Fluidic Games in Cultural Contexts

Mark J. Nelson, Swen E. Gaudl, Simon Colton, Edward J. Powley,
Blanca Pérez Ferrer, Rob Saunders, Peter Ivey, Michael Cook
The MetaMakers Institute
Falmouth University
metamakersinstitute.com

Abstract

We introduce fluidic games, a type of casual creator that blends game play and game design. Fluidic games have a core of built-in games that anchor a space of design possibilities around them, and encourage players to alternate between playing specific games and playing with the design space. Our Gamika Technology platform supports fluidic games on mobile devices, and we have thus far built three of them. In doing so, we have found that even for simple games, fluidic games require computational creativity support. This takes several forms intended to keep design sessions playful and fast-moving, including automated game design used as a form of brainstorming, mixed-initiative co-creative design to ease design-space navigation, and automated game playing to evaluate game dynamics. Finally, we have exhibited this fluidic-games concept in three distinct cultural settings: a series of rapid game jams lasting 1–2 hours each, an in-progress semester-long enrichment course with a local school, and an art installation that foregrounds an autonomous version of the system exploring a fluidic game on its own, at least if the audience will allow it to do so.

Introduction

Fluidic games are initially just games, playable as any other game. But in contrast to games that emphasise a single, carefully designed artefact, fluidic games emphasise that any individual game is always only a single point in a larger game-design space, from which many other games could also have been made, from trivial variants to significantly different games. Players are encouraged to explore this space with minimal context shift between the playing and design-exploration modes.

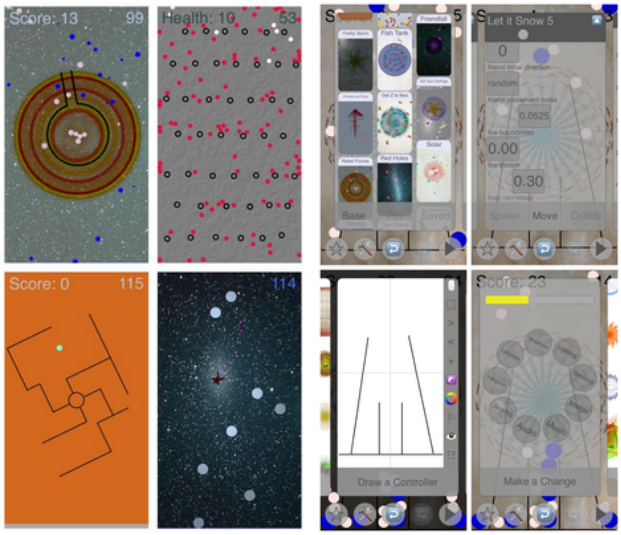
A focus of our research agenda is investigating different approaches to this exploration, which can be viewed as falling along the spectrum of mixed-initiative human/machine co-creativity in games (Smith, Whitehead, and Mateas 2011; Grace and Maher 2014; Yannakakis, Liapis, and Alexopoulos 2014; Liapis, Smith, and Shaker 2016; Nelson et al. 2017). At one end of the spectrum of mixed-initiative creativity is an orientation towards enabling human creativity (Shneiderman 2007); at the other is fully autonomous game creation (Cook, Colton, and Gow 2016).

We focus on casual games played on mobile devices, which many people play, but few design. We aim to help to democratise this situation by making the player/designer boundaries more fluid, so players can play individual games and also play with the design itself, within the same app and with frequent alternation between the two modes. In addition to minimal context shift from playing to designing, we also aim to have a difference in time commitment: users playing an iPhone game on the bus ride home should be able to spend 20 minutes designing a variation of that game on their iPhone, too, and then go back to playing their new game. Or, they might want to press a button and have an AI designer generate a new game—we have found that even when oriented towards human design, some degree of automated design is desirable to make navigating the space playful rather than tedious. A fluidic game is not just a game to play, but neither is it a traditional game-design tool.

Fluidic games therefore fit into the larger category of accessible, low-commitment, fun-to-use creative tools dubbed casual creators (Compton and Mateas 2015). They also fit into a broader class of user-modifiable games, which we'll call *maker-games*, which give players the ability to change some aspect of the game, most commonly by including a level editor (as seen in Nintendo's *Super Mario Maker*).

Mobile games are an especially good setting for casual creators, both because they are widely played even by people who don't necessarily see themselves as "creators", and because games foreground concepts such as initiative and agency that allow creative game design tools to piggyback on familiar game-playing terminology and concepts. Games also pose a challenge by integrating many creative design domains, from systems thinking to storytelling to visual aesthetics (Liapis, Yannakakis, and Togelius 2014).

