Stigmergic, Diegetic Guidance of Swarm Construction

Samuel Truman
Games Engineering
Julius-Maximilians University
Würzburg, Germany
samuel.truman@uni-wuerzburg.de

Jakob Seitz
Games Engineering
Julius-Maximilians University
Würzburg, Germany
jakob.seitz1@stud-mail.uni-wuerzburg.de

Sebastian von Mammen
Games Engineering
Julius-Maximilians University
Würzburg, Germany
sebastian.von.mammen@uni-wuerzburg.de

***Abstract*—No matter how autonomous a technical system is, in the end, the goals it is meant to serve are specified by humans. This also applies to systems implementing self-organized construction. In this paper, we present an approach to guide such systems by means of an accessible spatial user interface. We inferred its basic requirements from a previously published taxonomy on interactive self-organizing systems. We found solutions to these requirements that we could all realize based on the very simple idea to instruct all aspects of a swarm by means of spatially placed control points, similar to existing road-map-based approaches. We implemented a very simple, yet effective model of a terrain-shaping swarm to test the guidance approach and manually explored the generative space.**

***Index Terms*—Self-organization, swarm, swarm guidance, human-swarm interface, 3DUI, self-organized construction**

I. Introduction

The Organic Computing definition of self-organization refers to the system-wide ratio of controllers versus agents [1]. It implies that in systems of high degrees of self-organization, the agents can act independently, as there is no need for a centralized instance instructing its peers. Yet, despite the potential of a (sub-)system to act independently, its actual degree of autonomy is calculated based on the internally determined control data versus the amount of information introduced from the outside. This sharp distinction between the degrees of self-organization and autonomy, thus, integrates the technological potential of systems and subsystems to act autonomously and the socio-technical need to instruct them at the same time.

In the context of social insects, construction processes are informed by various stigmergic cues [2], i.e. instructions mediated through the environment, whether these are built templates, CO_2 gradients, or pheromone signals. In robotic contexts, swarms can equally be guided by means of built templates, possibly enriched by locally associated data, e.g. giving the built elements unique identifiers [3] or be guided by means of shepherding units that steer simpler ones to fulfill high-level goals [4]. In general, the goals that necessitate a swarm's guidance can roughly be specified considering target states or shapes, goal functions, or user-specified environmental templates [5].

This tandem of goal specifications and concrete guidance efforts, and accordingly, the question of a swarm's degree of autonomy, also plays an important role when pursuing self-organized approaches to architectural designs, as for instance elaborated by [6] or [7]: It yields the question whether (a) a concrete design artefact is fleshed out by an architect and built by a robot collective, whether (b) there are certain hard constraints or desirable attributes of the built artifact and a self-organized approach is chosen to explore viable solutions, or, whether (c) the local efforts of coordination and construction dominate the emergent structure.

II. Related Work

Next to the research directions that motivated our approach as outlined in the introduction, we consider two more research fields seminal: (1) 3D User Interfaces (3DUI) as a study focus in the greater concept of Human-Computer Interaction, as well as (2) Procedural Content Generation (PCG) techniques based on self-organized agent collectives.

A. 3DUI

User interfaces (UIs) define how an interactive application is informed by and informs the user. It is a very involved challenge to determine which kind of input technologies should inform which interaction (sub-)tasks in which ways and how concretely the results should be communicated to the user—to maximize the user's speed and accuracy, to minimize the learning curve, user fatigue, etc. Motivated by the long-term need and desire to work alongside robotic collectives and also based on the fact that spatial interaction tasks are facilitated in immersive environments, we decided to aim for a 3DUI [8] in an immersive, virtual reality (VR) context [9].

A very effective means to collect input for 3D manipulation tasks are 3D widgets, i.e. objects that act as tools. When they relate/respond to the spatial context that they are placed in, they are also referred to as 3D gizmos. In the context of self-organized models, the aforementioned construction templates as well as human-placed beacons etc. could be considered real-world manipulation gizmos—the analogy to the function of stigmergic cues seems appropriate, also in VR. Generally, when UI elements are part of a virtual environment, one also refers to them as diegetic. Selecting from a set of different options is usually realized by UI menus. In 3D, ring-menus, that arrange the selectable options around an object, have

shown to be rather efficient [10], especially if the user can select an option by simply pointing at it, e.g. by shooting a ray from a 3D controller and if there are no more than 9 to 11 options shown at once [11].

