2.

TOG

A Model for Innovation-Centric Design in Games and Expressive Interactive Media

Mirjam Palosaari Eladhari & Hartmut Koenitz

Transactions of the Digital Games Research Association
November 2021, Vol. 5 No 3, pp. 13-42. ISSN 2328-9422

ABSTRACT

This paper describes an approach to facilitate innovation in game design by increasing the designers' palette of playable and participatory computational expressions. The TOG model (Technology, Ontology, and Game Genre) can be used in teaching game design and related practices, but is also applicable to prototyping in professional settings. TOG is inspired by the processes of AI-based game design, and introduces the concept of the techno-artistic minimum. It was conceptualized when teaching

a course on computational expression at Malta University. The main aim for teaching with the TOG model was to facilitate innovation by challenging aspiring game designers to think ‘outside the box’ and come up with unusual and innovate creative solutions. In addition, TOG can complement existing design methods such as MDA and DDT in the practice of professional game designers.

Keywords

pedagogy, game design, teaching, AI-based game design, case study, computational expression, TOG model, expressive effect, intentionality, mental model

INTRODUCTION

With a professional practice extending back to the 1970s and an educational practice in higher education going back to the first game design program at Abertay University in 1996, we need to face the fact that “routine game design” becomes an issue for both the professional practice and education. It is therefore timely to consider planned interruptions using methods designed to disrupt the well-trodden paths (or rather, multi-lane boulevards by now) of game design and challenge designers to consider problems they would not otherwise have engaged with. In this paper, we are introducing a model designed to facilitate such creative interruptions: TOG (Technology, Ontology, and Game Genre). We describe the model’s foundations, components, and concrete application in game education. More concretely, TOG uses unexpected ontologies, technological approaches and applications not yet commonly used in games (for example, because the technological advances are so new that they pose a risk to stability in a shipped game) and settings/environment outside of classic (worn out) fictional universes of medieval fantasy, steam punk or space travel. Furthermore, the TOG model assumes an understanding of the minimal requirements for coherent games

systems (which we will describe as the ‘techno-artistic minimum’). The main aim of the TOG approach is to facilitate innovation by means of out-of-the-box design thinking. The TOG triad engenders new ideas and approaches growing on the fertile creative ground of unexpected combinations of technology, ontology, and game genre that serve as starting points for a given design.

We also discuss concrete results as examples of what this approach can accomplish. In order to illustrate how the TOG model can be used in teaching, we present a case study from a course of five ETCS credits, which was taught at masters’ level. Our example is of particular interest to educators who are teaching students from mixed educational backgrounds. In such settings, a TOG-based approach can help student groups to work in ways that enable them to harness their existing knowledge, and gain new means of expression in collaboration with their peers. Students participating in the course have described it as positively challenging, stating that it helped to expand their horizon, and that they were inspired to make things they did not expect to make. In addition, many of the students participating in the course later based their dissertation work on the design experiments and prototypes they made using the TOG model.

The TOG approach is informed by work reflecting on processes in AI-based game design (AIGD) (Eladhari et al. 2011). In this context, it is crucial to recognize that game creation is, fundamentally a liberal art, even when taught within the engineering disciplines. Game creators build worlds and formulate ontologies, and as a foundation for their work, often read up on a plethora of subjects for inspiration – biology, art, music, psychology, economics, politics, learning sciences, architecture, and more. It is in this particular sense that we use the term ‘computational expression’ to denote computational methods as a means for artistic expression. Given the particular nature of video games as expressive works made for the active exploration and co-development by audiences – we need to consider the need for

evaluation criteria different from those of traditional engineering disciplines. In this regard, Horswill et al. argue:

> […] the evaluation criteria of computational media are ultimately aesthetic: the value of a piece lies in its ability to engage its audience, and the value of a technology lies in its ability to allow artists and designers to develop engaging pieces. (Horswill et al. 2019)

This perspective continues Janet Murray's line of thought in her distinction between the affordances and aesthetic qualities of the digital medium (Murray 1997). The broadness and openness of systematic perspectives and combinatorics at the heart of game design give rise to a rich design space, full of unmapped terrain and novel opportunities. The TOG approach is intended to prepare and enable students to realize the expressive potential of computational expressions, to make creative use of the "digital plenitude", as Bolter (2019) recently put it.

In summary, the aim of this article is multi-fold: to a) offer an approach for overcoming what we call the 'techno-artistic minimum', the threshold of successful game design, b) introduce the TOG model as a means to facilitate innovation and exploration in computational expression, and c) demonstrate how the TOG model can be used in teaching.

BACKGROUND

In this section, we will describe the conceptual background of the TOG model and its triad of *technology*, *ontology*, and *game genre*, as well as the *techno-artistic minimum*. TOG model draws on work on AI-Based Game Design (AIGD). In short, AI-based game design is the creation of games where the game mechanics are intertwined with the AI systems used to realize the game. Examining design process in making AI-based games, Eladhari et al. (2011) identified distinct processes in that approach by means of case studies. The authors also found that a common

denominator in the practice was to consider the respective knowledge domains in triplets, e.g.

1. a main subject area, theme or theory,
2. an AI method, and
3. game genre convention(s).

For example, a game whose **main theme** is musical theory invites different types of play activities in comparison to a game based on collaborative storytelling. Often-used **AI methods** can both constrain and open up a design space. For example, adopting a belief-desire-intention architectural approach (Rao and Georgeff 1995) for autonomous entities in a game would imply that autonomous entities should be able to perceive a world, believe something about it, desire something, and have means to satisfy that desire. **Game genre** conventions, such as the typical challenges that players face in real-time strategy games, computer role-playing games, or first-person shooters shape the affordances designers create for players within the systems. The overall tripartite segmentation provides the inspiration for the TOG triad.

