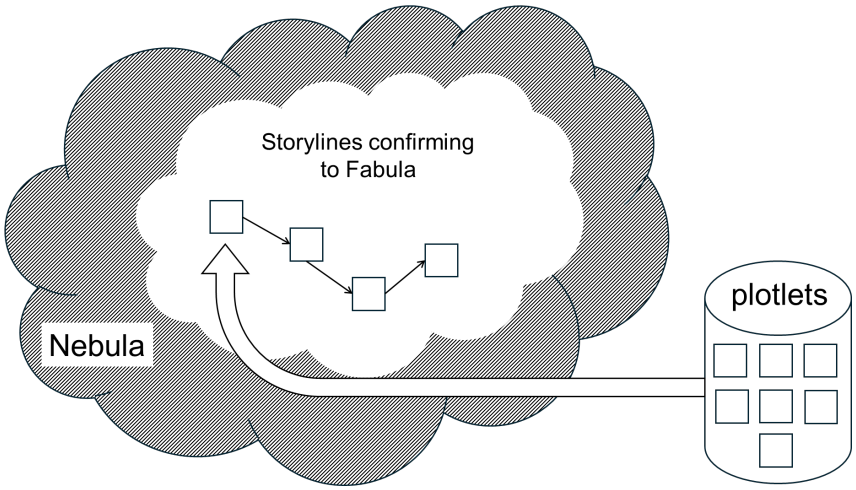


Figure 24 Dynamic storytelling.

- Description, which in general terms describes the situation that the player is in
- Selectable actions, which are generally formulated actions that a player can take when the plotlet runs
- Post-facts per selectable action, which define the new situation

When selectable actions are offered to a player, they need to be described in such a way that they make sense to the player within the confines of the story, and show that the dynamic storyteller understands what the player wants to do. For instance, if the story is the one described above, the player may be in a hiding place observing the scene, and as their action tell the game that they "cough loudly". In this particular setting, the dynamic storyteller should not assume that the player character has a heavy cold, but that they want to draw the attention from the opposing sides, or that they want to cause a distraction. This should be reflected in how the game continues.

The selectable actions in the scene above could be "attack bandits", attack "caravaneers", "step forward", or "distract". These options could be offered to the player in the situation, limiting them to actions that lead to other plotlets. The game could also offer the player free input as their action, but map this input to one of the selectable actions. For instance, if the player states that they make a handstand, the game could map this to "distract".

The overall plot of the game is restricted by the available plotlets, giving game developers some control over which stories are told, while still offering the player the freedom to progress the story in a way that appeals to them.

3.19.5 AI

When using plotlets, generative AI comes in as follows:

- The generative AI can describe the situation which exists at the start of a plotlet
- The generative AI can translate generally formulated selectable actions to actions which fit the current situation
- The generative AI can describe the result of the selected action based on the post-facts specified for the action

Moreover, as it is common that multiple plotlets would be available to continue a given story situation, AI analysis can be used to select a follow-up plotlet which best fits a good overall plotline.

3.20 Small, Safe LLMs for In-Game Generation

Tony Veale (University College Dublin, IE), Paolo Burelli (IT University of Copenhagen, DK), Amy K. Hoover (NJIT - Newark, US), Antonios Liapis (University of Malta - Msida, MT), Gwaredd Mountain (Square Enix Limited - London, GB), and Hendrik Skubch (Square Enix AI & ARTS Alchemy Co. Ltd. - Tokyo, JP)

Scaling laws for large language models (LLMs) have allowed LLMs to achieve dramatic improvements in prediction accuracy and generative quality as the depth of their architectures (the number of layers, and of learnable parameters) and the breadth of their training datasets (ever larger subsets of the world-wide-web) grow in size and ambition (Kaplan *et al.,* 2020). This impressive scaling allows LLMs to be applied to tasks that seem to demand more than mere gap-filling or next-word-prediction in text. However, the old truism, popularized by Stan Lee of Marvel Comics, that "with great power comes great responsibility" seems increasingly apt as LLMs grow in generative power. Their ability to produce novel and imaginative responses to arbitrary prompts – one might even say "creative" responses – gives them the ability to amuse and inform, but also an ability to misinform and offend. With the subtle and contextualized generalizations derived from their large training sets, and encoded in their large parameter sets, these models learn the best and the worst of the human condition. This duality, the ability to be used for good or (perhaps unintentionally) for ill, gives us pause when considering the role that LLMs might play in a new generation of computer games.