B. Swarm-Based PCG

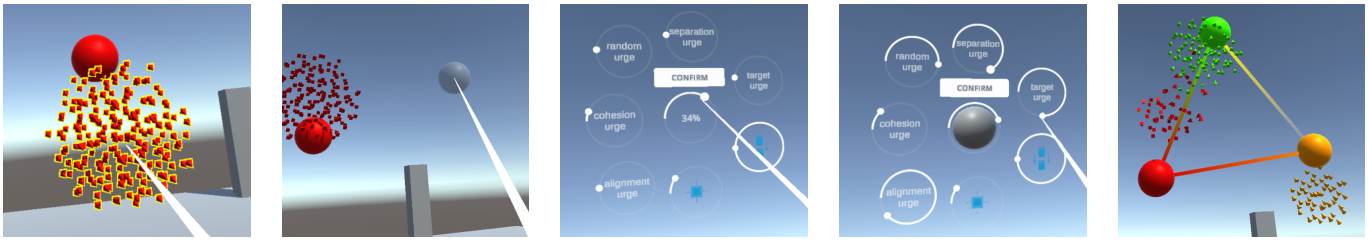
PCG is a term that is especially common in the context of computer games and virtual environments [12]. However, as it captures the idea to algorithmically create various artefacts such as audio streams and animations, but especially also geometric objects, it fits well the endeavor to generate architectural designs. Accordingly, we consider all those PCG approaches swarm-based which rely on self-organized models and simulations. We see a wide spectrum of disciplines that have contributed such works: Primarily biological works have retraced self-organized construction based on computational simulations, e.g. consider the coordinated 3D nest construction models presented in [13]. Architectural works have integrated these aspects, e.g. in [6], [7], [14]. The respective models typically implement reactive agents whose movements in 3D can simply be "random", that sense their environment, which, in turn, triggers the placement of construction elements with different virtual properties. These deposits can later be interpreted as control points of a parametric surface, or immediately serve to additively or subtractively sculpt a 3D model. Interestingly, such deposit-based approaches are also pursued when targeting game asset creation, especially when simulating effects of erosion when creating plausible 3D terrains for virtual worlds [15].

III. Concept

In this section, we first elaborate about our decisions regarding six design dimensions of the envisioned interactive self-organizing system based on [16]. Next, we consider the requirements of a representative scenario, which allows us to provide an overview of the basic interaction cycle.

A. Taxonomic Decisions

In terms of the first taxonomic dimension, *target of control*, we decided to manipulate the swarm itself, not its environment, as any indirect manipulation of the swarm could be introduced retrospectively, once all the necessary interaction tasks would be successfully implemented in the first place. Also, we did not see a reason why we should refrain from directly instructing the swarm. However, after several conceptual iterations, we arrived at a concept that could both be manipulating the swarm immediately but also draw from the benefits of indirect manipulation through the environment, i.e. spatial conditioning of manipulations as well as persistence over time. Further, by targeting manipulation of the topology, allowing to divide the swarm into arbitrary subsets and merge them again, we could modify subsets independently, thereby fostering heterogeneity of the population. In terms of the second dimension, the *level of control* exercised by the user, given the divisibility of the swarm, it would be possible to manipulate individuals of the swarm in the same way as subswarms or the whole swarm. In terms of the third dimension, *granularity of control*, we wanted to limit ourselves to rather simple and easily, visually traceable goals. Therefore, we decided to tightly couple a collective movement task with a construction task. For the first, we resorted to a simplified boids model [17] with a strong urge towards a "word center" that the user can set. For the latter, we simply let the agents drop deposits, as mentioned in Section II, that would fall to the ground and stick to whatever static object they would collide with first. Lower-level manipulation should be possible, too, by giving access to arbitrary configuration variables of the swarm, including those that would change the dynamic topology of the swarm. In particular, we wanted to allow the user to change parameters of the construction behavior, e.g. the depositing rate or the dimensions of the deposited element, as well as the boid flocking parameters, e.g. the separation coefficient. In terms of fourth dimension, *the time of interference*, we wanted to make it possible to interfere continuously, at real time, but also to offer the means to pause the simulation, introduce changes and resume the simulation afterwards. This would cover the whole spectrum of this taxonomic dimension. In terms of the fifth dimension, i.e. the user's *view*, we decided to aim for an immersive interface (see Section II-A) which translates the user's natural head movement to his/her view in the virtual environment. It would make any perspective, stereoscopic view possible from venture points the user could reach based on the teleportation-based navigation method that we decided to offer. It is a simple navigation technique that allows the use to instantaneously leap to a selected target point. It is a well-established method for travelling the virtual world, especially because it mitigates the effects of cybersickness [18]. In terms of the sixth and final dimension, i.e. concrete *user interface* elements, we arrived at the following insight after several agile iterations: Instructing the swarm about the next flight target (or world center), requires the specification of a three-dimensional position. Given more than one subswarms, we also need to specify which one the new target applies to. In order to establish a desirable flow of interaction, see again [8], we therefore ask the user to select a (sub-)swarm and drag a waypoint from its geometric center to an arbitrary position in space. This waypoint can also be used as a contextual 3D gizmo by allowing the user to decorate it with various other instructions affecting the swarm's configuration as soon as the target has been reached. For this reason, we also refer to them, more generically, as control points. In order to differentiate between where a swarm is heading to, we further distinguish between source and target points. Associating a proportion of individuals that should be drawn from a given (sub-)swarm to the target point, further allows for a convenient means to divide it—whereas directing several subswarms to a shared target would ensure they merge again. Finally, the user's freedom to not drag the target point anywhere, the intended manipulations could be effected immediately, providing a solution to cases of both immediate as well as stigmergic, diegetic manipulation.