Techno-Artistic minimum

A basic goal of game design pedagogy is to reach the *techno-artistic minimum,* by which we mean that technology and artistry need to form a minimal ‘happy alliance’. Both the technological and the artistic sides need to come together sufficiently well in a design to enable experiences that can be compared to the original vision. Thus, the *techno-artistic minimum* describes the threshold that an artifact needs to reach in order to be a viable video game prototype, understood as a playable experience. In terms of skillsets, this means that a) students and designers with engineering backgrounds need to have sufficient consideration for aesthetic and experiential aspects in order to create a satisfying player experience, and b) that those with an artistic background

need to acquire sufficient proficiency to have a technological palette of options to work with.

The challenge for educators here is to develop both sides of the techno-artistic spectrum and build a conceptual-aesthetic understanding, along with the technical skillset of their students so that they can reach the *techno-artistic minimum* in their own work. This also means that any educator originating in the humanistic or social sciences would be severely handicapped by not understanding the technologies underlying computational expression, while computer scientists and engineers would be hampered without an appreciation of expressive categories. It is a fundamental challenge of game design teaching to develop an understanding of the expressive opportunities afforded by the combination of technology and art, and how to reach the *techno-artistic minimum* as a foundation for more advanced skills.

THE TOG MODEL

Before describing the TOG model in detail, we want to be clear about our aims with it. The focus here is *not* on making artifacts ready for public consumption, but rather to create experimental works that demonstrate a concept, allow for play testing, and enable critical reflection on its underlying ontologies, systems, processes, and genre conventions. Consequently, the TOG model is not meant as a model for understanding how to develop a 'good' game, or to provide a mapping or framework for analyzing existing games. Instead, it is a tool to spur innovation in game design and thus enhance an individual learner's palette of computational expressions. As such, the focus is on the artistic process, the journey to innovation, and the expansion of artistic registers and design approaches.

The TOG Model (Figure 1), abbreviated from *technology*, *ontology*, and *genre*, aims to facilitate two important goals in teaching game design: to spur conceptual innovation by breaking out from conventional game theme/fictions and genres, and to

encourage experimentation with technology, both in terms of using existing computational processes and by developing new ones, as well as using non-conventional technologies for player input. As the authors found the triadic approach of AIGD useful for explorative research in AI-based games, a tripartite approach offers a promising foundation for a model used on teaching methods for computational expression in connection with game design.

The **technology** category covers both computational processes and the use of different types of hardware. Technology is thus used in very broad terms. When it comes to computational process, this aspect is cutting across different categories from the foundational architectural layer of a given game design, up to the representational level where the player interfaces with the game – what Walk et al. (2017) call the "experience layer" in the DDT framework (which in turn is an improvement of the MDA model for game design by Hunicke et al. from 2004). This aspect of the TOG model is similar to the concept of "operational logics" as defined by Mateas and Wardrip-Fruin (2009), and further refined by Osborn et al. (2017), in that the technology, or processes an operational logic can consist of, is neither beneath nor above mechanics, but represents a different slice through a game, cutting from system architecture to what effect it may have on the player experience.

We specify **technology** in this loose way to invite experimentation. For example, in the first iteration of the course (presented as a case study below), the original project description mandated the use of a small section of specific technological approaches such as procedural generation or machine learning. However, some students were keen to experiment with new hardware, and there was nothing in the learning goals of the course that would motivate curtailing this enhancement. On the contrary, it opened up a space for further and broader experimentation and a reflection on how expansive the notion of computational expression can be.

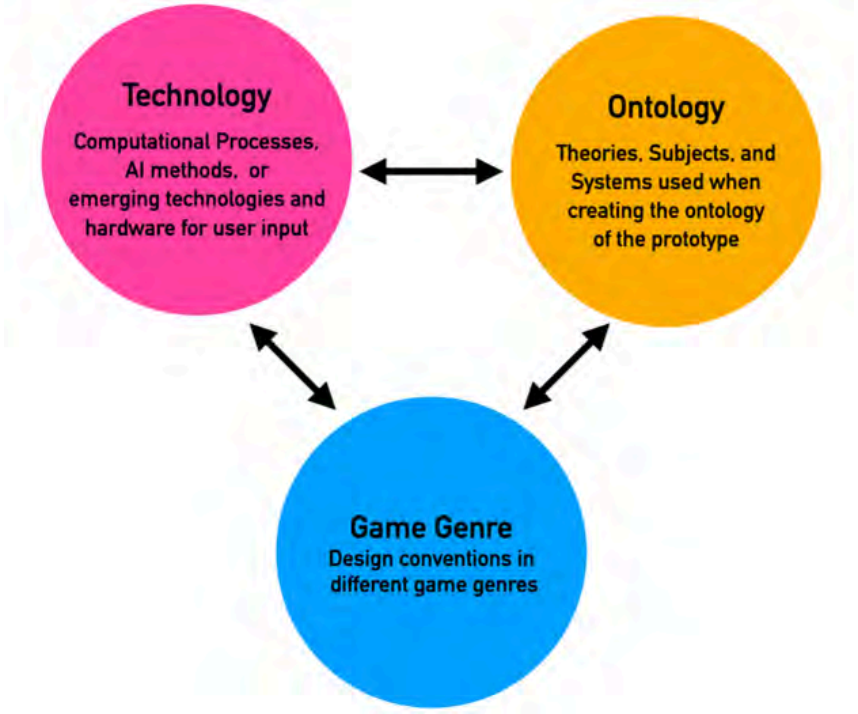

Figure 1: The TOG model

An **ontology**, in its lexical definition, has two meanings: "the branch of metaphysics dealing with the nature of being", and "a set of concepts and categories in a subject area or domain that shows their properties and the relations between them." In the context of TOG, we understand ontology in the latter sense. Creating ontologies is what game system designers do. It is important to reflect that an ontology is more than just a theme. Any game system has an ontology created by someone, since the virtual world must be defined in its entirety; which things exist in the world, what the things consist of, and how they relate to each other. Conversely, ontologies determine a user's perception of, and interaction with, a game. In the TOG model, we pick ontologies not commonly used in game design (for some examples, see the section describing prototypes in this paper), as a targeted intervention, as a layer of both creative constraint and inspiration, to facilitate problem solving in novel and potentially unexpected

ways. The selected ontology is used as a starting point for the construction of an ontology specific to the prototype, that informs the overall system design.

