An interactive simulation of control and coordination strategies for swarms of autonomous construction robots

Towards a post-carbon re-imagining of the interplay between natural and built environments, facilitated by bio-inspired robotic technology

Jordan H. Boyle [1]

[1] *Delft University of Technology, Faculty of Industrial Design Engineering (Netherlands)*

Abstract

There is an established idea – found in science fiction, architectural studios, and scientific papers alike – of stainable buildings crafted from bio-based materials, colonized by plant and animal life, and blending seamlessly into the natural surroundings. Such buildings might one day be built, maintained and remodelled by swarms of autonomous robots, allowing them to evolve in response to the changing needs of their inhabitants. Inspired by that vision, this paper contributes to the field of swarm intelligence with a focus on robotic construction and human-swarm interaction. Along with a short literature review on robotic building, swarm intelligence and biocompatible building materials, the paper presents an open-source simulation of abstracted termite-like swarm construction. The focus is mainly on human-swarm interaction, specifically how to influence the emergent behaviour of an autonomous swarm in order to elicit a desired outcome while retaining the robustness and adaptability of a self-organized system. The simulator is used to demonstrate a set of four autonomous swarm behaviours that are representative of construction tasks.

Keywords

Swarm robotics, robotic additive manufacturing, emergent behaviour, computational modelling, termites

DOI

https://doi.org/10.47982/spool.2024.1.01

1 Context

The mainstream construction industry is recognised as being unsustainable, leading to a growing interest in 'green building'. In the short to medium term, the most achievable approaches include the introduction of more sustainable materials, more efficient use of materials, and an emphasis on circularity of material use (Munaro et al., 2020). But other emerging technologies like additive manufacturing (Paolini et al., 2019), robotics and artificial intelligence (Debrah et al., 2022) also have potential to contribute in the longer term. Within robotics, an interesting emerging approach is the use of multi-robot teams or swarms (Dias et al., 2021; Petersen et al., 2019), which gain efficiency through parallelism and robustness through redundancy and self-organisation.

Combining all the above-mentioned approaches, one can envision a future in which buildings are constructed by autonomous robots from natural materials, becoming habitats for plant and animal life and utilizing the associated natural processes to help keep them cool and improve air quality. Construction, maintenance, remodelling and extension could be continuously performed by swarms of robots, influenced by the evolving needs of the building and its inhabitants. As suggested by Wiesenhuetter et al. (2016), the conventional perspective of buildings progressing linearly through design, construction, use and demolition phases could be reframed as an ongoing evolution in which design, construction and use occur simultaneously.

As a step in that direction, this paper starts by briefly reviewing a selection of prior work in the domains of robotic building, swarm intelligence and environmentally friendly materials. It then goes on to present an original contribution in the form of an interactive swarm building simulation, which is used to demonstrate a number of interesting mechanisms for human-guided swarm coordination.

2 Related work: Robot Building

A substantial body of prior work on robotic building exists, much of which is framed in a construction industry context. For a more thorough treatment of that field than space allows here, the reader is referred to Xu et al. (Xu et al., 2022) and Petersen et al. (Petersen et al., 2019). The use of robotics in construction can be divided into two main classes of approach, namely robotic assembly and robotic additive manufacturing (AM). Examples of robotic assembly include Bier et al. (Bier et al., 2020), which presents a `design to robotic production and assembly' approach, including a large-scale physical prototype made of wooden beams and robotically milled panels, connected by 3D printed nodes. Similarly, Chiang et al. (Chiang et al., 2018) considers computational design, fabrication and robotic assembly as a single scheme. A segment of a larger freeform structure is robotically assembled by stacking rectangular beams, with the assembly sequence optimised to ensure that the partially built structure is stable. At smaller scale, the TERMES project set out to develop `robotic termites' (Werfel et al., 2014). The system consists of a swarm of mobile robots that build structures from pre-manufactured, roughly cuboid blocks whose geometry facilitates alignment and interlocking. Each robot is capable of carrying and placing one block at a time. The system has a centralised controller (Deng et al., 2019) that uses an offline `compiler' to pre-processes the desired structure and derive an optimal assembly sequence. Once the sequence of block placements is determined the individual robots execute it autonomously through decentralized control. In a similar vein, Allwright et al. (2014, 2019) present a multi-robot construction system that builds structures from pre-manufactured cubes. These robots cannot climb the structure, and instead have a crane-like mechanism that allows them to stack blocks up

