Refining the Paradigm of Sketching in AI-Based Level Design

Antonios Liapis and Georgios N. Yannakakis
Institute of Digital Games, University of Malta, Msida, Malta
{antonios.liapis@um.edu.mt, georgios.yannakakis}@um.edu.mt

Abstract

This paper describes computational processes which can simulate how human designers sketch and then iteratively refine their work. The paper uses the concept of a map sketch as an initial, low-resolution and low-fidelity prototype of a game level, and suggests how such map sketches can be refined computationally. Different case studies with map sketches of different genres showcase how refinement can be achieved via increasing the resolution of the game level, increasing the fidelity of the function which evaluates it, or a combination of the two. While these case studies use genetic algorithms to automatically generate levels at different degrees of refinement, the general method described in this paper can be used with most procedural generation methods, as well as for AI-assisted design alongside a human creator.

Procedural content generation has traditionally been used in commercial games to create novel player experiences in every playthrough. More recently, generative algorithms have also been used during game development in an effort to speed up the design of content, especially for optional content such as trees with *Speedtree* (IDV 2011) or for tedious tasks such as terrain generation with *Worldmachine* (Schmitt 2005). There is increasing interest, both in academia and in the game industry, for computational creators which can work alongside human creators in a mixed-initiative manner (Yannakakis, Liapis, and Alexopoulos 2014), or which exhibit human-like design patterns (Dahlskog, Togelius, and Björk 2015) in order to be trackable, familiar and suggestive to their audience (be they human players or designers).

As a stepping stone towards expressive generators which exhibit a human-like design process, this paper proposes a generative approach which designs game levels via iterative refining. The paper builds on previous work on map sketches, which themselves borrow from human sketching as a design paradigm. Map sketches are low-resolution, high-level abstractions of game levels, and contain the bare minimal components which can describe a level of a game genre. As a design medium, a map sketch follows several of the properties of sketches according to Hugh Dubberly, as attributed by Buxton (2007): *"A sketch is incomplete, somewhat vague, a low-fidelity representation. The degree of fidelity needs to match its purpose, a sketch should have 'just enough' fidelity for the current stage in argument building."*

Sketching as a paradigm has a long history in computer-aided design, as exemplified by Sketchpad in the 1960s (Sutherland 1963). The current paper builds on the notion of sketching at varying degrees of fidelity with the final game level, increasing the fidelity as the design matures. Increasing the level of detail in a design process necessitates a change in representation (Grace, Gero, and Saunders 2014) which however transfers design knowledge from one iteration to the next (Goel 1997). The goal of this paper is to emulate, computationally, the iterative process of refining an idea, a product or an artwork which is followed by most creative people — from designers and artists to engineers and academics. The case studies in this paper iteratively refine game levels by using previously generated rough sketches as initial seeds for a constrained optimization algorithm which expands and adds details to the rough sketch, or use more accurate evaluations for improving its quality. On the one hand, iteratively refining procedurally generated content can benefit the computational processes themselves; this is especially true for the search-based methods used in this paper, where iterative evolution narrows the search space and makes optimization more efficient. On the other hand, using human-like design paradigms (such as increasing the resolution of a sketch or the simulation fidelity) makes the generator more transparent to human designers, which in turn allows them to better interact with it either at the code level or via a mixed-initiative design interface.

Related Work

As noted above, using sketching as a design paradigm when creating computer-aided design tools has a long history, dating back to Sutherland (1963). Another human-like design process popular in mixed-initiative design research is that of a dialog (Novick and Sutton 1997), where computer and human share a task initiative (who starts the conversation), a speaker initiative (when to speak) and an outcome initiative (who determines when the discussion is over). The iterative refining processes discussed in this paper assume that the computer initiative is leading the process with a human selecting when and what to refine, but otherwise not directly influencing this process. However, more agency and control can be afforded to the human user via an interface similar

to that of *Tanagra* (Smith, Whitehead, and Mateas 2011) or *Sentient Sketchbook* (Liapis, Yannakakis, and Togelius 2013a); suggestions for such an interactive tool for refinement are provided in the discussion section of this paper.

The concept of map sketches as low-detail descriptions of game levels is arguably not new; a large portion of procedural content generation (PCG) research revolves around small-scale level generation with local changes (Sorenson and Pasquier 2010; Bauer, Cooper, and Popović 2013). Most generators target specific, popular games (Pedersen, Togelius, and Yannakakis 2009; Togelius et al. 2010), while the low resolution of map sketches makes them appropriate for multiple game genres (as demonstrated in this paper). The concept of map sketches borrows from similar concepts such as procedural portraits (Mateas 2000), which use computational systems to represent human cultural processes, or more appropriately computational caricatures (Smith and Mateas 2011), which purposefully distill a game design process (to the degree of oversimplification and exaggeration) in order to more clearly communicate "nuggets of truth". The work described in this paper is in several ways a computational caricature of the human design process, simplified and streamlined to the point that it becomes suggestive and familiar to human design practitioners.