Game genre is the third component of the TOG model. Game genre conventions impact both the design process/choices and players' expectations. For example, in the genre of role-playing games (RPGs) we would expect to find a facility for skill selection and a way of levelling up skill values. As designers, we bring past play and development experiences to our projects, often unconsciously. As Bartle (2003) noted, we have a tendency to want to re-create our first deeply meaningful game experience. Furthermore, as designers, we are often asked to create a work of a certain genre – consequently thinking in terms of genre is widespread in game development.

In the TOG model, we use game genre in two ways: first, as a starting point for the design, an established set of conventions for game rules and game mechanics, and secondly, to instill the awareness that the choice of game genre is an *active*, conscious one, with considerable consequences. Making the choice of game genre an explicit decision in the design process helps to raise awareness of its benefits, but also potential pitfalls. Specifically, there is a danger that genre conventions are taken for granted and thus become a foregone conclusion, unnecessarily limiting the design space. An explicit consideration of game genre enables productive engagement with the concept, and can help foster novel computational expressions. Games genre also serve as an indicator of difference from established genres, since the use of unconventional ontologies is designed to create a productive tension with the concept of game genre.

At this point, some readers may wonder where in the TOG model they can find an equivalent to Hunicke's (2004) and Schell's (2008) layer of "aesthetics", or – as Walk (2017) and Winn (2009) in their models call it – the experience (of the player) layer. In the TOG model, this aspect is not specifically spelled out, as our

focus is on experimentation and a focus on the designer and their learning process, and not on creating products for the end-user. However, the player experience is central to the reflection phase in an implementation of the TOG model.

Implementing TOG: Concept and Realization Phases

In a concrete application of the TOG model, we differentiate the three phases of *concept*, *realization*, and *reflection* (Figure 2). In the concept phase, students develop a concept taking into account the three given elements of a TOG challenge. In the realization phase they develop a project, with the aim to reach the techno-artistic minimum necessary for a playable prototype and as a prerequisite for the reflection phase.

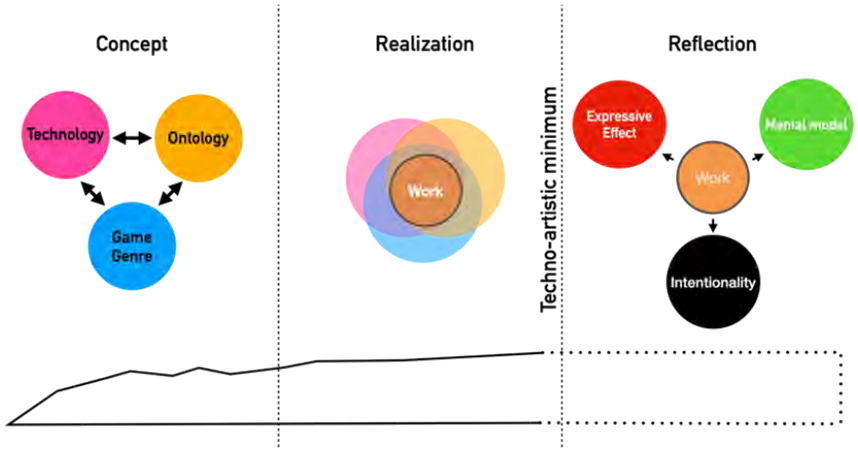

Figure 2: TOG model and implementation phases

It is important to point out that the TOG triad is only the starting point of the design process: once the work begins, the different parts of the triads feed into and affect each other. Through these interactions, seams, ruptures, and undefined spaces appear at multiple levels, fostering innovation. For example, a technology may not be able to cater to the design – hence an existing computational method needs to be improved, modified or invented. In another instance, a particular design may not fall into

any existing category, or established game design conventions may not cater to them, and consequently, a novel game category, design approach or type of game play is created. In these ways, the TOG approach is of particular use to facilitate innovation,

Reflection Phase

In the reflection phase, designers consider the work from three different perspectives: mental model, expressive effect, and intentionality.

We routinely form **mental models** of phenomena we encounter, starting in childhood, e.g., when we first try to understand what a cat is. Such models are changed and reinforced through learning, and inform our actions in daily life. In particular, they enable us to perform both routine and new tasks (by contrasting to existing knowledge and adjusting to new circumstances) and thus also inform players' conception about how something should work in a game. For the purposes of game design, it is important to take these models into account and make productive use of them (Puerta-Melguizo et al. 2002). In the reflection phase, we identify the mental models that players have formed and whether these reflect the original design intention reflected in the constructed ontology.

The reflection on the **expressive effect,** here understood as the process-experience ratio, is concerned with the workings of the underlying computational system and how a player perceives that aspect. This category is inspired by Noah Wardrip-Fruin (2009), who in *Expressive Processing*, describes three different effects of authored computational processing. The first is the ELIZA effect, where the user ascribes more computational capability to a system than there actually is. Joseph Weizenbaum's original ELIZA program (1966) was an AI experiment simulating a Rogerian therapist (a style of therapy where therapists ask questions based on patients' remarks). ELIZA was able to sustain shorter "therapy sessions" based on a clever combination of computationally simple methods such as keyword identification and repetition of the users'

utterances. In game AI programming the ELIZA effect is often seen as a form of cheating, as 'smoke and mirrors.' Yet, if a given design creates an enjoyable experience for the player, then there may not be a need for more complex computation in the first place.