(a) (b) (c) (d) (e)

Fig. 1. Interaction sequence of our guidance approach: (a) Selecting a swarm, (b) specifying a target waypoint, (c) defining a division ratio, (d) decorating the waypoint gizmo with further instructions, (e) sharing a common waypoint to merge subswarms.

B. Scenario Design

For a first example scenario, we identified the following important requirements: Next to spatially traceable local interactions and global effects, the impact of parallel work should also be visible. Hence, next to flight formations of the swarm, the built artefact should reflect the interaction and construction processes. At the same time, we wanted to clearly see the impact of our guidance approach.

These requirements in combination with our conceptual decisions outlined above, led to the following interaction loop of our example implementation, which is depicted in Fig. 1.

1) A flock of boid agents is spawned at the center of the virtual world (Fig. 1(a)).
2) The user selects the swarm, drags a waypoint outwards and drops it at a location in space, automatically establishing the required source-target relationship (Fig. 1(b)).
3) The user enters the size of the target swarm with respect to the source swarm by dialing an according angle in the waypoint's circular representation; the user may skip this step, if no division is desired (Fig. 1(c)).
4) The user decorates the waypoint with further instructions that manipulate the (sub-)swarm once it is within reach, e.g. changing the agents' deposit frequency or their flocking urge coefficients, etc. (Fig. 1(d)).
5) When directing several subswarms to an identical target, the arriving agents mingle and merge into one greater (sub-)swarm again (Fig. 1(e)).

IV. Model

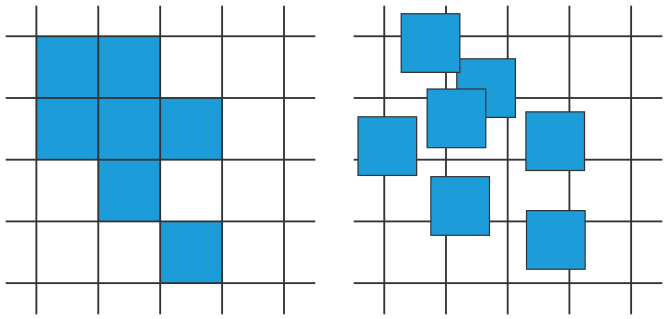
In this section, we detail the models and their realization for the swarm's flocking dynamics, the construction process, as well as the UI.