This working group, which explored the topic of "smaller, safer language models for games", explored whether smaller LLMs (so-called SLMs), with smaller and more selective training sets, can mitigate some of the concerns that are foreseen in a games context. The findings of our group are briefly summarized in the following sub-sections.

3.20.1 Are smaller LLMs inherently safer, or perhaps easier to make safer?

One can imagine that smaller LLMs that are trained on smaller and more controlled trawls of the world-wide-web will visit fewer of the dark corners of the web and include much less of the toxic content that lurks within them. Smallness is not in itself any guarantee of quality, for it is not the case that language users only learn to be abusive or offensive at a certain scale of language acquisition, after they have first learnt to be fluent and felicitous. Indeed, one can make the argument that large LLMs incorporate more knowledge of the world, tacit and vicarious as is is, and so can better reflect on why a certain response may be inappropriate or objectionable. That is, larger LLMs have more fully developed *moral imaginations*, or at the least the makings of such a capacity, and can better reason about their own goals and high-level instructions.

Offensive capabilities come in two forms: the ability to use words that are offensive in themselves, the kind of words that would traditionally go on a "blacklist" or "blocklist",

and the more creative ability to use inoffensive words to achieve offensive ends. Even if we carefully control the training of a small LLM so as to exclude the latter, we cannot guarantee that the LLM lacks the capacity for the latter, intentionally or otherwise.

Nonetheless, for tightly-controlled tasks in tightly-controlled domains, such as a well-defined game world, smaller LLMs can be trained from scratch to serve the specific needs of that world, and to only speak the language of that world. Our group considered, for instance, the fictional world of an Old Wild West game, and the generative needs of this domain, including e.g., the language of settlers, of bandits, and of Native Americans (who would not be described as such within the game). We certainly would not want our NPCs to use modern idioms or other anachronisms that would diminish the suspension of disbelief by the game's players. A smaller LLM can be more easily moderated for such a domain, to exclude e.g., references to Native Americans as "savages." These references were commonplace in the Wild West movies of the 1950s and 1960s which still shape our expectations of such a game today, but they are no longer acceptable, even if placed into the mouths of fictional characters that players are not expected to identify with. In short, then, smaller LLMs better allow us to tailor their generative capabilities to the sensitivities and needs of a specific game or brand.

3.20.2 Can smaller, bespoke LLMs be packaged with games and used locally?

A key issue with using LLMs for games concerns the latency and cost of using commercial models in the cloud. For an individual user of OpenAI's LLM offerings, say, the costs are typically low: less than a penny for most calls to the API. However, a game with many thousands of users will accumulate API costs that can prove onerous to the developers of games that rely on real-time access to commercial models. It becomes more practical then for games developers to bundle their own, smaller LLMs with their games, so that these SLMs can be run locally, on the machines of individual players rather than in the cloud infrastructure of the developer (or a third-party provider).

If we accept that SLMs can be bundled for local use in this way, the question now becomes: is an SLM capable of generating what a game needs, when it needs it, with the necessary quality, and with the sufficient speed so as not to drag on the game play? To explore this question, our group conducted some experiments with a small-ish LLM (or a large-ish SLM) called *TinyLlama*. The tiny model is a member of Meta's *LLama* family of LLMs that has just 1.1 billion parameters (contrast this with the 8 billion, 70 billion and 405 billion of different versions of Llama 3.1). To further shrink the model's footprint, we used a 4-bit quantized version of *TinyLlama*, which allocates just 4 bits apiece to each of its 1.1 billion parameters. Although *TinyLlama* has been pre-trained on a smaller corpus than its larger Llama siblings, its dataset is still substantial at three trillion tokens of text. The model is compact enough to load into a basic (unpaid) *Google colab* environment and to run with the benefit of a single T4 GPU.

Our experiments focused on narrative generation within games, by exploring whether *TinyLlama* can be fine-tuned (using a library named *Unsloth*) for the following tasks: to generate a short textual narrative from a skeletal plot structure (or "fabula"); to generate these skeletal plots for itself after it is fine-tuned on a set of more than 10,000 fabula structures (derived from a symbolic story-generation system named *Scéalextric* (Veale, 2021); and to generate dialogue for the characters in these plots, for each step in the plot (as numbered in the fabula). We built a composite training set with examples of each of these three tasks, and 100 steps of fine-tuning on this dataset requires less than 5 minutes of GPU time in a Colab environment.