There have been several attempts at emulating the process of iterative refining in PCG, although not always termed as such. Sentient World (Liapis, Yannakakis, and Togelius 2013b) refines game terrain by increasing the heightmap's resolution to conform to patterns of a low-resolution sketch. Hartsook et al. (2011) use a linear story as a seed for evolving a tree data structure of locations in an adventure game, and then convert the tree into a grid-based map via constructive methods. Dormans (2010) uses generative grammars to produce missions in adventure games (which describe the sequence of challenges in a level), which then inform the generation of the level's spatial arrangement via space grammars. Game action sequences have also been refined with spatial information for dungeons (van der Linden, Lopes, and Bidarra 2013) and puzzle levels (Smith et al. 2012).

Refining Map Sketches

As an indicative sample of the ways in which human designers refine their work iteratively, this paper proposes several ways in which map sketches can be refined:

1. **Increasing the resolution of the sketch**, i.e. the map size and the number of its components (tile types). This approximates the iterative refinement of artwork, where an artist starts from a pencil sketch and refines it by adding shading or color and eventually touches up the details.
2. **Increasing the evaluation fidelity of the sketch**, i.e. moving from high-level evaluations on path connectivity to game-specific evaluations and eventually to in-game simulations. This approximates the iterative refinement of engineering schematics such as a bicycle, where a designer starts from a general interaction principle, then runs computer-based simulations of a refined schema and finally builds a physical prototype to test in the real world. Buxton (2007) covers such an example of bicycle design.
3. **Expanding with multiple sketches**, e.g. linking sketches to form a larger level. The addition of more sketches may require changes in the original sketch (e.g. re-balancing a strategy game level if a sketch expands the map with two more players), or it might not (e.g. if the added sketch is an entirely new level of a *Diablo* dungeon). This approximates the iterative refinement of a storyworld in fantasy literature, where the author in the first book fleshes out a portion of the world while in future books introduces new lands, new characters, new lore and new subplots.

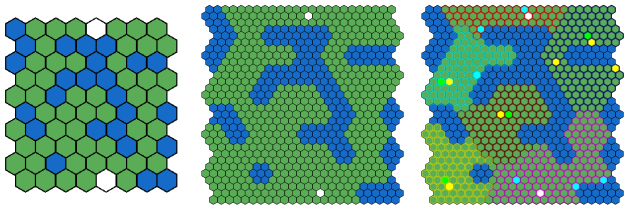
This paper presents three case studies which demonstrate how the different ways of iterative refining suggested above can be achieved in level generators. The case studies use different game genres as target domains, and use different approaches to move from one stage of refinement to the next. For the sake of brevity, the case studies demonstrate refining from a *first stage* (a simple, rough sketch) to a *second stage* (a more elaborate or more accurate sketch) of the design process. More stages of refinement can be added, however, as suggested in the discussion section of this paper.

While the case studies in this paper attempt to present different refinement methods applied on levels for different game genres, they share a common method for generating and evaluating these levels. All case studies represent a map sketch as an array of integers, with each integer defining the tile type of a specific map tile. All case studies use constrained optimization to evolve feasible content (which satisfy constraints depending on the case study) alongside infeasible content, via a feasible-infeasible two-population genetic algorithm (FI-2pop GA) (Kimbrough et al. 2008). Unless otherwise noted, constraints on generated levels pertain to connectivity of important game-specific tiles (e.g. weapons for a shooter game, resources for a strategy game) via passable paths; for such constraints, the infeasible population of the FI-2pop GA minimizes the number of disconnected paths. Similarly, most case studies use the heuristics of Liapis, Yannakakis, and Togelius (2013c) to evaluate feasible levels. In short, these heuristics consist of:

- *exploration* as $f_e(S_N)$ which evaluates the effort made to discover tiles in the set S_N starting from other tiles in the same set, and its balance dimension as $b_e(S_N)$, i.e. if all tiles in S_N are equally difficult to find from each other.
- *safe areas* as $f_a(S_N)$ which evaluates the number of passable tiles significantly closer to one tile in the set of S_N than other tiles in the same set, and its balance dimension as $b_a(S_N)$, i.e. if tiles in S_N have similar-sized safe areas.
- *strategic resource control* as $f_s(S_N, S_M)$ which evaluates whether tiles in the set of S_M are significantly closer to one tile in the set of S_N than other tiles in S_N, and its balance dimension as $b_s(S_N, S_M)$, i.e. whether each tile in S_N has equal nearby tiles in S_M.