A. Simplified Boid Model

Instead of selecting the perceived neighbors of a boid agent based on its field of view as described in [17], we consider a whole subswarm as the neighborhood of its comprised agents. Thus, we calculate the averages of all its n agents' position and velocity states (yielding a linear runtime of $\Theta(n)$), and integrate the resultant *cohesion* and *alignment* urges as the respective individuals' differences to said averages, with constant runtime of $Theta(1)$ for each individual. The urge to *separate* opposes the geometric center of those neighbors which violate a spherical non-collision constraint around the respective individual. This information is efficiently provided by a rigid body physics engine which is tightly integrated into the Unity game engine that we rely on for our first implementation. In addition to the aforementioned urge towards a user-specified *waypoint*, some *random* urge was added as well. The total of these five normalized and weighted urges yielded the overall acceleration of an individual. The respective weights w_c, w_a, w_s, w_w and w_r, the radius r of the spherical non-collision constraint as well as the maximal norms of the acceleration and velocity vectors, a_{max} and v_{max}, respectively, complete the set parameter set of our simplified boid implementation.

B. Construction Behavior

At fixed intervals (by default $1s$), each swarm agent drops a cube which is fixed in the position of collision with the ground or another cube. There are two implementations which vary slightly: In the first one, a ray is shot from the agent to the ground, the first collision determines the spawning point of a new cube (offset by half its edge length) (see Fig. 2(a)). The second one additionally rounds the xy-values of the spawning position to let the cube snap into the cells of a virtual grid (see Fig. 2(b)). In the current implementation, the cubes persist, which implies a growth in the number of geometries in the scene proportional to the number of constructing swarm agents and their respective depositing intervals.



(a) (b)

Fig. 2. Both of our cube spawn variants: (a) the cubes snap to a virtual grid, and (b) they are placed directly below the boids.

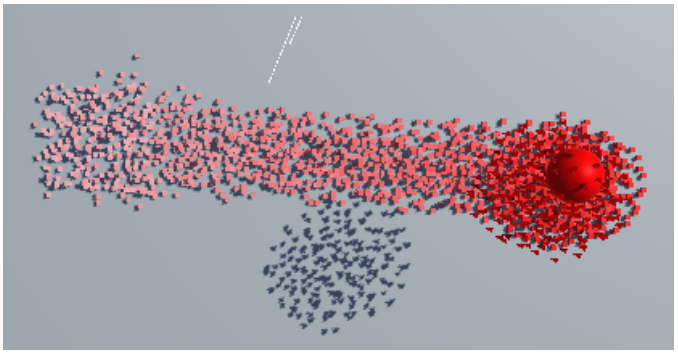


Fig. 3. The saturation of the generated cubes indicates their age. The swarm moved from the left to the right in the displayed top-down view.

C. User Interface Details

The overall interaction cycle has been outlined in Section III. We now provide some technical details and highlight the responsiveness and the immediate feedback the UI provides.

The user can perform all UI and locomotion interactions by means of a laser pointer. The beam's color changes from red to white to indicate that an action is valid, e.g. when an UI element or a valid teleportation location is selected. The user can change the beam's length to better interact with the UI and place control points far from his/her position. When the user selects a (sub-)swarm, boid agents of that swarm are highlighted. We generally implemented interaction highlighting of objects by means of an outline of their 2D silhouettes. Visual feedback is also provided when the user places a control point: A semi-transparent preview is displayed at the end of the laser beam where the actual control point would be placed, if the user lets go. The ring menu (Fig. 1(c)) is presented to the user immediately after a control point has been placed. We use radial sliders to allow the user to change the swarm division ratio and further properties. In order to interact with these sliders, the user can either select and drag a handle to the desired position along the circle, or directly select its target position. The functionality of each radial slider is explained by means of an according icon or label at its center. Once the user is satisfied with the parameters, the changes need to be confirmed by clicking a button. Changing any of the parameters is optional. All sliders store their respective last value so the user only has to specify changes from previous choices. A similar radial slider is also used for the simulation speed UI which is available to the user at all times. The simulation speed UI is placed at one fixed side of the virtual environment. This slider allows the user to modify the simulation speed or pause the simulation. Finally, we also provide visual feedback on the generated structures. Each swarm agent and control point is colorized differently to indicate to which (sub-)swarm they belong. Also, the cubes that are dropped by the swarm agents are colorized to visualize which swarm generated what structures. The cube colors' saturation is reduced over time to additionally visualizes the time of their placement (Fig. 3).

TABLE I
THE PARAMETER VALUE RANGES WE USED TO CREATE OUR EXAMPLE LANDSCAPES.