We have piloted the fluidic-games concept in three distinct cultural settings, in addition to planning the public release of two fluidic games, *Wevva* and *No Second Chance*, on the iOS App Store. We have used both *Wevva* and *No Second Chance* to host rapid game jams, a version of a game jam in which players can make their own games in as little as 10 minutes, with the overall jam lasting no more than 1–2 hours (traditional game jams typically last 24–72 hours). We are currently engaged in a more extended educational experiment using *No Second Chance* to teach game-design and elementary physics principles to students in a lo-



(a) (b)

Figure 1: (a) Four Gamika games designed with (b) *Cillr*. *Cillr* design panels clockwise from top left: List of editable saved games, a screen of movement-related sliders, brainstorming wheel to randomise subsets of parameters, and drawing interface to edit controllers.

cal school. And finally, we designed and exhibited an artistic installation called *I Create, You Destroy*, based on a fully autonomous version of *No Second Chance* in which an autonomous creative system designs and plays its own games, and human participants (if they choose to participate) play only a destructive role in the creative process.

Gamika Technology

To enable the design of fluidic games, we have built a platform, Gamika Technology, that parameterises a game-design space with 284 parameters, plus associated visual and audio assets. Parametric design is not the only technique for building fluidic games, but one we think is well suited to the task, as it casts the problem of design-space navigation in a concrete setting suited to both user-interface design and automated exploration. On this platform, we have built three fluidic games. One, *Cillr*, is based on the entire space and is used primarily as an in-house app to explore the design space in full generality; while initially intended as a fluidic game, it may be seen as closer to a design tool, as discussed below. Two others, *Wevva* and *No Second Chance*, are more focused fluidic games in specific genres, soon to be publicly released on the iOS App Store.

The basis of the Gamika Technology platform is a 2D game engine parameterised by 284 features that we have identified as core to a diverse range of casual games. This set includes parameters controlling the physics engine, player interactions and scoring/win-conditions. Physics parameters expose common features of a 2D physics engine: object spawn rates/locations, collision responses, attractive/repulsive forces, etc. Interaction parameters specify how players interact with the physics world, such as when and how objects respond to the player tapping or dragging on the screen. Scoring and win-condition parameters specify how events impact the game outcome (the more narrowly conceived rules of the game). A more detailed parameter overview is given in (Powley et al. 2016, Section III).

Games for the Gamika platform are encoded in parameter chromosomes; the term is borrowed from evolutionary algorithms, as automated game generation is a part of each fluidic game. The chromosomes are augmented with data such as graphical and sound assets. Given a chromosome, the Gamika platform can run the game via an interpreter that allows runtime changes to the game specifications. Figure 1a shows four example Gamika games, each designed using the *Cillr* app, described in the next section.

Apps

Cillr

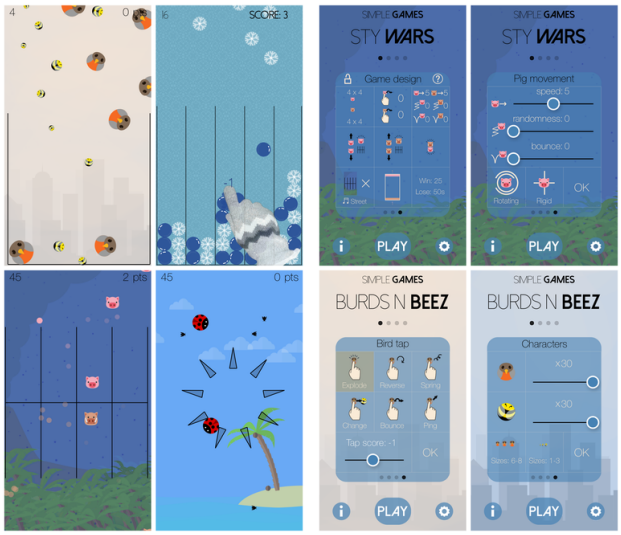
The first app built on the Gamika Technology platform is called *Cillr*, which enables navigation of the entire Gamika design space. Although this can be seen as a type of fluidic game, due to the size of the design space and relatively unfocused nature of the app, we use it primarily as an in-house design tool, from which we have drawn lessons used to build the more focused fluidic games discussed in the subsequent two sections, which are intended for public consumption.

Cillr implements baseline versions of both manual and automated navigation of a parametric design space. The most direct way of manually navigating the 284-dimensional design space is to give the user 284 sliders, with which they can set each parameter. While this approach – implemented in *Cillr* – is simplistic, it does work fairly effectively. The sliders are grouped into categories with related functionalities to make them more discoverable (the spawning-related sliders are collated, the collision-related sliders likewise, etc.). A few panels of the app are shown in Figure 1b.

The simplest way of automatically navigating a large parameter space is to randomise the parameters. However, we have found that this produces too low a yield of playable games, and hence *Cillr* mutates subsets of parameters from existing games instead. Randomly mutating multiple sets to produce a new random game, and then trying to figure out what it is, can be a fun interaction loop. If the user isn't interested in understanding and exploring the entire design space, however, the proportion of playable games remains too low for the mutation approach in *Cillr* to be ready for end-user consumption.