Refining the Map Resolution

This case study uses strategy games to demonstrate how a map sketch can be refined by increasing its *resolution*. The intended game, which informs the map's tiles and its evaluation functions, is *Endless Legend* (Amplitude 2014). Levels for *Endless Legend* are laid out on a hex grid of tiles.



(a) First stage evolved sketch (b) Refined version of Fig. 1a (c) Second stage evolved sketch

Figure 1: Map sketches for *Endless Legend*, at different degrees of refining. Map sketches contain traversable land (dark green), impassable water (dark blue), the players' starting positions (white), and during the refinement stage also expansion tiles (light green), glassteel (yellow) and titanium (cyan) special resources.

A player can place one city per *region*; regions consist of a (usually large) number of connected tiles and only one player can control each region. Players can harvest *strategic resources* in a region where they have a city, by building extractors on them (regardless of their distance to the city). Strategic resources can be used to equip powerful units and build special structures in cities, and in the early game consist of *titanium* and *glassteel*. For brevity, advanced technologies which unlock other strategic resources or which allow travel through water are not considered in this case study; the maps created will focus on early-game expansion. Moreover, different types of terrain, varying terrain elevation and other features of *Endless Legend* will be ignored, but can be added with more refinement stages.

First (sketching) stage: For maps at the first stage of sketching, the minimal components of an *Endless Legend* level are land tiles, water tiles (which can not be traversed without extensive research), and the starting locations of players. At this level of detail and with those tile types in place, the goal of a generator is to create maps where players do not start next to each other — allowing for some breathing room and area for expansion before players engage in border warfare. This is evaluated by F_1 in eq. (1):

$$F_1 = f_e(S_P) + b_e(S_P) \quad (1)$$

where S_P is the set of players' starting positions.

The FI-2pop GA evolves via mutation, which transforms land tiles to sea tiles (and vice versa) or swaps the position of adjacent tiles. When evolving a map for two players at this stage, the best result of 20 independent runs is shown in Fig. 1a; players start the game at opposite sides of the map and a large body of water blocks direct access to one another.

Second (refining) stage: Evolution at the first stage has secured a rudimentary map sketch, which can be refined by increasing the map's resolution at the second stage; in this case study, the map dimensions are tripled as each hex in the first stage sketch is essentially represented as 7 hexes in the second stage sketch (see Fig. 1b). With more granularity and free space due to a larger map, more details can be added. New tile types are added at the second stage: glassteel and titanium tiles (representing those strategic resources), and expansion tiles. Expansion tiles represent the center of a region defined by any land tile which is closer to the expansion tile than to any other expansion tile or player starting location (similar to Voronoi diagrams); players can settle regions in *Endless Legend* by building cities within their borders.

Placement of titanium, glassteel and expansion tiles can be optimized using genetic search; in the current study, the best evolved sketch of the first stage defines and constrains the topology of all sketches of the second stage. Essentially, the genetic algorithm focuses on placing titanium, glassteel and expansion tiles while leaving sea, land and starting location tiles intact. Despite the larger map size, the number of parameters to be optimized (i.e. coordinates of titanium, glassteel and expansion tiles) remains manageable and optimization is going to be faster and more efficient. Moreover, since the best first stage sketch is feasible (as is its refined version), it is very unlikely that any refined sketch will be infeasible. For the second stage, sketches are evolved to form regions of similar (large) size with a balanced number of strategic resources, as evaluated by F_2 in eq. (2).

$$F_2 = f_s(S_P \cup S_E, S_G \cup S_M) + b_s(S_P \cup S_E, S_G \cup S_M) + f_a(S_P \cup S_E) + b_a(S_P \cup S_E) \quad (2)$$

where S_E is the set of expansions; S_G is the set of glassteel resource tiles; S_M is the set of titanium resource tiles. Note that f_a and b_a have a lowest safety threshold of 0 (i.e. any tile even marginally closer to one expansion or starting location is considered part of its safe area).

The best evolved sketch of the second stage of refining, using the sketch of Fig. 1b as a seed, is shown in Fig. 1c; regions are shown in different colored hex borders. In the sketch of Fig. 1c, each region has at least one titanium or glassteel, although they are not distributed entirely fairly (compare e.g. the bottom-left region with one glassteel versus the top-right region with three glassteel). The player starting at the top can immediately expand to two regions, while the player at the bottom can expand to three; both players can immediately expand to the resource-rich top-right region, which is expected to cause border tension.