The second is the Tale-Spin effect (Wardrip-Fruin 2009). It happens when an elaborate and ambitious system does not result in a level of player experience quality that matches the effort of creating it. The TaleSpin system (Meehan 1977) was a masterpiece of system engineering that could generate stories, but it was more appreciated by fellow system designers than by the users. Thirdly, there is the SimCity effect (Wardrip-Fruin 2009), and this is where the computer processing made for the system creates an immersive, complex, and dynamic experience for the player. SimCity (Wright 1989) is a simulation game where players define cities. As the cities grow, areas respond differently, and players learn to understand the system's operation as a process of play. The SimCity effect occurs when players' understanding of a system matches its actual operation.

Thus, we can see expressive effect as a scale reaching from the overestimation of the capabilities of a computational system to a match between perception and actual abilities (we might call this the 'techno-experiential balance') to an underestimation of the computational system on the other end, where there is a lack of transparency of its capabilities.

When the notion of effect is considered in the reflection phase it is from this backdrop: where does the prototype fall on this 'techno-experiential scale'? This also means that individual works can be evaluated along the scale – they do not have to be placed at the extremes or at the perfect center – resulting in a more granular instrument of reflection.

Intentionality is described by Dennet (1987) as the player's observation that an agent in a system is acting rationally, towards internally held goals. When players encounter an AI system in

a game, they assign intentionality to that system, “using words whose meanings go beyond the mathematical structures” (Agre 1997). They create narratives that rationalize the AI’s actions and reason about the AI’s goals (Sengers 2000). Hence, it is an indication of a successful design, if a player can read intentionality into a system or into components or agents in a system.

When a system does not have sufficient reactive-expressive capability to support the intentionality players read into it, or when a system fails to communicate its technical capabilities, when it strays too far from the balance of the center at the techno-experiential scale, this means that the believable (in Loyall and Bates’s sense, 1997) immersion of the SimCity effect is lost. Therefore, reflecting upon how players perceive the intentionality of a system and its acting component is a crucial part of using the TOG approach. If players assigned intentionality, this is positive because it means that players are actively creating belief (Murray 1997) (in contrast to Coleridge’s suspension of disbelief (1894)). In particular, it is useful to reflect on what entities, or parts of a system, users assign intentionality to, and how. The following table contains a matrix of how to categorize and evaluate intentionality (Table 1). In concrete usage, not all columns need to be filled, e.g., a given process might not be perceived as breaking.

Perceived intentional process	**What is the perceived intention?**	**Is the perceived intention sustained? How?**	**Does the perceived intention break? Why?**
Encountered NPC (agent, human)	Independent life	Dialog in line with overarching narrative and NPCs personality	Repetitive actions, out of character utterances
Weather (in-game system)	Weather phenomena governed by gods	Weather is changing according to gods’ moods	No weather effect, even when the gods are angry.
Political society (in-game system)	Dynamic politics with competing factions	Political events happen (e.g., mayor is elected, new laws are enacted)	Player character not affected by changes in society. Faction representatives act out of line with their ethos

Table 1. Intentionality

The categories of *mental model*, *expressive effect* and *intentionality* provide a rich toolbox for reflection on a prototype designed with the TOG model approach. As such, these criteria may be seen as aspects of the type of aesthetic evaluation Horswill et al. (2019) called for (and before them, Murray (1997) described as foundational to the digital medium). The development of a full set of aesthetic criteria for the evaluation of AI-based games, as well as other designed digital interactive experiences is outside the scope of this article and remains a task for the future.

In the following section we will report on the course where the TOG model was conceptualized.

Development of the TOG model and use as a method in education

The course "Computational Expression" was designed by the first author at the University of Malta and was offered for two consecutive years (after which the first author moved to a different institution). In the first year, the course was focused entirely on AIGD. In the second year, the approach was broadened, allowing students to base their designs on any significant technology of their choosing – the technology did not have to be an AI approach, but instead could apply new tools for interaction, for example bio-feedback sensors or virtual reality headsets.

The courses were taught at masters' level. The majority of students had their main educational background in computing, which helped lower the threshold for using the technological approaches involved. However, students with other backgrounds were accommodated with development tools that did not require prior programming knowledge, but still provided hands-on experience in using AI approaches, authoring systems and different types of input and display systems. The initial course had five students, the second iteration eleven. The course was structured into the following work phases. The first phase in the course, *knowledge*

gathering, is not represented in the TOG model proper, but part of its implementation was used as a method of education and a prerequisite for the subsequent phases.

1. Knowledge gathering,
2. Conceptualization,
3. Development and play testing,
4. Reflection: Presentation, feedback, and write-up.

In the following sections, the work conducted in these phases is described.

Knowledge Gathering Phase

The *knowledge gathering phase* was dedicated to giving students an introduction to the possibilities of AI techniques commonly used in games. This part was structured as in-class discussion seminars followed by hands-on practice in workshops, allowing students to expand their creative palette as designers. The discussion seminars were focused on different themes, including AI-Based Games, Software Studies, Interactive Narrative, Characterization and Agents, Procedurally Generated Content, Computational Creativity, and Artificial Life.

In the first seminar, students chose themes, texts and tools, which they later presented to their peers. Doing so, they became the group's experts on different approaches, the 'experts in resident' for their chosen themes.