Besides producing Gamika chromosomes (both manually and with randomisation), *Cillr* includes editing tools for graphical elements such as sprites, level layout, and lighting, so complete games can be produced, including games with level progressions and multiple levels of difficulty. We have used the interface to produce clones of classic games like frogger, asteroids and space invaders, as well as a variety of novel casual games; a narrated set of design sessions is reported in (Colton et al. 2016).

As an initial baseline, *Cillr* is usable, at least by experts, though it does not yet contain interesting levels of automated



(a) (b)

Figure 2: (a) Four *Wevva* games produced with the in-app design interface (b) which is described in the text.

game design. Its main drawback is that it is complicated to navigate, requiring some time to hunt for the correct slider to change to make something specific happen. Furthermore, even after having found the desired parameter, it can be difficult to understand why the game didn't change as expected.

In a preliminary user test with game-design undergraduate students, we found them somewhat frustrated by the experience of using *Cillr* to make games. Interface complexity was one issue, but more importantly, the difficulty of understanding the high-dimensional design space made it hard for these initial testers to grasp what they wanted to do in the app, and how they would begin to do it. Therefore, rather than focus initially on improving *Cillr*'s interface or including more automated co-design elements, we have instead focused on producing design tools for more cohesive, lower-dimensional design subspaces of the Gamika Technology platform that do not expose the entire design space at once. The first two of these are discussed below.

Wevva

Using *Cillr*, we made a four-in-a-row game called *Let It Snow*, where snow and rain pour down from the top of the screen (as white and blue balls respectively). When four or more white balls cluster together, they explode and the player gains a point for each in the cluster. Each white ball that explodes is replaced by a new one spawned at the top, with a maximum of 20 on screen at any one time. Likewise with blue balls, except the player loses one point for each that explodes. Players can interact with the game by tapping blue balls to explode them, losing a point in doing so.

While the game rules are straightforward, we have found *Let it Snow* to be challenging and require puzzle-solving strategies. There is a grid structure which collates the balls into bins, and the best way to play the game involves trapping the blue balls in groups of twos and threes at the bottom, while the whites are exposed and are continually refreshed through cluster explosions. Occasionally, when all blues are trapped in small clusters, only whites will spawn, which is akin to snowing (hence the games name) and is a particularly pleasing moment to aim for.

We used *Cillr* to produce a number of variations of *Let It Snow*, initially also with the winter precipitation theme, but since expanded to include multiple settings and seasons, as well as characters such as pigs, bees and frogs. The latter were added because feedback from playtesters in our rapid game jams (described later in this paper), who had used an early version of the app lacking living characters, found it difficult to invent narratives explaining what each game was about. This expanded design space will be released as an iOS game entitled *Wevva* (Figure 2).

This app further includes two aspects that are not common in casual games: (a) an AI player that can assist novice players, and (b) a design screen enabling players to edit the games' mechanics, as well to generate levels in a semi-random way as a source of inspiration. In *Let It Snow*, the AI player appears on-screen as a gloved hand that taps the blue balls to keep clusters of four from forming (Figure 2a, top right), implementing one part of a winning strategy. A slider lets the player change the level of AI assistance. At 50%, it feels like having an in-game partner helping out. At 100%, the game is quite different, as the AI player takes care of one aspect of the game (avoiding losing points), freeing the player to concentrate on gaining points.

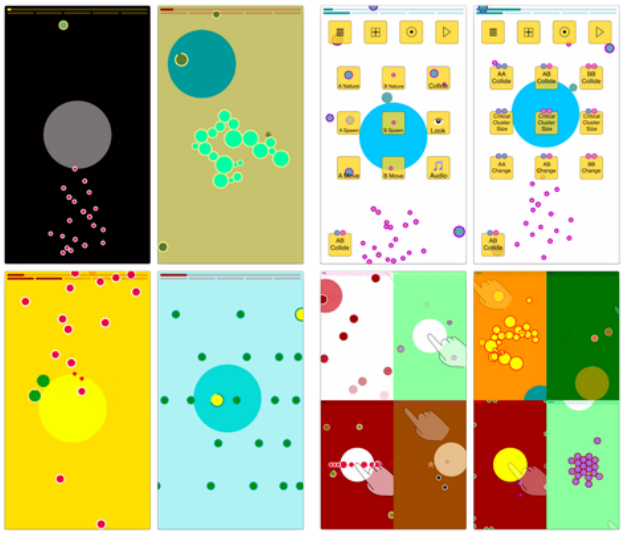
The design screen (Figure 2b) exposes many elements of the game design to the player: (a) what happens when the player taps on a sprite, such as exploding, changing direction or transforming into another kind of sprite (b) the shape, size and control scheme for the controller or grid (c) the sprites that exist in the game (d) scoring attached to events such as sprites exploding, being tapped, hitting screen edges or forming clusters (e) sprites' spawning locations, speeds, and limits (f) sizes of sprites (g) physics parameters, namely bounciness, wind strength and initial speeds (h) win/loss parameters, namely a time limit and score target (i) background art setting the location and (j) music selection.