Results: The best sketches in 20 independent runs for the first stage had on average 33% of the map covered by water and a passable path branching factor of 4.62 (out of 6); while maps included peninsulas and narrow straits, there were also large open areas of land as well. The best sketches for the second stage had each region spanning on average 82 tiles. Interestingly, regions centered on expansion tiles spanned 95 tiles on average while regions of player starting locations spanned 55 tiles; as the previous stage optimizes exploration of player locations, those are placed in areas cut off from the rest of map with few connected (and safe) passable tiles.

Refining the Approximations of Gameplay

This case study uses levels for the *MiniDungeons* game to demonstrate how a map sketch can be refined by improving the fidelity of its evaluation. These levels will be optimized at the first stage according to the general heuristics of Liapis,

Yannakakis, and Togelius (2013c) while, at the second stage, on more game-dependent and accurate evaluations based on simulated gameplay. *MiniDungeons* is best described as a roguelike puzzle game where a player controls a hero's journey through a dungeon (Holmgård et al. 2014); its levels are small and its gameplay relatively simple. A *MiniDungeons* level contains an entrance and an exit, immobile monsters which deal a random amount of damage before dying in a fight with the hero, treasures which increase the score when collected and potions which immediately heal an injured hero when collected. The only stochasticity of gameplay is in monster damage: each monster deals between 5 and 14 hit points (HP) of damage of a hero's total 40 HP. In order to model human players, several *procedural personas* were developed to emulate archetypical decision making styles (e.g. kill all monsters). This case study will use procedural personas for creating synthetic playtraces at the second stage of generation; included personas are a *baseline* persona which is rewarded for reaching the exit (thus tending to go on the shortest path to the exit), a *monster killer* which is rewarded for each monster slain and exit reached and a *treasure collector* which is rewarded for each treasure collected and exit reached. Previous work which used such personas as critics of a constrained evolutionary algorithm (Liapis et al. 2015) did not account for a prior sketching stage, but is largely an inspiration for the current case study.

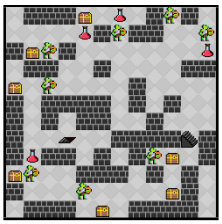
First (sketching) stage: Unlike the other case studies in this paper, *MiniDungeons* levels on both stages of refinement use the same map size (12 by 12 tiles) and the same tile types: empty, walls, monsters, potions, treasures, entrance and exit. All maps have the same number of monsters, potions and treasures, and only one entrance and one exit. Both stages use an FI-2pop GA to evolve the levels via mutation (which swaps adjacent tiles and transforms walls to empty tiles and vice versa). The first stage evolves *MiniDungeons* levels via F_1 in eq. (3), aiming to hide the exit in a remote location, disperse monsters throughout the level and place treasures or potions near monsters (as a reward); in feasible levels all non-empty, non-wall tiles must be connected.

$$\begin{aligned} F_1 =& f_s(S_M, S_T \cup S_P) + b_s(S_M, S_T \cup S_P) \\ &+ f_a(S_E \cup S_X \cup S_M) + b_a(S_E \cup S_X \cup S_M) \\ &+ f_e(S_E \cup S_X) + b_e(S_E \cup S_X) \end{aligned} \quad (3)$$

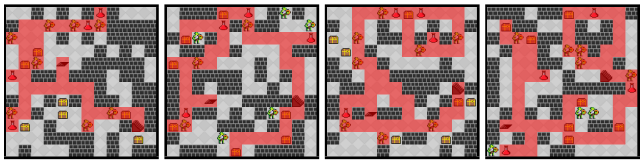
where S_E is the set of entrances; S_X is the set of exits; S_M, S_T, S_P is the set of monsters, treasures, potions respectively.

The best evolved first stage sketch among 20 independent runs is shown in Fig. 2a; the patterns targeted by F_1 are present: the exit is reachable from the entrance via two winding paths, and monsters are faraway from each other, but near treasures and potions — although not necessarily blocking the path of a hero who tries to collect them.

Second (refining) stage: Although it targets several desirable patterns, F_1 focuses on path distance without accounting for the game's rules or the hero's survival. Using the evolved levels of the first stage as an initial seed, the evaluations and constraints can be refined to account for the game's rules by using procedural personas to play through a level and evaluate whether it satisfies the persona's idiosyncratic priorities. The baseline persona is used to test if the level satisfies a playability constraint; that constraint is satisfied if the baseline persona reaches the exit in a worst-case scenario where monster damage (normally random) is set at maximum value. Another constraint that the baseline persona explores at least 12 tiles before reaching the exit precludes the entrance and exit being too close. The other personas are used to evaluate feasible individuals which satisfy all constraints (including connectivity constraints of the first stage). The monster killer persona can evaluate levels on the number of monsters it kills and exits it reaches (F_{MK}), while the treasure collector persona can evaluate levels on the number of treasures it collects and exits it reaches (F_{TC}). To cater for the stochasticity of combat, fitnesses are averaged from 10 simulations with random monster damage. F_{MK} and F_{TC} evaluate how well the level accommodates the goals of that persona. However, trivial solutions can be found for these goals (by e.g. placing all treasures on a line between entrance and exit). Instead, decision making assumes a certain risk-reward mechanism; this risk can be measured via the standard deviation of monsters killed and exits reached for the monster killer (D_{MK}) and the standard deviation of treasures collected and exits reached for the treasure collector (D_{TC}) across 10 independent simulations.