3.20.3 Findings and Conclusions

Our findings at the stage are mixed. We are impressed with the competence displayed by the *TinyLlama* model on these three tasks, and consider the model's own outputs to be of good quality, very much in keeping with the tenor of the fine-tuning data. The model can generate new fabula structures of its own; impose a fluent rendering on this skeletal form; and generate apt dialogue for two characters that reflects the core plot and the narrative rendering. However, this competence also incurred a noticeable latency that we feel is too great for inline use of these generative capabilities in games. Still, as some careful engineering may yet allow us to reduce this latency to allow for real-time generation with an SLM in games, we are encouraged by the current triumph of competence over performance when using SLMs for "creative" games-related tasks.

References

1 Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeff Wu and Dario Amodei. *Scaling Laws for Neural Language Models*. ArXiv abs/2001.08361, 2020.
2 Tony Veale. *Your Wit Is My Command: Building AIs with a Sense of Humor*. Cambridge, MA: MIT Press, 2021.

3.21 Transferability of Game AI

Vanessa Volz (CWI - Amsterdam, NL), João Miguel Cunha (University of Coimbra, PT), and Tristan Smith (Creative Assembly - Horsham, GB)

3.21.1 Motivation

While many interesting discoveries have been made in the field of Game AI in recent years and beyond, their application in the games industry is still very limited. This is due to a broad range of challenges, including non-technical issues. One obstacle is undoubtedly that it is seldom obvious how a given algorithmic problem encountered in a game industry setting (e.g. a resource-efficient and believable opponent) can reliably be solved with AI, i.e. with some form of performance guarantees. This is especially important if the problem is expected to change during the course of development of the game; it needs to be guaranteed that changes will not break the AI performance completely.

To do this, we would need to be able to identify what part of an existing solution is transferable to which other problem (in the same game or beyond). In the following, we therefore first address which different types of transferability may exist. We then discuss some approaches to comparing problems as one popular way to identify opportunities for transferring knowledge and/or algorithmic approaches. Finally, we point out directions for future research.

3.21.2 Transferability dimensions

Transferability can have various realisations in different settings. For example, we have seen the great care that had to go into training AlphaStar in order to show somewhat promising behaviour against different opponent strategies and game versions [1].

Even algorithms intended to be more general, like the participants in GVGAI competitions, tend to have different strengths and weaknesses [2]. This also holds for different hyperparameter settings of said algorithms. While we can certainly explain some behaviour, a reliable method to identify different games where we would expect to see similar performance patterns is an unsolved problem [3].

In addition, in image recognition and generation, we know that re-training only the last few layers of a neural network can achieve results faster than training from scratch (transfer learning). Some learned parameters and structure of the neural network must thus transfer to images in general. Relatedly, there has been some work on training weight-agnostic networks that might be faster to adapt to different environments [4].

We can characterise the above examples of transferability challenges based on two dimensions of transferability:

- Size of gap between contexts: Are we transferring between different game versions of the same game? Different game levels? Different games of the same genre? Across genres? To contexts beyond games?
- Level of abstraction: What is it that we attempt to transfer? A trained model? The training setup and associated hyperparameters? Knowledge about strengths, weaknesses, and performance estimates of different approaches? Predictions on training costs and resources?

3.21.3 Problem similarity

In order to investigate which classes of transferability are realistic, a comparison approach between different problems (games) is needed (among other things). We list some existing ideas below:

- Anecdotally, in the game industry this topic is approached by characterising different tasks based on traits like independence/parallelisation, allocation/optimisation, domain considerations, necessary accuracy and speed, and degree of design control.
- In games research, game genres (platformer, shooter etc) are often utilized to characterise games. Games of the same genre are often assumed to be similar to each other, as well as levels of the same game.
- There are also hand-crafted features for game characterisation in the context of different general game playing frameworks [5, 6, 7].
- Research on level blending in the context of procedural content generation (via machine learning) often requires representing different (selected) games with the same representation, which allows for comparisons in that representation space [8]. See also the section on distance and density measures 3.13.
- In related domains (e.g. evolutionary optimisation), various hand-crafted and data-driven features exist for the purposes of problem characterisation as well as for automatic algorithm selection. However, this is still an active area of research when it comes to interpretability and robustness of the features, as well as transferability of the results in different dimensions and settings. [9]
- In popular culture, games are often described in the context of other games. A concept of similarity is therefore implicitly established. This includes gameplay, but also cultural, location and historical connections [10, 11].