There is an inspiration button designed as a brainstorming assistant, which will set these parameters in a varied way, but designed so that the clustering score mechanic is balanced in terms of expected score. We achieved this by running online simulations of novice players and recording the number of times that clusters of each size and type occurred. Finally, there is a *clean slate* feature, which resets parameters to a standard starting point.

We have conducted a series of "rapid game jams" using *Wevva*, described in the Cultural Contexts section below. These have helped to promote mixed-initiative creativity in fluidic games through events, and to refine their concept and design by observing what people do with fluidic game apps.

No Second Chance

Again using *Cillr*, we designed a game of patience and concentration, *Pendulands*. Here, balls move in a pendulum-type motion and annihilate each other if they collide; the



(a) (b)

Figure 3: (a) Four *No Second Chance* games produced with (b) the in-app design interface. The top two design-interface panels show manual navigation of the space of parameters, and the bottom two show automatic game generation with split-screen auto-playtesting.

player must catch five of them by hovering under them with a large round target until they stick. By varying parameters within this theme, we discovered that a whole set of *Pendulands* variants (or levels) can be created. The anchor points defining this sub-space of Gamika games are: the player always controls the target by dragging and must catch five balls on the target. Within these parameters, very different games can be created, both in terms of their game dynamics and in terms of the types of challenges they pose.

No Second Chance is our third app, built around this space of games, a few of which are shown in Figure 3a. The name comes from a meta-game mechanic: players can send games to each other in such a way that they are deleted if the receiver doesn't beat the game on first playing (in five minutes). This emphasises the disposable nature of games in a generative space, where part of the challenge is exploring the space of games and figuring out how each one works when first encountering it.

As with *Wevva*, a design screen (Figure 3b, top) lets players make new *No Second Chance* games. It is laid out as a hierarchical menu, with submenus allowing visual style and a variety of physics parameters to be changed. Since what is fixed about *No Second Chance* games is the control and scoring mechanism, new games are made by varying physics, spawning and scoring options, which can produce very different game dynamics and mechanics. To demonstrate the types of games that can be produced (and to provide an initial challenge), the app comes with 100 games we designed using this interface, which we've categorised into three primary types of challenges: skill games, where the primary challenge is dexterity; ingenuity games, where the primary challenge is figuring out a specific trick or strategy; and patience games, which involve waiting for the right situation to arise and capitalising on it accordingly.

The generation button creates a new game via an evolutionary process. In particular, meaningful blocks from existing games' chromosomes are crossed over, randomly mutated, and then filtered using static heuristics to reject clearly bad candidates. The first four candidates that pass the filter are auto-playtested on the device in a split-screen view (Figure 3b, bottom) that plays them at 8x speed for 5 seconds, the equivalent of 40 seconds of game time. We want games to be playable but not too easy, so the app chooses the game that the automated playtester was able to catch the most balls on, without being able to catch all five.

The split-screen visualisation of playtesting isn't strictly necessary; games could simply be silently generated and given to the user, which would be computationally cheaper as well. But this kind of "Hollywood AI" visually externalises to users what the apps' AI components are doing. It can also be entertaining in itself to watch the generation process, as new games are created and then played at rapid speed by the automatic playtester.

The term *Hollywood AI* comes from the frequent use in films of flashy computer interface mock-ups, which we use as a reference point. These are designed to look glitzy and active and to convey an idea of what the system is doing. While sometimes derided by technologists for not bearing much resemblance to real computers, the entertainment and progress-externalisation aspects of imagined film interfaces can be usefully adapted in real designs (Shedroff and Noessel 2012). They translate especially well to computational creativity systems, where it is key for the AI software to communicate when it takes the initiative and what it is doing, in this setting ideally through the use of readable visual conventions, e.g., the usage of an on-screen hand painting pictures with The Painting Fool (Colton and Ventura 2014).

Applications in Cultural Contexts

Rapid game jams

Game jams are events in which small teams make a game in a much shorter period of time than in traditional commercial game development. Typically lasting between 24 and 72 hours, these events bring some of the community-building and culture of LAN parties, in which players *play* games (Taylor and Witkowski 2010), to the process of game development. This produces "accelerated, constrained and opportunistic game creation events with public exposure" (Kultima 2015). They have had a large cultural impact on the indie game community, and have helped in developing a more experimental and inclusive game-development scene (Westecott 2013; Bonaiuto et al. 2014).

We aim to capture these positive aspects of game jams, but at a much shorter timescale, and with more focus on design experimentation and less on implementation. Despite being much less rigid than traditional game development, game jams still largely resemble a software hack-session, in which teams spend several days implementing an initial idea as a prototype (Musil et al. 2010). Researchers have found

that teams tend to start with an idea that remains largely intact (even if scaled down or modified where it turns out to be infeasible), with most of the time spent implementing a working prototype of that idea, rather than more free-form design experimentation (Zook and Riedl 2013).