(a) 1st stage ev. sketch



(b) 2nd stage sketch evolved for F_{MK} (c) 2nd stage sketch evolved for F_{TC} (d) 2nd stage sketch evolved for D_{MK} (e) 2nd stage sketch evolved for D_{TC}

Figure 2: Map sketches for *MiniDungeons*, at different degrees of refining. Fig. 2b-2e include a heatmap of the procedural persona evaluating them: monster killer or treasure collector (MK or TC notation respectively).

Figure 2 shows the best results of evolutionary runs targeting different fitnesses. Every evolutionary run used the entire population of the first stage as the initial seed; since constraints are refined in the second stage via playability checks with a baseline persona, some previously feasible individuals become infeasible. Additionally, the best map of the first stage may receive low fitness with the new evaluations; for instance, the (previously best) map in Fig. 2a receives a fitness of 0 for D_{TC} as the treasure collector can gather all the

(mostly unguarded) treasures without ever dying. Evolved maps for D_{MK} and D_{TC} are quite different from Fig. 2a, although some patterns (e.g. entrance placement) largely remain. In contrast, the map of Fig. 2a is optimal for F_{TC} so evolution does not improve it; Fig. 2c is the same as Fig. 2a.

Results: After the first stage of evolution, the best sketches of 20 independent runs were tested with the refined, simulation-based evaluations using procedural personas (i.e. artificial agents): a monster killer playing through them killed on average 85% of monsters, drank all potions and died before reaching the exit in 90% of the simulations; a treasure collector gathered on average 98% of treasures and died in 2.5% of simulations. After the second stage of refinement, in sketches evolved towards F_{MK} a monster killer died in half of the simulations and killed on average 99% of monsters; in those evolved towards F_{TC} a treasure collector never died and collected all treasures. In sketches evolved towards D_{MK} a monster killer died in 71% of the simulations and killed on average 78% of monsters; in those evolved towards D_{TC} a treasure collector died in 31% of the simulations and collected on average 79% of treasures.

Refining via Multiple, Linked Sketches

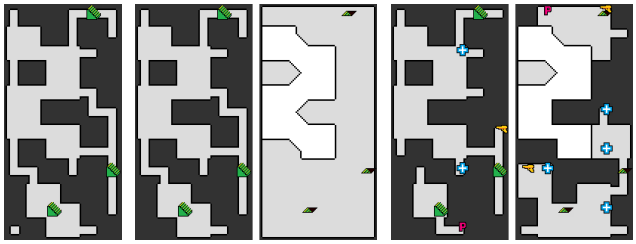
This case study uses archetypical first person shooter games to demonstrate how refinement can occur by adding extra sketches to the original, refining both in the process. Unlike previous case studies, levels created by this refinement process are not for an actual game (although most shooters revolving around multiplayer deathmatch sessions would fit the description). This genre of games involves two or more teams competing to score more kills against their opponents by using powerful weapons which can be picked up at specific locations in the level. Players lose health when shot, and upon dying reappear (re-spawn) at the team's spawn point; wounded players can heal by picking up healthpacks at specific locations in the level. This case study aims to create a two-floor shooter level; the first stage of sketching defines an initial level architecture for the 1st floor and ways of reaching the 2nd floor (stairs), while the second stage adds another sketch (the 2nd floor) which is linked to the first stage sketch, and adapts both sketches with more shooter-specific gameplay tiles such as weapon pickups and team spawn points.

First (sketching) stage: The first stage of sketching builds an initial level architecture for the 1st floor of the shooter level using empty tiles and walls. In anticipation of the 2nd floor during the refining stage, *stair* tiles which connect the two floors are also placed. First stage sketches evolve via F_1 of eq. (4), to place stairs in remote locations (allowing different entry points to the 2nd floor, when it is added). Moreover, F_1 ensures that stairs will be hidden behind extensive walls and thus defensible (limiting opponents' line of sight); stairs are expected to be chokepoints for reaching the 2nd floor.

$$F_1 = f_e(S_s) + b_e(S_s) \tag{4}$$

where S_s is the set of stairs.