	Ridge	Canyon	River delta
Number of swarms	1	2	4
Split percentage	100	ca. 50	$33, 50$
Drop frequency	100	100	100
Drop size	$[10, 27.98]$	20	20
Alignment urge	1	1	1
Cohesion urge	$[2.21, 6]$	8	$[1.25, 6]$
Random urge	$[1, 9.90]$	1	$[1]$
Separation urge	$[1, 11.54]$	1	$[1, 2.40]$
Target urge	2	6	2

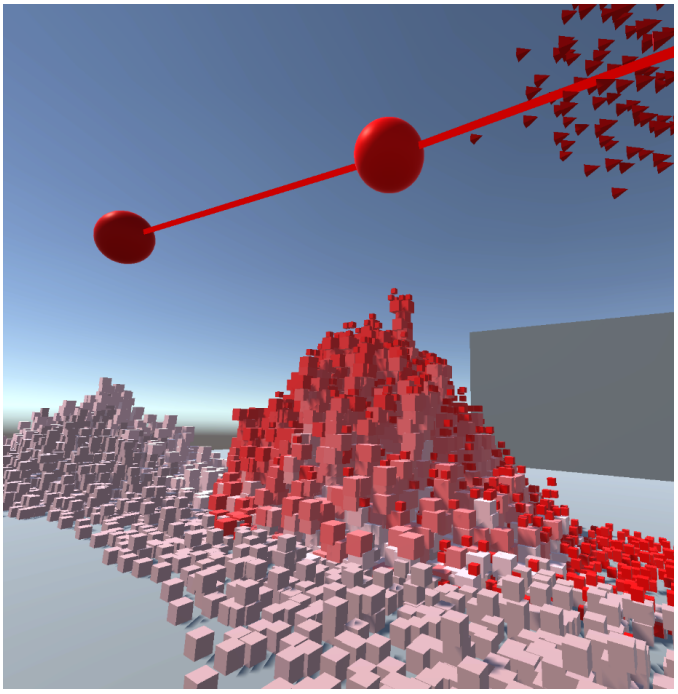
V. Exploration of Swarm-Based Terrain Formation

In this section, we show several examples of terrain formations that can be created using the proposed model (Section IV), construction behavior (Section IV-B), and configuration options (Section IV-C). To explore the versatility of our approach, we chose to simulate three distinct landscape formations that occur in nature. These three landscapes are a mountain ridge, a canyon, and a river delta. To achieve the differences in appearances, we tweak the boid urges and construction parameters accordingly. An overview of the parameter values we used to generate our three example landscapes, and their min and max values, is provided in Table I.

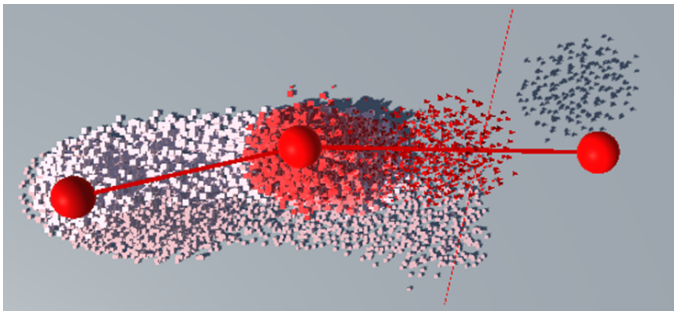
A mountain ridge can have several peeks that vary in height, width, and distance. To generate such a landscape, we focus on the drop size, cohesion urge, random urge, separation urge, and distance between control points. We use a single swarm to generate the ridge. A possible result can be seen in Fig. 4. We could also have used the drop frequency to control the height of the mountain peaks. However, the same results can be achieved by changing the simulation speed. We did not snap the cubes to a grid to generate a smoother surface.

A canyon is characterized by a valley that is surrounded by steep mountains. Therefore, we use our split mechanic to split up an initial swarm into two distinct subswarms, which we control individually. We use the target and cohesion urges to reduce the spread of the generated cubes. Reducing the target urge leads to a higher urge for the boids to fly towards the control point. This means, if the urge is very low, the boid-swarm orbits the control point by a higher radius. As higher the urge gets, as nearer do the boids stick to the control point. That means a low target-urge-factor leads to more scattering of cubes around the control point. Increasing the cohesion urge leads to a more compact formation of the boid-swarm, which leads to less scattering as well.