We would like to foster game jams that emphasize design experimentation to a greater extent, with radically cut down time commitments. While making a game in a weekend is already much less of a commitment than forming a company and spending months on it, it is far more of a commiment than playing a casual game during spare time, which is our reference point for the context in which fluidic games should be both playable and designable.

Rapid game jams, lasting from as little as ten minutes up to a maximum of two hours, fit this role. In their shortest form, we have held in-office ten-minute game jams with people already generally familiar with the apps, to test and improve the ability to support this kind of tightly constrained creative design. We have also held longer 90-minute sessions (described below) with several large groups of children who had never used the apps before. This allows some time for participants to initially play the fluidic games' built-in games to orient themselves in the design space, followed by exploring new designs through the automatic generator and/or the design interface.

The first large-scale rapid game jam we held with *Wevva* was with 65 members of Girlguiding Cornwall's Brownie programme (i.e., girls aged 5-9), who visited Falmouth University as part of a larger Girls Can Code event on 18th February 2017. This was conducted in two sessions, with 35 users in the first and 30 in the second. Users were paired up with typically two (occasionally three) users per iPad. We began by asking them to play the included games for ten minutes. Then, we provided a brief introduction to the design interface and gave them about an hour to design their own games. This was followed by a period where they shared their games with other participants. Concluding the sessions we handed out a feedback form which contained a set of questions about their experience with the app.

The playtest of the four built-in games produced largely negative results. These games are puzzle-oriented, requiring the player to be patient and come up with a winning strategy; but here, very few were able to discover a winning strategy. The design side of the user test was more successful, however, as the children proved adept at using the built-in design tools to design other types of games in this space of games, which they preferred to the built-in games. The games they designed were generally more action-oriented, where tapping quickly on things is the winning strategy, and where there is more instant feedback about when the player took a good or bad action.

We conducted two further rapid game jams with 40 members of Girlguiding Cornwall's Guides programme (i.e., girls aged 10-14), who visited Falmouth University on 23rd February 2017, also as part of the Girls Can Code event. We used the same approach and structure as in the first game jams, starting with an introduction to the included games and a ten-minute game playing session to familiarise them with the game, its controls and mechanics. In contrast to the younger participants from the first two sessions, the older participants spent more time playing the four included games. They also approached the four games more closely to our expectations, probing and trying out different strategies. Repeating the structure, we gave them an initial introduction to the design space, and, as with the first two groups, the possibility to explore the design space by creating and sharing their own games. Similar to the younger groups, we also concluded the sessions by handing out our feedback form which contained questions about their experience with the app. Our first observations of their exploration of the game space showed that the designed games sometimes still focused on fast tapping game mechanics, but we also saw a wider range of games which required more sophisticated strategies.

Mechanic	Used as primary	Used at all
Herding to collect	21	39
Tap-em-up	13	42
Keeping separate	12	30
Catching to collect	8	10
Batting away	7	17
Spawning flow	6	13
Toy-like	3	10
Protect sprites	1	7
Protect zones	1	6
Steady hand	0	2
Fast reaction	0	1

Table 1: Classification of the game mechanics used in the 72 games saved by participants in the Girlguiding Cornwall rapid game jams.

During each of these rapid game jams, we encouraged participants to save games they liked and share them with others. At the end of the sessions, we collected the saved games, totalling to 72, for analysis. Table 1 presents one way to get an overview of the design space explored in the game jams by grouping the games according to the game mechanics they use. We labeled each game with the primary game mechanic it makes use of and one or more secondary mechanics. The game mechanics we identified are:

- Herding to collect: group together clusters of sprites.
- Tap-em-up: tap as rapidly as possible on certain sprites.
- Keeping separate: keep some sprite types apart.
- Catching to collect: catch sprites with the controller.
- Batting away: knock some sprite types off the screen.
- Spawning flow: try to manipulate spawning patterns.
- Toy-like: focus on enjoyable interaction, not scoring.
- Protect sprites: keep a sprite type from exploding.
- Protect zones: keep sprites from going off screen at certain places.
- Steady hand: make careful movements, e.g. to thread through a narrow gap.
- Fast reaction: make rapid, precise movements or taps.

As can be seen in Table 1 the *herding to collect* and *tap-em-up* mechanics featured in some form in more than half of the games. This is not surprising, since scoring by forming clusters and scoring by tapping are two of the more straightforward options in the design space. The least common mechanics, seen in only three games and not in any case as the primary mechanic, were those based on skill in controlling the controller or sprites, whether the steady-hand or fast-reaction kind of skill. We had designed a number of games using these mechanics in our own 10-minute game jams, so that was an interesting difference to notice.