The best first stage sketch among 20 independent runs is shown in Fig. 3a: stairs are far apart, often in corridors, with several "rooms" (formed from extended walls) in-between.



(a) 1st stage ev. sketch (b) Refined version of Fig. 1a (c) Second stage evolved sketch

Figure 3: Map sketches for a multi-floor shooter level, at different degrees of refining. Map sketches contain walls (black), passable tiles (gray), open-air tiles (white), stairs (green), spawn points (magenta), weapons (orange) and healthpacks (cyan).

Second (refining) stage: The first stage only created a basic architectural sketch of the 1st floor, in preparation of a 2nd floor; the second stage adds a sketch for the 2nd floor and can now estimate the gameplay of the full level. The second stage introduces the following game-specific tiles, which are the core components of shooter gameplay: *team spawn points* where teams start from and players reappear when they die, *weapons* where players can pick up powerful weapons, and *healthpacks* where players can heal when injured. Moreover, the second floor requires that a new tile with architectural properties is introduced: *open-air* tiles signify areas where players can drop to (or shoot at) the floor below — essentially holes in the architecture which players can not cross except by dropping to the floor below[1].

The 1st floor of the previous stage of sketching informs, to a degree, the appearance of the two-floor level before it is further evolved. The wall tiles and stairs are frozen, and cannot be changed (or moved) during refining. Moreover, while the 2nd floor starts with no wall tiles, the closed spaces created by walls in the first stage sketch determine where open-air tiles are placed on the 2nd floor. Any 1st floor segment (i.e. chunk of passable tiles isolated from other areas via chokepoints such as doors and corridors) which does not contain a flight of stairs creates an open-air section in the 2nd floor above it (see Fig. 3b for an example). This constructive algorithm maintains the appearance of defensible "rooms" where stairs are located while allowing players between different floors to interact in "arena" parts of the level where access to the second floor is not easy.

Since this second stage introduces gameplay tiles, evolution must target several objectives. The FI-2pop GA ensures satisfaction of a constraint that all gameplay tiles must remain connected[2]. Feasible individuals evolve towards placing spawn points of opposing teams far away from each other, placing weapon pickups faraway from those spawn

[1] In contrast, passable tiles on both floors assume a solid floor where players can walk through (including balconies or bridges).

[2] For instance, a player must be able to reach a healthpack by jumping down from the 2nd floor via an open-air tile, but must also be able to get back to the 2nd floor from there via a stair.

points (and from each other) in order to prompt movement through the level, and distributing healthpacks evenly throughout the level so that all players have a chance of healing up and getting back to the fight. Therefore, second stage sketches evolve towards F_2 in eq. (5); the two f_e (and b_e) heuristics put more emphasis on spawn points being far away and a lesser emphasis on weapons being far away.

$$F_2 = f_a(S_H \cup S_P) + b_a(S_H \cup S_P) + f_e(S_P) + b_e(S_P) + f_e(S_W \cup S_P) + b_e(S_W \cup S_P) \quad (5)$$

where S_P is the set of team spawn points; S_H is the set of healthpacks; S_W is the set of weapon pickups.

Fig. 3c shows the best evolved individual of the second stage, seeded from the sketch of Fig. 3b. While the placement of the stairs and walls of the first stage is intact, more walls were added to block paths between stairs on the 1st floor; stairs are now connected only via the 2nd floor. The first team team spawns on the 2nd floor, while the second team spawns on the 1st floor. Interestingly, the first team controls more area on the 1st floor, as its members can drop down through the open-air section next to their spawn point. The second team can access the 2nd floor via a nearby stair, and can control half of it easily since its members can aim at the first team's members who navigate the arena of the 1st floor. The first team can access that area of the 2nd floor as well, but only through a faraway stair hidden in a corridor.

Results: The best sketches in 20 independent runs for the first stage had an average of 2.75 stairs (with a maximum of 4 stairs), with 38% of the 1st floor tiles being walls and 27% being corridors (i.e. deadends, crossroads and chokepoints). In refined versions of these sketches, 13% of 2nd floor tiles were open-air tiles on average (but 6 of 20 sketches had no open-air tiles). The best sketches for the second stage had walls on 44% of 1st floor tiles and 20% of 2nd floor tiles, with corridors on 26% of 1st floor tiles and 16% of 2nd floor tiles. Refinement seems to disperse wall tiles on both floors, although the first floor is more cramped (narrow winding passages) since walls of the first stage sketch are retained.