In the workshops, students explored a range of topics and technologies related to the seminar themes, including Oculus Rift and various bio data gathering devices. The emphasis of the workshops was to provide hands-on experience that would be meaningful for both students proficient in programming, and those who were not. For example, in a workshop on interactive

narratives, all students participated in playing the card game, *Harold in Trouble* (Hoffman, Spierling and Struck 2011), which demonstrates how planning (as a computational approach) can be used for creating narratives. Then, students could choose between systems of different difficulty levels to implement a short story themselves (from paper prototyping with cards or TWINE[1] to Inform 7[2] or TADS[3]).

Figure 3. Workshops. Left: Play of card game Harold in Trouble as a way to introduce STRIPS planning in interactive story worlds. Top right: MindWave device, Bottom right: Interacting with ELIZA.

Learning was also accomplished through reading and by hands-on experimentation with technologies. In addition, subject experts were invited to give guest lectures via teleconferencing. The motivation was to give the students a range of examples illustrating what can be accomplished with different approaches. For the initial course, four guests were invited. Brian Magerko described ongoing work with Viewpoints AI (Jacob and Magerko 2015), Gillian Smith presented her work on Tanagra (Smith et al. 2010), and Richard Evans expanded upon his work on building a

1. http://twinery.org
2. http://inform7.com
3. http://www.tads.org

world of rules via the Praxis language he designed for the Versu engine. Evans also answered students' questions about the development of the AI for the creatures in the game, *Black and White* (Electronic Arts 2001), the behavior of which is determined by the players' actions, and about the AI for *The Sims 3* (The Sims Studio 2009) for which he was part of the engineering team. A central text in the course literature was *Expressive Processing* (Wardrip-Fruin 2009), as the volume describes the experiential and aesthetic effects of computational expressions, and in one of the seminars, Noah Wardrip-Fruin gave a lecture and discussed the topic (especially the SimCity effect) with the group.

Concept Phase

In the *concept phase*, students worked in groups, brainstorming and creating game prototypes. Their first task was to narrow down what they wanted to make; what type of game, what type of technology to have at its core, and what subject area or theory to use as the main ontology in the design. Early on, group members needed to agree on design goals in terms of player experience. The following questions were used as guidance:

1. What are the underlying theories or subject areas used as metaphors for the design of the game?
2. What, if any, game genre conventions are used?
3. What technologies, AI systems, or tools are used, and how could they affect the design of the game world and the game mechanics?

Development and Play Testing Phase

In the *development phase* students created digital playable games. For this they used commonly established workflows, including iterative design as described by Fullerton (2004). For play testing, students were asked to consider whether the impact of certain

domains, or ontologies, affected the game in a way that was appropriate according to the design goals. Results of play testing sessions fed into the next iteration of the prototypes.

Figure 4. Students play test each other's prototypes in the workshop.

In the following section, we describe prototypes made in the course. For each prototype, we state the starting technology, ontology and game genre used at the outset of the design process.

Examples of Prototypes made

Haiwaicode (see Figure 5) was made by Vincent Farrugia and Alan Pirotta. As starting points, they used machine learning as technology, "car traffic" as ontology, with racing as game genre. In the prototype, the cars' acceleration and deceleration was AI controlled. The player's role was to observe the car's behaviors, and inform the game if the cars were over- or underspeeding (top left corner buttons), and how severely (top right slider). Each new car that is spawned has a modified behavior depending on what the user did and other things the car observes by itself (collisions and such). The idea is that the player manages to get the cars to behave "decently" – to not collide and to not move too slow or too fast. The slider at the bottom was a rough indicator of how good / bad

the cars were doing. The play objective was to get the slider to the extreme right, which would open up new levels.

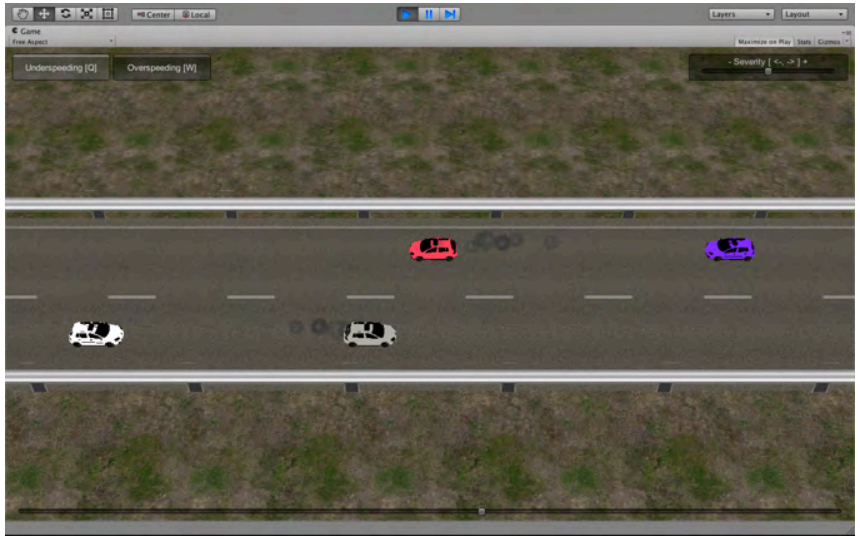

Figure 5. Haiwaicode prototype.

Compoblocks was made by Luke Aquilina and Karl Grech. The main computational approach was procedural generation, the ontology was musical composition, and the genre adopted was platformer. At the starting screen, players chose one of four moods; normal, stressed, relaxing or sad. The player controlled a ball, and the music changed depending on how the player moved the ball up or down the screen. The game experience was intended to be meditative, so there were no losing criteria. Normally, players lose when falling from a platform, but in Compoblocks, new platforms spawned under the player-controlled ball in concert with the music.