One aspect of automation our current apps do not have is automatic fix-up of games to balance them and avoid exploits, although we have done research on a version of automatic tweaking that runs server-side (Powley et al. 2016). To see whether such a feature would be important to add, we classified the 72 games according to whether we, as expert players, were able to quickly find an easy exploit in the game design. We were able to do so in 31 of the games. Of these, the two most common exploits were being able to win by indiscriminately tapping (seen in 22 games) and being able to win by doing nothing at all for a short period of time and win (found in 7 games). On the other hand, since *tap-em-up* games were one of the two most common mechanics used, it's not clear that winning by indiscriminate tapping would actually be considered an exploit by the designers. We observed, for example, some pairs of users sharing an iPad taking turns playing a very easy game requiring rapid tapping, but competing to beat each others' best scores.

Besides this analysis of game mechanics and exploits, we asked the game-jam participants to fill out a survey about their experiences. We have performed a preliminary analysis of these survey results, for 30 girls of average age 12, responding to survey questions quantitatively using the visual analog scale. While a full user-study analysis is beyond the scope of this paper (which focuses on cultural applications of fluidic games), we have found a few interesting results so far. The following are four statistically significant correlations we observed:

- Positive: Between using the inspiration button and finding games produced by the inspiration button useful.
- Negative: Between interest in a career in game design and using the clean-slate restart.
- Positive: Between interest in a career in game design and enjoying using the app.
- Positive: Between playing a lot of games and enjoying using the app, as well as feeling more creative.

Wevva therefore seems to attract people who tend to frequently play games; those frequent players also feel more creative using the app than novice users. Those who are interested in becoming game designers used the clean slate less as they seem to feel that the games they produce from that point in the fluidic space are less interesting. Those not interested in a career in game design explore more around the clean slate, perhaps because it gives a familiar starting point for exploring the game space.

Classroom usage

Through a collaboration with a local school, the Camborne Science and International Academy, we have been developing fluidic games into a curriculum suitable for use in classrooms. The first version of this curriculum, currently in progress, teaches game design through weekly sessions of 90 minutes each, designed around experimentation in *No Second Chance*. Each week's lesson introduces a new aspect of game design, and most lessons also use that element of game design as a hook through which to teach material from another relevant subject. For example, the lesson on game visual aesthetics introduces students to colour theory, the lesson on object movement and collision response also teaches elementary physics, and the lesson on the included automated game generator introduces students to artificial intelligence and Computational Creativity.

Use game design to organise a technology-based curriculum is similar in some respects to curricula that introduce programming in schools through visual programming languages such as Scratch (Resnick et al. 2009) or its tablet version ScratchJr (Strawhacker et al. 2015), which often use games as the motivating example that shows students what can be done if one learns to code. While we draw inspiration from these projects, our goal is to focus less on teaching *programming* specifically, and more on teaching *design* in a computational setting, emphasising that programming, while an important skill, is not the only aspect of game design, nor of computational thinking more generally. The lessened focus on coding as the specific skill to teach also frees up a larger part of the curriculum to focus on the connections to other fields, such as the colour-theory and physics examples mentioned above.

Since we are currently part of the way through the initial pilot of this curriculum based on fluidic games, we report only preliminary observations. Game design in *No Second Chance* is essentially exploring a physics-based game space through parameters that produce new types of gameplay dynamics, so the curriculum is based around introducing parts of the parameter space each week, explaining what these parameters mean, how to design in that space, and how the new set of parameters impacts game design. The exercises were designed in a way to incrementally build up knowledge about the physics and parameter space of *No Second Chance*, starting with basic control over when and where objects appear on screen to how to use more sophisticated combinations of parameters to balance designed games. Each lesson sheet contains a set of parameters explained in detail with examples to guide further exploration. As proposed by (Resnick 2004), the usage of games should still be playful, so instead of giving them a fixed set of provable exercises, each is focused on open-ended exploration of the design space, with a soft peer evaluation/assessment based around sharing and critiquing each others' games (an activity that also, through playing others' games, helps students notice aspects of the design space they may have missed).

Structuring an introduction to *No Second Chance* as a series of lessons also helped us to better understand its design space as a fluidic game. To aid the students with their exploration of the game space and at the same time teach them

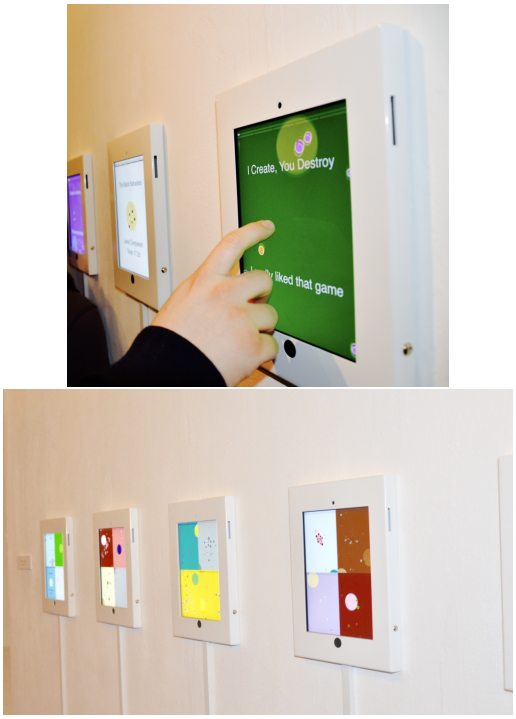


Figure 4: The *I Create, You Destroy* interactive installation. Top: An audience member destroys a generated game just as the AI was playing it. Bottom: An array of iPads happily generates and automatically playtests new games.