Discussion

The three case studies described above suggest three indicative ways of refining map sketches by increasing the level of detail either on the map level (by increasing the map size or by combining multiple sketches) or on the evaluation level (by improving fidelity with the end-product playthrough of humans). Moreover, these case studies provide distinct examples of how the first stage of sketching can inform the second stage of refining: in maps for *Endless Legend*, all tile types optimized on the first stage are frozen in place during the second stage; in *MiniDungeons* levels, all tile types are available (and can be manipulated) at both stages; in shooter levels, certain tiles of the previous stage (e.g. walls) are frozen in place but can be enhanced (by e.g. adding more walls) based on the new needs of a refined level. Finally, the case studies have demonstrated how even simple evaluations (such as exploration and safety) of Liapis, Yannakakis, and Togelius (2013c) can be adapted and reused with alternate pathfinding methods such as the hex-based paths of *Endless Legend* maps or the multi-floor paths of FPS levels, with stairs for moving up and open-air tiles for dropping down.

The refinement step allows generators to add level details (such as new tile types) iteratively, working in a similar way to how humans move from the big picture to the minutiae. However, increasing the map size, the number of tile types, or the fidelity of evaluations incurs a computational burden and slows down evolution. To be of use in AI-assisted design tools, certain shortcuts or interface changes must be made, either to provide faster feedback (by e.g. limiting the genotype to only those elements that can change) or to change the human-computer co-creation paradigm to a more asynchronous process (by e.g. humans and machines performing the refining stages individually and presenting each other their verdicts). Another issue with the iterative process of refining is that variation of results is purposefully limited; for instance, *Endless Legend* maps of the same population look similar during the refining step, since only a handful of tiles change between them. Running evolution "from scratch" rather than in stages creates more variation but likely struggles to find adequate results in a much larger search space. Finally, converting an evolved first stage sketch to a second stage seed for refinement (in the case of *Endless Legend* maps and shooter levels) was ad-hoc based on expert knowledge; in other types of levels (or in other refinement stages) such a conversion may be less straightforward.

While the case studies in this paper used constrained optimization via genetic algorithms for both sketching and refining stages, the general process of iterative refining lends itself well to many different PCG methods. For instance, while a rough sketch of a platformer level can be produced with stochastic search, more details can be added via Answer Set Programming as implemented in *Tanagra* (Smith, Whitehead, and Mateas 2011) by converting the platforms of the rough sketch (and the desired properties of the refined sketch) into constraints. Similarly, in levels such as those of *MiniDungeons* but with lock and key puzzles, planning approaches are likely more robust in ensuring all preconditions for completing a level are met. Constructive methods can also be applied on different stages of the refinement process; arguably, converting first stage sketches to second stage large maps for *Endless Legend* already uses constructive methods, which could be enhanced to e.g. ensure all regions have exactly the correct number of strategic resources.

It should be noted, however, that not all generators can be used with all of the refining processes described in this paper. For instance, increasing the fidelity of evaluation assumes generators which either evaluate (e.g. search-based methods) or test for playability (e.g. generate-and-test and constraint-based methods). Similarly, increasing the resolution of a map sketch implicitly assumes generators able to consume and produce spatial arrangements of maps, and would not be applicable to e.g. a grammar-based generator.

Case studies in this paper covered only two stages of refinement, a first (sketching) stage and a second (refining) stage. However, many iterative steps of refining (of various sorts) could be added to the examples of this paper. For instance, the larger *Endless Legend* maps could be refined further with more tile types such as strategic resources

of later eras or with different terrain (e.g. forests), or could be refined via evaluations of basic strategic gameplay with agents such as those of μRTS (Ontañon 2013). Shooter levels could be refined by assigning specific weapons (e.g. rocket launcher or sniper rifle) to each weapon tile, or by refining the evaluation via simulations with artificial agents in the *Cube 2* engine as per Cardamone et al. (2011).

Conclusion

This paper proposed a method for imparting generators with more human-like design processes. Inspired by human art, design and engineering practices, several methods for computationally modeling the iterative refinement of ideas, artwork and designs were put forth. These methods were demonstrated in different case studies, targeting the creation of levels for a diverse set of game genres. While case studies in this paper relied on artificial evolution to optimize certain heuristics from the literature, the process of refinement can be used with most generative methods.

Acknowledgments

The research was supported, in part, by the FP7 ICT projects C2Learn (project no: 318480) and ILearnRW (project no: 318803), and by the FP7 Marie Curie CIG project AutoGameDesign (project no: 630665).

References

Bauer, A.; Cooper, S.; and Popović, Z. 2013. Automated redesign of local playspace properties. In *Proceedings of the 8th Conference on the Foundations of Digital Games*.