Organatron (see Figure 6) was made by Noel Cuschieri and Matthew Agius. For computational approaches they used procedural generation and genetic algorithms. Their ontology was robot wars and the game genre was strategy. In Organatron, two players could experiment with evolving dueling hybrid creatures, playing together on the same keyboard using different keys. In

the beginning, each player received five creatures that have one weakness and one strength each, represented by dots in different colors. Each turn consisted of a battle and a mutation phase, and it was in the latter that the strategy element came in, and where players picked the strengths to evolve.

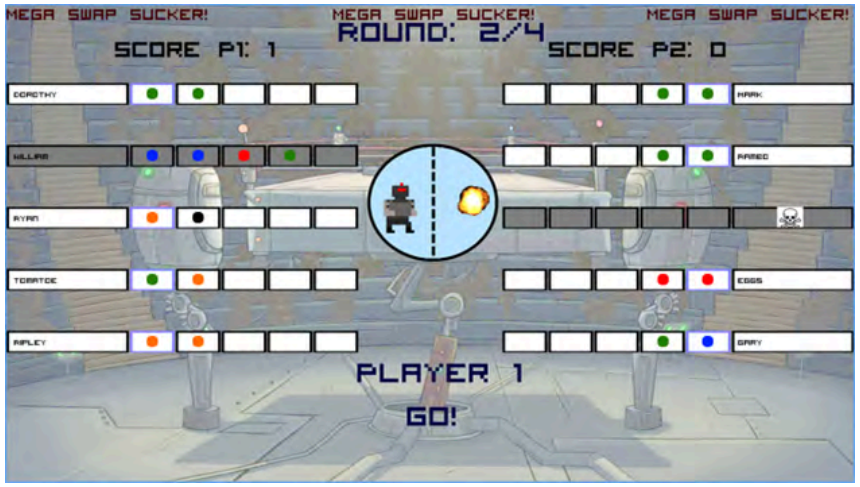

Figure 6. Organatron prototype

Dungeons & Maybe Dragons (see Figure 7) was built by Jean-Luc Portelli and Andrea Piano. As technology, they used procedural generation in combination with quest flags, adopted the common RPG ontology of dungeon crawlers, having the game genre in game mastering of RPGs. They created a hybrid digital/analog system where game masters could use mobile devices in order to author dungeons for table-top RPGs.

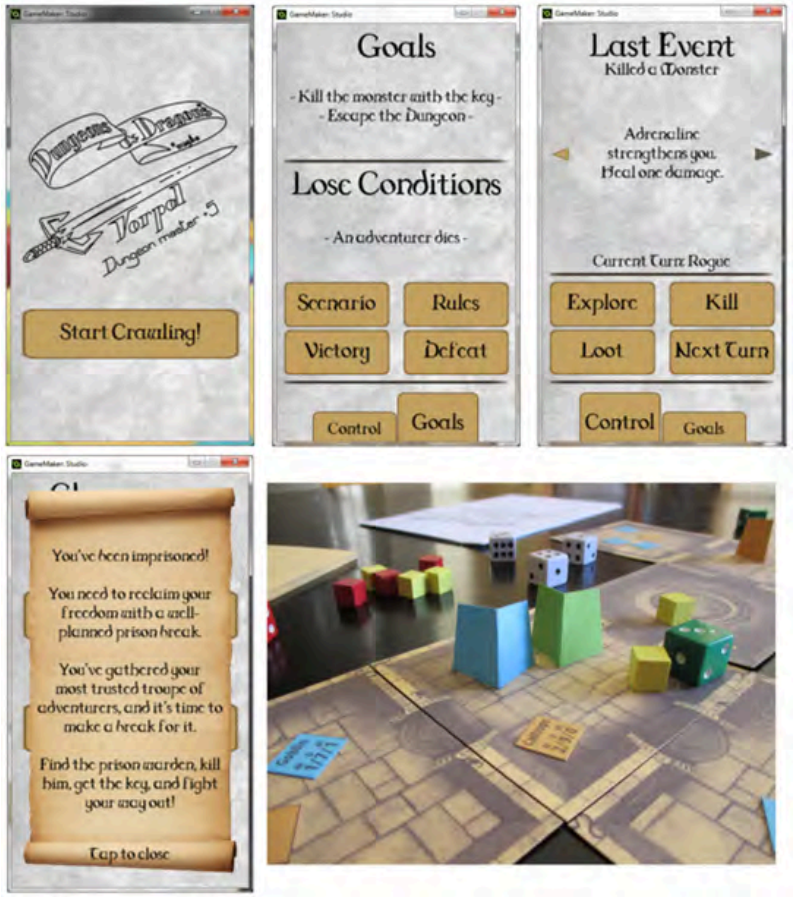

Figure 7: Dungeons and Maybe Dragons prototype

Heracles (see Figure 8) was built by Stelios Avramidis, Joseph Darmanin and Michael Camilleri. They used the functionalities of the gyro as their main technical approach, their ontological realm of choice was Greek mythology, and the game genre was shooter games. In this project, the notion of technology shifted from using computational approaches – instead it led to an exploration of the affordances of gyro functionalities. The group built a custom device (see to the left of Figure 8), a bow that was used in connection with a mobile device. The goal for the player was to shoot birds, aiming with the bow. In addition to seeing the birds on the screen, players where given audio cues to help find the birds.

Figure 8: Heracles prototype. Left: bow with gyro. Right: screen view on mobile device.

Line (see Figure 9) was built by David Chircop and Gary Hili. This was another prototype that focused on a non-standard input method: drawing on a tablet. For the ontological inspiration they used the concept of minimal art (line), and for the game genre they used the convention from *Yellowtail* (Levin 1998). *Yellowtail* repeats a user's strokes, which are received as gestures, and produce a dynamic display of textures. In Line, Chircop and Hili introduced simultaneous, competitive line input that resulted in minimalist artwork, both as a result of the interaction, and as an evolving art piece for players and spectators of the game play.