Additionally, we use the snap to grid option (see Section IV-B). This leads to a steep landscape (Fig. 5). To generate more realistic canyons, several terraces could be generated by splitting both swarms into further subswarms that use different drop frequencies. This would result in different terrain heights. Also, by introducing and carefully placing more control points,



(a)



(b)

Fig. 4. (a) Perspective and (b) top-down view of a mountain ridge generated from a single swarm. Swarm and construction parameters are changed for each control point, resulting in distinct mountain peaks.

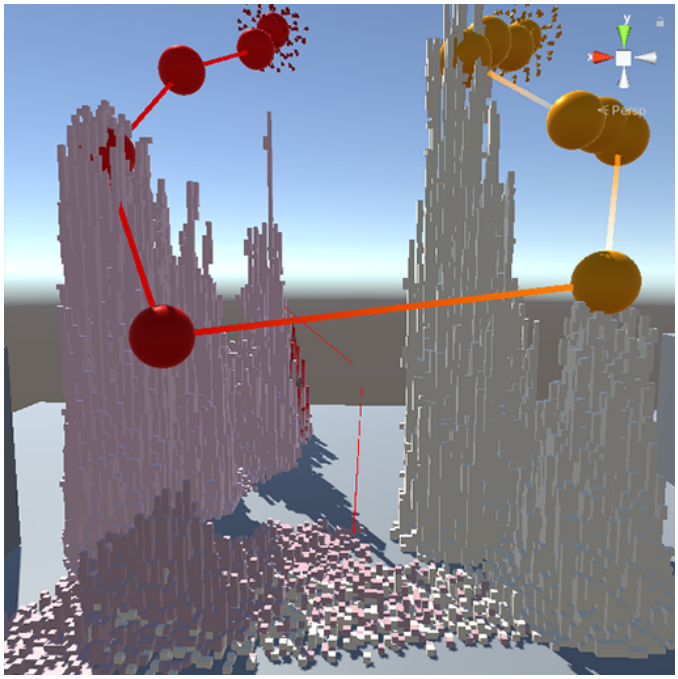
meander-shaped canyons can be created. For now, we refrained from generating a more detailed landscape that includes terraces to not degrade the performance too much.

As our final example we present a river delta. This is an apt use-case for our split and merge mechanics. The individual strands of water can each be generated by a different subswarm. Every (sub-)swarm can have its individual set of values for all parameters we introduced earlier. Thus, resulting in a great variety of branches that can further split up or merged with one another. To give an example, this branches can be used to create small tributaries (yellow), which connect the 2 bigger rivers (red, orange). The thickness of the branch is regulated by the number of agents.

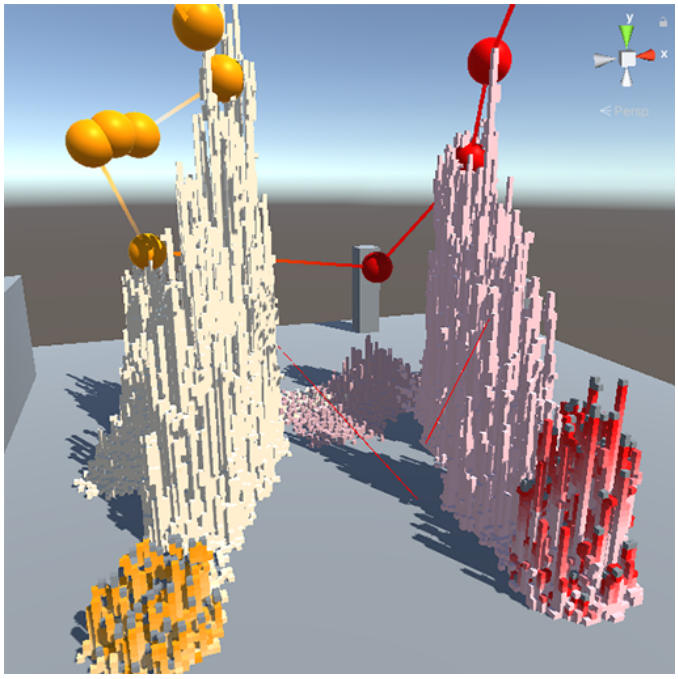
In Fig. 6, we show the generation of such a river delta.