a further lesson each week (such as elementary physics), forced us to critically think about sets of parameters and their relation to each other and how this can be used to conceive of the parameter space in a more structured manner. While observing students' exploration, we also became aware of potentially new and interesting areas of the design space we have not taken into account yet. The amount of interest we saw during those sessions both motivated us and demonstrated the potential of employing more explorative and creative games into teaching and learning.

An art installation

I Create, You Destroy is an installation of six iPads created by @ThoseMetaMakers, the art collective alter ego of The MetaMakers Institute. This first exhibition by @ThoseMetaMakers was shown at the first Games As Arts / Arts As Games festival, which was held at The Poly in Falmouth, Cornwall, from the 12th to 22nd October 2016.[1]

In this work, an autonomous version of *No Second Chance* creates, plays, critiques, selects, and shares abstract artistic games. It happily makes and plays games all day long, alternating between generating games, using the split-screen rapid-fire auto-playtesting of the regular *No Second Chance* app, followed by playing the generated game at normal speed. It continues in this generate-and-play loop unless a visitor touches one of the screens, in which case the game that was being made or played is destroyed forever, and the visitor is told this (Figure 4). Referencing and challenging the long-standing but highly topical worries that machines will take jobs and pose existential threats, *I Create, You Destroy* is an artificial intelligence system, but it is not the bad guy in this case.

In aiming to challenge mainstream assumptions about AI being a threat, the concept of the piece resides in the balance produced by using a mainstream platform (iPad) and a mainstream genre (casual games) to challenge this mainstream assumption. The AI hides behind the bold, sharp and colourful surface of six iPads fixed to the white background wall. With white cases and white background, only the playful images of Gamika stand out. The small balls move fast, appearing from everywhere, going anywhere. New games are created directly in front of visitors, who are faced with a choice to make: either merely watch, or act, interact and then themselves become a threat to creativity.

Breaking the assumption and questioning what we are expected to do is the core of *I Create, You Destroy*, as the audience is presented with a tablet, which they are normally expected to touch, running a a game, which they are normally supposed to play. However, this is not what they should do in this case; in moral terms the audience acts destructively if they fulfill the usual expectations of interaction.

The exhibition included works from a number of other artists working with games as a medium, such as Alan Meades, Ian Gouldstone and Oliver Sutherland; as well as artwork from games such as *Lumino City* (State of Play) and *Machinarium* (Amanita Design). One point of connection, where *I Create, You Destroy* meets Ian Gouldstone's piece *Cruise Control 2020* and Oliver Sutherland's *Untitled (Loosing it)*, is the idea of the continual moving image. In the tradition of art built on game technologies (Bittanti and Quaranta 2009; Sharp 2012), these works play with game tropes, but are not presented as games the viewer themselves can play. And yet the spectator is not merely watching a pre-recorded movie, when the creative process is happening right there. What is going on will never occur again. We could consider this uniqueness one of the holy quests of contemporary art and Computational Creativity. Since we lost the unique nature of the artefact, we seek the exceptional character of the experience. Games technologies and AI enable the rise of a new hybrid art form between installation and performance with something sculptural and cinematographic about it.

Conclusion

We have introduced and investigated fluidic games, which are casual creator apps for hand-held devices. Fluidic games are in the genre of *maker-games*, games that can be modified by users in various ways. However, with fluidic games, users can not only design levels or skin games, as is common in other maker-games, but can change underlying game mechanics such as the physics, scoring mechanisms, player input, and how objects interact.

In contrast with game-creation environments such as Scratch Junior, game design in fluidic games can be achieved without any coding requirements. Indeed, we have designed

[1] `metamakersinstitute.com/gamesasarts`

our fluidic games to have fun, efficient user interfaces, so that the line between making and playing a game is fairly blurred. To achieve this has required some element of Computational Creativity support in the app, and we have described how automatic game generation and auto-playtesting have enabled users to search larger spaces of the game space, with less frustration than they would do without these tools.

Since game-playing and game-making are always situated in cultural contexts, we have been experimenting with how fluidic games fit in three cultural contexts, in order to drive both our design and technology development through real-world experience. Our three pilot settings are: 1) using fluidic games to host rapid game jams, which last no more than 1-2 hours and focus more on exploring design possibilities than game implementation, 2) integrating fluidic games into a school curriculum in order to both teach game design, and use game design as a hook with which to introduce concepts such as colour theory, physics, and artificial intelligence, and 3) adapting an autonomous version of fluidic games as an art installation, with an AI agent both designing and playing the games, in this case to comment on the assumption that AI is likely to play destructive, dangerous roles in society.