Buxton, B. 2007. *Sketching User Experiences: Getting the Design Right and the Right Design*. Morgan Kaufmann.

Cardamone, L.; Yannakakis, G. N.; Togelius, J.; and Lanzi, P. L. 2011. Evolving interesting maps for a first person shooter. In *Proceedings of Applications of Evolutionary Computation*, volume 6624, LNCS. Springer-Verlag. 63–72.

Dahlskog, S.; Togelius, J.; and Björk, S. 2015. Patterns, dungeons and generators. In *Proceedings of the 10th Conference on the Foundations of Digital Games*.

Dormans, J. 2010. Adventures in level design: Generating missions and spaces for action adventure games. In *Proceedings of the FDG workshop on Procedural Content Generation*.

Goel, A. K. 1997. Design, analogy, and creativity. *IEEE Expert* 12(3).

Grace, K.; Gero, J. S.; and Saunders, R. 2014. Interpretation-driven mapping: a framework for parallel search and re-representation for computational analogy in design. *AIEDAM Special Issue on Analogical Thinking*.

Hartsook, K.; Zook, A.; Das, S.; and Riedl, M. O. 2011. Toward supporting stories with procedurally generated game worlds. In *Proceedings of the IEEE Conference on Computational Intelligence and Games*.

Holmgård, C.; Liapis, A.; Togelius, J.; and Yannakakis, G. N. 2014. Evolving personas for player decision modeling. In *Proceedings of the IEEE Conference on Computational Intelligence and Games*.

Kimbrough, S. O.; Koehler, G. J.; Lu, M.; and Wood, D. H. 2008. On a feasible-infeasible two-population (fi-2pop) genetic algorithm for constrained optimization: Distance tracing and no free lunch. *European Journal of Operational Research* 190(2):310–327.

Liapis, A.; Holmgård, C.; Yannakakis, G. N.; and Togelius, J. 2015. Procedural personas as critics for dungeon generation. In *Proceedings of Applications of Evolutionary Computation*.

Liapis, A.; Yannakakis, G. N.; and Togelius, J. 2013a. Sentient sketchbook: Computer-aided game level authoring. In *Proceedings of the 8th Conference on the Foundations of Digital Games*.

Liapis, A.; Yannakakis, G. N.; and Togelius, J. 2013b. Sentient world: Human-based procedural cartography. In *Proceedings of Evolutionary and Biologically Inspired Music, Sound, Art and Design (EvoMusArt)*, volume 7834, LNCS. Springer. 180–191.

Liapis, A.; Yannakakis, G. N.; and Togelius, J. 2013c. Towards a generic method of evaluating game levels. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*.

Mateas, M. 2000. Expressive AI. In *SIGGRAPH 2000 Electronic Art and Animation Catalog*.

Novick, D., and Sutton, S. 1997. What is mixed-initiative interaction? In *Proceedings of the AAAI Spring Symposium on Computational Models for Mixed Initiative Interaction*.

Ontañon, S. 2013. The combinatorial multi-armed bandit problem and its application to real-time strategy games. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*.

Pedersen, C.; Togelius, J.; and Yannakakis, G. N. 2009. Modeling player experience in Super Mario Bros. In *Proceedings of the IEEE Symposium on Computational Intelligence and Games*, 132–139.

Smith, A. M., and Mateas, M. 2011. Computational caricatures: Probing the game design process with AI. In *Proceedings of the AIIDE workshop on Artificial Intelligence in the Game Design Process*.

Smith, A. M.; Andersen, E.; Mateas, M.; and Popović, Z. 2012. A case study of expressively constrainable level design automation tools for a puzzle game. In *Proceedings of the 7th Conference on the Foundations of Digital Games*.

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.

Sorenson, N., and Pasquier, P. 2010. Towards a generic framework for automated video game level creation. In *Proceedings of Applications of Evolutionary Computation*, volume 6024, LNCS. Springer. 131–140.

Sutherland, I. E. 1963. Sketchpad: A man-machine graphical communication system. In *Proceedings of the Spring Joint Computer Conference*, 329–346.

Togelius, J.; Preuss, M.; Beume, N.; Wessing, S.; Hagelback, J.; and Yannakakis, G. 2010. Multiobjective exploration of the Starcraft map space. In *Proceedings of the IEEE Conference on Computational Intelligence and Games*.

van der Linden, R.; Lopes, R.; and Bidarra, R. 2013. Designing procedurally generated levels. In *Proceedings of the AIIDE workshop on Artificial Intelligence in the Game Design Process*.

Yannakakis, G. N.; Liapis, A.; and Alexopoulos, C. 2014. Mixed-initiative co-creativity. In *Proceedings of the 9th Conference on the Foundations of Digital Games*.