VI. Summary and Future Work

In the previous section we discussed some of the landscape types that can be generated using our swarm guidance approach. In this section, we first summarize our work and then discuss possible future work.



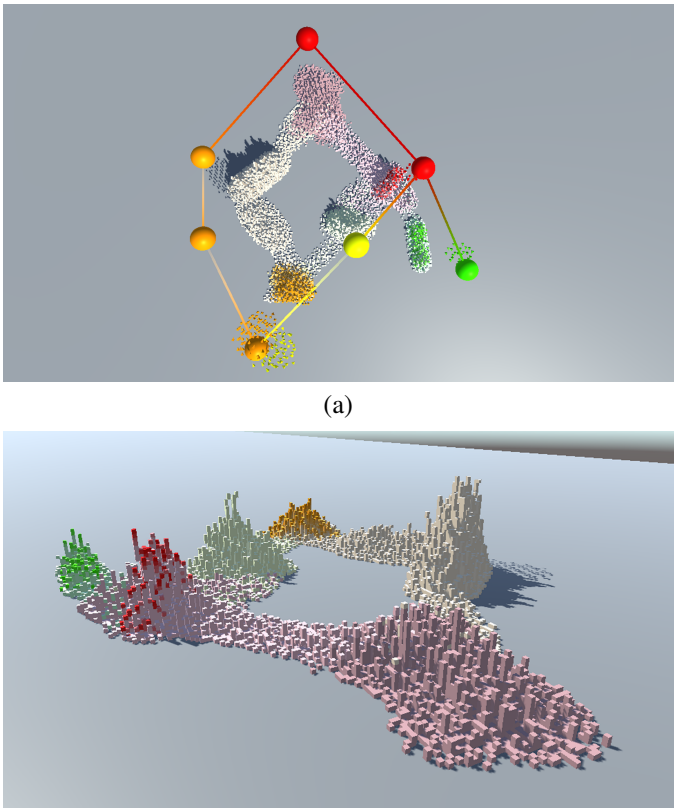
(a)



(b)

Fig. 5. (a) Front and (b) back view of a canyon generated by two subswarms.

In this paper we presented a stigmergic approach to guide a self-organized construction process. Basing our requirements of a previously published taxonomy on interactive self-organizing system, we used a 3DUI in VR to guide simplified boid swarms. Therefore, we used a ring menu to adapt boid urges and construction parameters at control points that can be



(a)

(b)

Fig. 6. (a) Top and (b) front view of a river delta generated by many subswarms. The yellow and orange swarm got merged again. In this scenario, the generated cubes represent water instead of rock.

placed in the virtual environment. Swarms can be split up and merged at control points. To generate landscapes, we adopted a particle deposition approach. We presented and compared three distinct example landscapes that can be generated using our approach: a mountain ridge, a canyon, and a river delta.

Several performance-wise improvements are conceivable that would allow for both more boid agents and larger generated structures. The need for good performance becomes especially clear in the context of VR, whereby lag could cause cybersickness [19]. To improve performance, an Entity-Component-System (ECS) could be used. To that end, Unity's Data-Oriented Technology Stack (DOTS) seems to be a natural choice. Also, in an effort to reduce the vertices in the scene, the generated structures could be simplified once they become static. Furthermore, there are several ways the user interaction and swarm guidance could be extended. The radial menu presented in Section IV-C could be augmented with further configuration parameters for both the swarm behavior and generation. Exploring alternative interaction techniques, e.g. gesture-based approaches, could possibly be also beneficial. We'd also be very interested in other formations and artefacts that can be generated using the presented approach, or extensions thereof. For example, our approach may also be used to generate cities: Streets could be generated using a similar approach as our river delta scenario, whereas other swarms could drop houses. To detect valid positions, the house-building swarm agents could implement local behavior, similar to [20], e.g. to ensure all houses are adjacent to a street.

References

[1] C. Müller-Schloer, H. Schmeck, and T. Ungerer, eds., *Organic Computing - A Paradigm Shift for Complex Systems*. Autonomic Systems, Birkhäuser Verlag, 2011.