Open technical research problems include: improving automated game generation and automated game playing in open-ended spaces, enabling more close coupling between the generative mode (currently used for brainstorming) and the user-operated design interfaces, and developing new methods to enable users to playfully explore design spaces (the latter includes developing methods to better familiarise users with the Computational Creativity concept of design-space navigation in the first place).

Acknowledgments

This work is funded by EC FP7 grant 621403 (ERA Chair: Games Research Opportunities). We are grateful to Girlguiding Cornwall and the Camborne Science and International Academy for their collaboration.

References

Bittanti, M., and Quaranta, D. 2009. *Gamescenes: Art in the Age of Videogames*. Johan & Levi Editore.

Bonaiuto, A.; Mingrino, M.; Sampugnaro, R.; Fallica, S.; and Mica, S. 2014. Participation at the Global Game Jam: A bridge between consumer and producer worlds in digital entertainment. *GAME* 3(2):35–45.

Colton, S., and Ventura, D. 2014. You can't know my mind: A festival of computational creativity. In *Proc. Intl. Conference on Computational Creativity*.

Colton, S.; Nelson, M. J.; Saunders, R.; Powley, E. J.; Gaudl, S. E.; and Cook, M. 2016. Towards a computational reading of emergence in experimental game design. In *Proc. Computational Creativity and Games Workshop*.

Compton, K., and Mateas, M. 2015. Casual creators. In *Proc. Intl. Conference on Computational Creativity*.

Cook, M.; Colton, S.; and Gow, J. 2016. The ANGELINA videogame design system. *IEEE Transactions on Computational Intelligence and AI in Games*.

Grace, K., and Maher, M. L. 2014. Towards computational co-creation in modding communities. In *Proc. Workshop on Experimental Artificial Intelligence in Games*, 15–20.

Kultima, A. 2015. Defining game jam. In *Proc. Conference on the Foundations of Digital Games*.

Liapis, A.; Smith, G.; and Shaker, N. 2016. Mixed-initiative content creation. In *Procedural Content Generation in Games*. Springer. 195–214.

Liapis, A.; Yannakakis, G. N.; and Togelius, J. 2014. Computational game creativity. In *Proc. Intl. Conference on Computational Creativity*.

Musil, J.; Schweda, A.; Winkler, D.; and Biffl, S. 2010. Synthesized essence: What game jams teach about prototyping of new software products. In *Proc. Intl. Conference on Software Engineering*, 183–186.

Nelson, M. J.; Colton, S.; Powley, E. J.; Gaudl, S. E.; et al. 2017. Mixed-initiative approaches to on-device mobile game design. In *Proc. CHI Workshop on Mixed-Initiative Creative Interfaces*.

Powley, E. J.; Colton, S.; Gaudl, S. E.; Saunders, R.; and Nelson, M. J. 2016. Semi-automated level design via auto-playtesting for handheld casual game creation. In *Proc. IEEE Conference on Computational Intelligence and Games*, 372–379.

Resnick, M.; Maloney, J.; Monroy-Hernández, A.; et al. 2009. Scratch: Programming for all. *Communications of the ACM* 52(11):60–67.

Resnick, M. 2004. Edutainment? No thanks. I prefer playful learning. *Associazione Civita Report on Edutainment* 14.

Sharp, J. 2012. A curiously short history of game art. In *Proc. Conference on the Foundations of Digital Games*.

Shedroff, N., and Noessel, C. 2012. *Make It So: Interaction Design Lessons from Science Fiction*. Rosenfeld Media.

Shneiderman, B. 2007. Creativity support tools: Accelerating discovery and innovation. *Communications of the ACM* 50(12):20–32.

Smith, G.; Whitehead, J.; and Mateas, M. 2011. Tanagra: Reactive planning and constraint solving for mixed-initiative level design. *IEEE Transactions on Computational Intelligence and AI in Games* 3(3):201–215.

Strawhacker, A.; Lee, M.; Caine, C.; and Bers, M. 2015. ScratchJr demo: A coding language for kindergarten. In *Proc. Intl. Conference on Interaction Design and Children*.

Taylor, T. L., and Witkowski, E. 2010. This is how we play it: What a mega-LAN can teach us about games. In *Proc. Conference on the Foundations of Digital Games*, 195–202.

Westecott, E. 2013. Independent game development as craft. *Loading...* 7(11).

Yannakakis, G. N.; Liapis, A.; and Alexopoulos, C. 2014. Mixed-initiative co-creativity. In *Proc. Conference on the Foundations of Digital Games*.

Zook, A., and Riedl, M. O. 2013. Game conceptualization and development processes in the Global Game Jam. In *Proc. FDG Workshop on the Global Game Jam*.