3.21.4 Future directions

Beyond what was described in section 3.21.3, further avenues for research exist in terms of problem comparison and transferability of game AI research in general:

- Compute or automatically select features, for example based on:
  - an encoding obtained from large language models processing a game description (prose, game description language, player reviews).
  - performance comparison of different AI and human players.
  - human labelling to allow for supervised learning.
- Identify what aspects of a game might be subject to change during the development process, and ensuring robustness against these changes in the game-playing AI (Transfer across versions, unclear abstraction level).
- Computing worst-case performance for games as a way of establishing guarantees (Transfer performance estimates, unclear context gap).
- Based on the identified dimensions described in section 3.21.2, survey and categorise existing work on transferability / generalisability.
- Continue research on robustness of game AI using techniques such as adversarial training and curriculum learning.

References

1 O. Vinyals et al., "Grandmaster level in StarCraft II using multi-agent reinforcement learning," Nature, vol. 575, no. 7782, pp. 350–354, 2019.

2 D. Perez-Liebana, J. Liu, A. Khalifa, R. D. Gaina, J. Togelius, and S. M. Lucas, "General Video Game AI: A Multitrack Framework for Evaluating Agents, Games, and Content Generation Algorithms," IEEE Trans. Games, vol. 11, no. 3, pp. 195–214, 2019.

3 H. Horn, V. Volz, D. Pérez-Liébana, and M. Preuss, "MCTS/EA hybrid GVGAI players and game difficulty estimation," in IEEE Conference on Computational Intelligence and Games (CIG), 2016.

4 A. Gaier and D. Ha, "Weight Agnostic Neural Networks," in Advances in Neural Information Processing Systems, Curran Associates, Inc., 2019.

5 P. Bontrager, A. Khalifa, A. Mendes, and J. Togelius, "Matching Games and Algorithms for General Video Game Playing," AIIDE, vol. 12, no. 1, pp. 122–128, 2021.

6 A. Mendes, J. Togelius, and A. Nealen, "Hyper-heuristic general video game playing," in IEEE Conference on Computational Intelligence and Games (CIG), 2016.

7 M. Stephenson, D. J. N. J. Soemers, É. Piette, and C. Browne, "Measuring Board Game Distance," in Computers and Games, vol. 13865, in Lecture Notes in Computer Science, vol. 13865., pp. 121–130, 2023.

8 V. S. R. Atmakuri, S. Cooper, and M. Guzdial, "Game Level Blending using a Learned Level Representation," in IEEE Conference on Games, 2023.

9 A. Nikolikj, S. Džeroski, M. A. Muñoz, C. Doerr, P. Korošec, and T. Eftimov, "Algorithm Instance Footprint: Separating Easily Solvable and Challenging Problem Instances," in Proceedings of the Genetic and Evolutionary Computation Conference, in GECCO '23, pp. 529–537, 2023.

10 J. P. Zagal, "A framework for games literacy and understanding games," in Proceedings of the 2008 Conference on Future Play: Research, Play, Share, in Future Play '08, pp. 33–40, 2008.

11 A. de Voogt, A.-E. Dunn-Vaturi, and J. W. Eerkens, "Cultural transmission in the ancient Near East: twenty squares and fifty-eight holes," Journal of Archaeological Science, vol. 40, no. 4, pp. 1715–1730, 2013.

4 Panel discussions

4.1 Discussion

Pieter Spronck (Tilburg University, NL), Duygu Cakmak (Creative Assembly - Horsham, GB), Setareh Maghsudi (Ruhr-Universität Bochum, DE), and Diego Perez Liebana (Queen Mary University of London, GB)

As per usual, on the Friday before lunch we held a plenary meeting where we were going to reflect on the past week, and look forward to a potential follow-up. This is a report on the discussion.

4.1.1 Preparing for the event

Several participants remarked that they found it a positive element that it was not necessary to specifically prepare for the event. However, for newcomers this was not immediately clear from the invitation. Moreover, some preparation possibilities could be welcome. The Discord group set up for the event would make this relatively easy, by, for instance, having a resources channel with documents, links to tutorials, and tools that may be used during the event in case delegates want to install them beforehand.

More work could also be done to connect people with each other. People spontaneously connected through Discord, specifically for travel arrangements, but we could try to let people also connect for topic discussions.

It might be nice if some sponsoring can be found to support people with less financial means to come.