[2] P.-P. Grassé, "La reconstruction du nid et les coordinations interindividuelles chezbellicositermes natalensis etcubitermes sp. la théorie de la stigmergie: Essai d'interprétation du comportement des termites constructeurs," *Insectes sociaux*, vol. 6, no. 1, pp. 41–80, 1959.

[3] V. Gerling and S. von Mammen, "Robotics for self-organised construction," in *2016 IEEE 1st International Workshops on Foundations and Applications of Self* Systems (FAS* W)*, pp. 162–167, IEEE, 2016.

[4] N. K. Long, K. Sammut, D. Sgarioto, M. Garratt, and H. A. Abbass, "A comprehensive review of shepherding as a bio-inspired swarm-robotics guidance approach," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 4, no. 4, pp. 523–537, 2020.

[5] K. H. Petersen, N. Napp, R. Stuart-Smith, D. Rus, and M. Kovac, "A review of collective robotic construction," *Science Robotics*, vol. 4, no. 28, 2019.

[6] S. von Mammen and J. Taron, "A trans-disciplinary program for biomimetic computing and architectural design," *ITcon: Special Issue CAAD and Innovation*, vol. 17, pp. pp. 239–257, September 2012.

[7] M. K. Heinrich, S. von Mammen, D. N. Hofstadler, M. Wahby, P. Zahadat, T. Skrzypczak, M. D. Soorati, R. Krela, W. Kwiatkowski, T. Schmickl, *et al.*, "Constructing living buildings: a review of relevant technologies for a novel application of biohybrid robotics," *Journal of the Royal Society Interface*, vol. 16, no. 156, p. 20190238, 2019.

[8] J. J. LaViola Jr, E. Kruijff, R. P. McMahan, D. Bowman, and I. P. Poupyrev, *3D user interfaces: theory and practice*. Addison-Wesley Professional, 2017.

[9] R. Skarbez, M. Smith, and M. C. Whitton, "Revisiting milgram and kishino's reality-virtuality continuum," *Frontiers in Virtual Reality*, vol. 2, p. 27, 2021.

[10] R. Dachselt and A. Hübner, "Three-dimensional menus: A survey and taxonomy," *Computers & Graphics*, vol. 31, no. 1, pp. 53–65, 2007.

[11] S. Gebhardt, S. Pick, F. Leithold, B. Hentschel, and T. Kuhlen, "Extended pie menus for immersive virtual environments," *IEEE transactions on visualization and computer graphics*, vol. 19, no. 4, pp. 644–651, 2013.

[12] N. Shaker, J. Togelius, and M. Nelson, "Procedural content generation in games," 2014.

[13] E. Bonabeau, D. d. R. D. F. Marco, M. Dorigo, G. Théraulaz, G. Theraulaz, *et al.*, *Swarm intelligence: from natural to artificial systems*. No. 1, Oxford university press, 1999.

[14] P. Gruber, "Biomimetics in architecture [architekturbionik]," in *Biomimetics–Materials, Structures and Processes*, pp. 127–148, Springer, 2011.

[15] E. Galin, E. Guérin, A. Peytavie, G. Cordonnier, M.-P. Cani, B. Benes, and J. Gain, "A review of digital terrain modeling," in *Computer Graphics Forum*, vol. 38, pp. 553–577, Wiley Online Library, 2019.

[16] S. von Mammen, "Self-organisation in games, games on self-organisation," in *2016 8th International Conference on Games and Virtual Worlds for Serious Applications (VS-GAMES)*, pp. 1–8, IEEE, 2016.

[17] C. W. Reynolds, "Flocks, herds, and schools: A distributed behavioral model," *Computer Graphics*, vol. 21, no. 4, pp. 25–34, 1987.

[18] J. Mayor, L. Raya, and A. Sanchez, "A comparative study of virtual reality methods of interaction and locomotion based on presence, cybersickness and usability," *IEEE Transactions on Emerging Topics in Computing*, 2019.

[19] J. J. LaViola, "A discussion of cybersickness in virtual environments," *SIGCHI Bull.*, vol. 32, p. 47–56, Jan. 2000.

[20] S. Truman and S. von Mammen, "Interactive self-assembling agent ensembles," in *Clash of Realities Conference: Game Technology Summit*, in press 2021.