Reflection Phase: Presentation, Feedback and Write-up

In the reflection phase, students finalized their games and presented them in the seminar. We reflected on the design process, and discussed promising aspects. Students were given a date by which to halt all further development of their prototypes, in order to ensure that they had enough time to reflect on what they had achieved with their work. Finally, students authored their reports, reasoning about how the TOG triad affected their design processes and the created prototypes (see Figure 2). More concretely, students considered the following:

1. Do players' mental models reflect the designers' (your) intentions? (Mental Models)
2. Are the workings of the underlying computational systems transparent to the players? (Expressive Effect)

3. Do players assign intentionality to computational processes in the game world? (Intentionality)

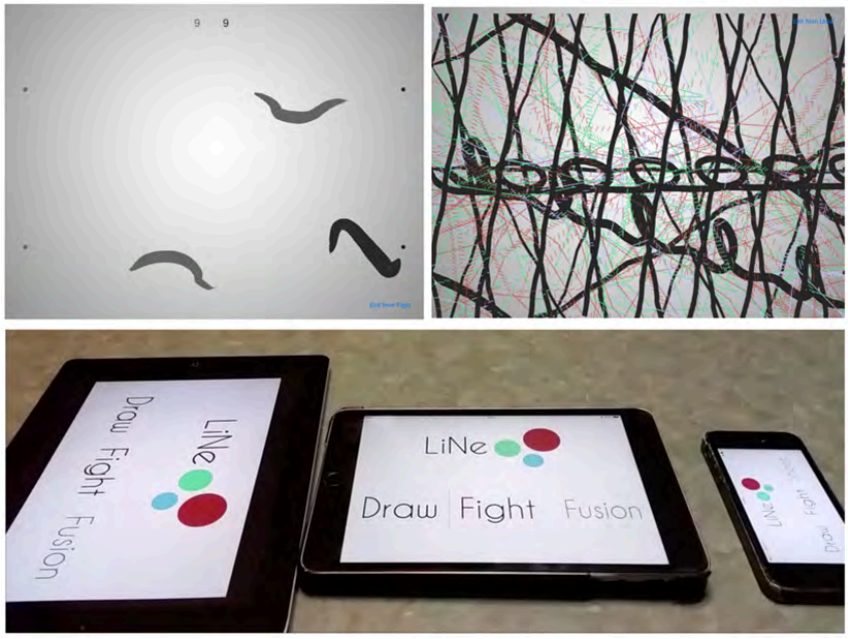

Figure 9 Line prototype.

Post Mortem

Student's reflections

After the course, students reflected on their learning process. They each answered a survey, wrote an individual piece as part of their assignments, and participated in a discussion in the last seminar. A common element in students' reflections after the course was that they found the approach useful for ideation. They also appreciated the resulting non-standard and innovative projects. Regarding learning new technological approaches, students appreciated the structure where they each were able to champion one or several approaches in seminars, and thus became the resident experts for

each other, increasing the shared knowledge of the course as a "hive-mind".

Often, students individually focused on those approaches they wanted to use or develop in their future careers. Hands-on experience in using different tools and technological approaches in the workshops was mentioned as another positive aspect, and it was also described as improving confidence in their future use. For the development process, many noted that the focus on one specific technology or AI method, along with their game design, helped to make it feasible to produce a playable prototype within the given time frame.

Output

Evaluation of the materials produced during the course shows that the student groups who had put a strong effort into studying and integrating their ontology into the game and technology design, produced the most interesting projects. This assessment is based on the play-test evaluation conducted by the students during the course. Another indication of the success of the overall approach is in the level of participation and engagement in the course, which was exceptional: In both iterations of the course all the students returned all deliverables on time, and according to instruction. Two years after the first course on computational expression, almost half of the students based their exam projects on ideas developed during this particular course.

CONCLUSION: TOG IN GAMES EDUCATION

In summary, the TOG (Technology, Ontology, and Game Genre) model is designed as an intervention that enables innovation through the challenge of unconventional combinations of technology, ontology and game genre. The use of the TOG approach in teaching game creation enables students to reflect on genre conventions and learn about particular technologies. In

addition, it facilitates the adaptation of ontologies outside the realm of established approaches for game design, and increases their productivity of concrete game designs, leading to the creation of non-standard prototypes. The concept of the techno-artistic minimum is used to emphasize the dual nature of video games as both a technological artifact and an artistic one. Both of these aspects need to come together sufficiently well in a given video game in order for it to be considered playable, and thus at least a viable prototype. Reaching the techno-artistic minimum is also a prerequisite for the reflection phase of the TOG model as a method in games education. In this final phase, students reflect on the artifacts in terms of mental models, intentionality, and expressive effect.

The TOG approach was developed during a two-year period of teaching a course on computational expression of five ETCS at the University of Malta. Course evaluation and direct feedback showed the approach to be successful. In addition, many of the students' final masters' projects were based on this course. For implementation in different educational settings, the TOG approach can be modified according to the needs and technological proficiency of the students, and to the learning goals set by the educators. Students with computer science backgrounds can reflect more deeply on aesthetic aspects in the use of technology. Students with artistic, humanistic or social science background can apply prior critical perspectives while getting hands-on experience in using various computational methods. Hence, the TOG approach can be used to increase the common understanding of the expressive opportunities afforded by the combination of technology and game design. The seminar topics mentioned in the case study serve as examples, to be adjusted by educators in accordance with their specific learning goals and available resources. The aim in implementing the TOG model as a method in games education is not to produce the perfect game for the player, but to facilitate a process in which students improve their skills as developers, find their favorite tools of trade, learn to use them with confidence, and spawn ideas that can be prototyped within the safe

space of the course. The TOG approach invites game designers to see themselves also as artists in computational expression and ideally, this experience will lead to further experimentation and innovation in their future academic work and industrial careers.