4.1.2 Schedule

The general schedule for each day was a plenary session of about half an hour after breakfast, followed by workgroups. In most cases, the workgroups that ran before lunch were also run after lunch. There was some time set aside right after lunch for those who wanted to take a walk. Around 5 o'clock a one-hour plenary session was held in which the workgroups reported on their results. After dinner there was time for social activities; some of these were organized (a pub-quiz – which should probably be limited to three questions rather than six) and some spontaneously organized (a tutorial, a movie, and two roleplaying games), but most consisted of people spending time together in discussions and game-playing.

The organizers had been struggling a bit to reserve time for people exercising. It was considered to move the last plenary session of the day to after dinner, so that the time slot between five and six o'clock could be used for exercises, but ultimately on Monday morning it was decided not to do that, and just allow people to leave a workgroup earlier if they want to exercise. Most agreed at the end of the week that this is the best approach, as the days are long and intensive and we should not schedule work activities after dinner.

4.1.3 Monday

The Monday tends to be a day where too much happens. We started with an introductory game which took quite a lot of time. The game used was "two truths and a lie," which took over an hour-and-a-half to run. This is clearly too long. It was suggested to simply

let everyone introduce themselves quickly with one sheet (and have the sheets available for perusal in Discord), and maybe have an introductory game on Monday after dinner. It would even be possible to do this on Sunday night, though not everyone may be present then, and some who had to travel for a long time might already have retired. A small questionnaire up front, to which the answers are shared via Discord, may also work.

After the introductory game workgroups were proposed. There was just enough time to do this before lunch, so that the workgroups could start after lunch. The problem is that these workgroups were relatively short, lasting only half a day; no more than two hours total (excluding coffee breaks). With a bit more preparation this time could be used more effectively. We could plan tutorials on the Monday afternoon, and/or preparation work for workgroups that are going to be run from Tuesday to Thursday. This would work best if some workgroups would already be known before the event takes place. For instance, collecting information before the event on workgroup ideas via online forms could make the use of time more effective.

4.1.4 Organizing workgroups

Some suggestions were made to streamline workgroup organization. In particular, sometimes people were trying to find the rooms where certain workgroups were held. A good suggestion was to have a large grid hanging in the plenary meeting room, with a list of rooms and an indication where workgroups are located. It would also help if it is made explicit somewhere (either on such a grid, or in Discord) what a workgroup intends to do. This will help participants who want to have a ‘ bumblebee’ approach to visiting workgroups.

4.1.5 Workgroup presentations

Workgroup presentations are very useful to communicate to all participants what a workgroup did. Doing this with sheets is extra helpful, as those sheets can then later be used to base proceedings on, and can be used as a point of reference for all participants. However, there are at least two issues with the sheets. The first is that some presentations were rather long, and since there usually were five presentations in the one-hour time slot before dinner, sometimes the later presentations were pressed for time too much. The second is that sometimes participants were still working on sheets during the presentations, which meant that they could not follow the presentations others were giving.

The first may be solved by limiting the number of sheets to five, and keeping stricter track of time. The second may be harder to solve (though limiting to five sheets may help for that as well). However, this shouldn’t mean that information that is relevant is not kept track of using sheets, only that a maximum of five sheets should be presented.

Another helpful suggestion to save time was that if a workgroup intended to continue the next day (which happened for multiple workgroups), their presentation should be really short and mostly focusing on what will be done the next day. This would mean that several presentations would be use a considerably reduced amount of time.

As support for the presentations, people also suggested using our Discord server having a dedicated channel for outputs of the workgroups, or a Google Drive folder dedicated to this purpose, where each group stores their final materials. This could contain presentations, links, code and even their final workgroup reports.

4.1.6 Rules of conduct

Because we want the event to be a safe place, we introduced rules of conduct for a previous event, and continued using them now. However, we now realized that if someone has an issue, having them contact the organizers may not (always) be the best approach. We should appoint two trusted people (one man, one woman) as independent confidential supports. We still do not expect that they will have a lot of work to do, but having them present would be appreciated.

4.1.7 Topics for future events

The *Creative Game AI* theme for the event was really topical, and almost all (if not all) workgroups focused on the theme, and almost all (if not all) topics presented in the proposal were researched. The organizers elicited suggestions for future topics. The following were suggested:

- Games and open-endedness, artificial life, creativity
- Emergence
- Relatedness between human intelligence and game AI
- Transferability from games and computer science to other disciplines
- Research practices, holes in methodologies, comparing methods
- Game AI and education
- Ethics and morality, both their influence on games and how they can be "improved" with games

It was also suggested to replace the term "game" with the term "play", to expand the subject area.

Considering the success of the event and the enthusiasm of the participants, an event in 2026 would be very welcome.