BIBLIOGRAPHY

Agre, P.E. 1997. Computation and Human Experience (Learning in Doing: Social, Cognitive and Computational Perspectives). Cambridge University Press.

Bartle, R., 2003. *Designing Virtual Worlds*. New Riders.

Bolter, Jay. 2019. *The Digital Plenitude: The Decline of Elite Culture and the Rise of New Media.* The MIT Press. ISBN: 978-0-262-03973-4.

Coleridge, S.T., 1894. Biographia Literaria; Or, Biographical Sketches of My Literary Life and Opinions; and Two Lay Sermons, London: George Bell and Sons.

Dennet, D. 1987. The Intentional Stance. MIT Press.

Eladhari, M P. 2015. "AI-Based Game Design." In *GDC Education Summit at the Game Developers Conference.* San Francisco, CA. http://gdcvault.com/play/102177 9/AI-Based-Game.

Eladhari, Mirjam Palosaari, A. Sullivan, G. Smith, and J McCoy. 2011. *AI-Based Game Design: Enabling New Playable Experiences*, technical report. University of California Santa Cruz. https://www.soe.ucsc.edu/research/technical-reports/ ucsc-soe-11-27.

Electronic Arts. 2001. *Black & White*. Lionhead Studios [Computer Game].

Fullerton, T. et al. 2004. *Game Design Workshop: Designing, Prototyping, and Playtesting Games*. CMP Books.

Hoffmann, S., Spierling, U. and Struck, G., 2011. A practical approach to introduce story designers to planning. *Proceedings of GET*, pp.22–24.

Hunicke, R., LeBlanc, M. and Zubek, R., 2004. *MDA: A Formal Approach to Game Design and Game Research*. [online] AAAI Press. Available at: <http://cs.northwestern.edu/ hunicke/pubs/ MDA.pdf>.

Horswill, Ian, Michael Cook, Mirjam P. Eladhari, Alex Zook, Adam J. Summerville, Ben Samuel, and James Ryan. 2019. "Game Design is a Liberal Art." In *Tenure and Promotion Workshop, Foundations of Digital Games Conference.* San Luis Obispo, California, USA, August.

Jacob, Mikhail, and Brian Magerko. 2015. "Viewpoints AI" [in en]. In *Proceedings of the 2015 ACM SIGCHI Conference on Creativity and Cognition – C&C '15*, 361–362. Glasgow, United Kingdom: ACM Press. ISBN: 978-1-4503-3598-0. http://dl.acm.org/citation.cfm?doid=2757226.2757400.

Levin, G., 1998. *Yellowtail*. Interval Research Corporation.

Loyall, A.B. and Bates, J., 1997. Believable Agents: Building Interactive Personalities Thesis Committee: (May).

Mateas, M. and Wardrip-Fruin, N., 2009. Defining Operational Logics. In: B. Atkins, H. Kennedy and T. Krzywinska, eds. *Breaking New Ground: Innovation in Games, Play, Practice and Theory.* DiGRA Conference. Brunel University, UK. Available at: <http://www.digra.org/dl/db/09287.21197.pdf>.

Meehan, J.R., 1977, August. TALE-SPIN, An Interactive Program that Writes Stories. In *Ijcai* (Vol. 77, pp. 91-98).

Murray, J.H., 1997. *Hamlet on the Holodeck*. The Free Press.

Osborn, J.C., Wardrip-Fruin, N. and Mateas, M., 2017. Refining operational logics. In: *Proceedings of the International Conference on the Foundations of Digital Games – FDG '17*. [online] the International Conference. Hyannis, Massachusetts: ACM Press.pp.1–10. Available at: <http://dl.acm.org/citation.cfm?doid=3102071.3102107> [Accessed 26 Jun. 2020].

Puerta-Melguizo, M.C., Chisalita, C. and Van der Veer, G.C., 2002. Assessing users mental models in designing complex systems. In: *2002 IEEE International Conference on Systems, Man and Cybernetics.*

Rao, A. S., and M.P Georgeff. 1995. "BDI-agents: From Theory to Practice." In *Proceedings of the First International Conference on Multiagent Systems (ICMAS'95).* https:// www.aaai.org/Papers/ICMAS/1995/ICMAS95-042.pdf.

Sengers, P. 2000. Narrative Intelligence. K. Dautenhahn, ed. John Benjamins Publishing Company.

Schell, J., 2008. *The Art of Game Design: A book of lenses*. Morgan Kaufmann.

Smith, G., Whitehead, J. and Mateas, M. 2010. "Tanagra: a mixed-initiative level design tool" [in en]. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games – FDG '10*, 209–216. Monterey, California: ACM Press. ISBN: 978-1-60558-937-4. http://portal.acm.org/citation.cfm?doid=1822348.1822376.

The Sims Studio. 2009. *The Sims 3*. Electronic Arts [Computer Game].

Walk, W., Görlich, D. and Barrett, M., 2017. Design, Dynamics, Experience (DDE): An Advancement of the MDA Framework for

Game Design. In: O. Korn and N. Lee, eds. *Game Dynamics*. Springer International Publishing. pp.27–45.

Wardrip-Fruin, Noah. 2009. *Expressive Processing: Digital Fictions, Computer Games, and Software Studies.* The MIT Press. ISBN: 978-0-262-01343-7.

Weizenbaum, J., 1966. ELIZA—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1), pp.36–45.

Winn, B.M., 2009. The design, play, and experience framework. In: *Handbook of research on effective electronic gaming in education*. IGI Global.pp.1010–1024.

Wright, W. 1989. *SimCity.* Maxis [Computer Game].