to three high. The control strategy is abstractly but strongly bio-inspired, in that it is fully decentralized and uses behavioural rules for individual robots combined with stigmergy (indirect coordination via changes to and sensing of the environment, which in termites is believed to be mediated in part by pheromones). The building blocks contain microcontrollers and multicolour LEDs, allowing them to display different colours under different conditions in lieu of a pheromone signal. Finally, flying robots have also been developed for building structures. Compared to ground robots they have the significant advantage of being able to move in 3D, but the disadvantage of lower payload capacity and greater energy expenditure. Augugliaro et al. (2013) use a quadcopter carrying a spool of rope to build tensile structures by wrapping the rope around anchor points, while Willmann et al. (2012) use a team of four quadcopters to build a 6m tall tower out of lightweight rectangular blocks.

Examples of robotic AM include Oskam et al. (2022), which presents the computational design and robotic AM (using a bio-based material) to produce 'plantetoids' intended as habitats for animal and plant life. Tiryaki et al. (2019), Sustarevas et al. (2018) and Rivera et al. (2021) all present different variants on the concept of a conventional mobile robot platform carrying an industrial robot arm equipped with a material extruder. To varying degrees, they couple the kinematics of the mobile base with those of the arm, enlarging the build space beyond that of the arm alone. Zhang et al. (2018) further considers a 'team' of two robots and addresses the control and sequencing challenges associated with them collaborating to build a single structure. Finally, Zhang et al. (2022) presents an impressive 'Aerial-AM' system consisting of 'BuilDrones' and 'ScanDrones'. The former are multirotor UAVs equipped with a lightweight actuated arm (for fine position control) and extruder, which deposits a lightweight cementous-polymeric composite. The latter are UAVs equipped with 3D scanners to monitor print quality and correct for deviations. The system is demonstrated printing simple structures like a cylindrical tower.

3 Related work: Swarm Intelligence in Architectural Design

The field of swarm intelligence explores the underlying mechanisms and applications of self-organizing multi-agent systems. It is inspired by the behaviour and organizing principles of biological swarms, as described in the seminal book by Bonabeau et al. (1999). Swarm robotics embodies swarm intelligence in physical agents (sometimes simulated 'physical agents'), and Dias et al. (2021) provides a good review of this topic. Swarm intelligence concepts have been applied to good effect for developing optimization algorithms (Tang et al., 2021) for various applications including architectural design. Buus (2006) adapts earlier models of termite-inspired building to elicit design of 'human-like architecture', which they characterize by key features like straight walls, right angled corners and openings for doors and windows. Von Mammen & Jacob (2008) use artificial evolution to tune the behaviour of an abstract swarm model inspired by bird flocking to create 'architectural idea models'. The focus is on form, not physics, and the resulting 3D structures have striking and unconventional shapes. Wiesenhuetter et al. (2016) discusses the role of swarm intelligence in architecture, and argues that it could be used as a tool for optimizing a design based on one or more specific metrics, or as a means of creating adaptive structures that respond to environmental or other changes. Finally, Agirbas (2019) uses the software combination of Grasshopper and Rhino, along with an add-on called Locust which implements a swarm-based optimisation algorithm, to design Non-Euclidian geometries for a building façade that are optimised to give desired lighting conditions in the building.

4 Related work: Materials

Materials are not the focus of this paper, but a few potential examples are provided here. Humans have made buildings from `rammed earth' (compressed sand, gravel, silt and clay) for thousands of years, and the fact that some of these buildings still stand is testament to the quality of this material. Another interesting approach is the use of microbially-induced calcite precipitation, in which ureolytic (Farrugia et al., 2019) or photosynthetic (Heveran et al., 2020) bacteria create biominerals (usually CaCO3), between grains of sand or soil, thereby increasing stiffness and strength. One could also potentially use living materials as explored in Camere & Karana (2018). The most plausible example they discuss is mycelium, which can be grown in various different organic substrates (including waste plant matter) and forms a solid network within that bulk material as it grows. Heinrich et al. (2019) discusses the possibility of creating `living buildings' by integrating biological organisms into automated construction tasks. They suggest using a combination of static scaffolds, biological organisms and manual manipulation (which could be performed by robots) to shape the growth of the biological elements as desired.

5 Innovations

The key benefits of swarms, namely robustness, flexibility and scalability (Dias et al., 2021) are due in large part to the use of distributed, self-organizing control strategies. But this approach also makes it hard to 'design' a desired group level behaviour, because that behaviour emerges through the interaction of the agents rather than being explicitly programmed. In the context of swarm construction, it is similarly challenging to develop control and coordination strategies that lead to the swarm following the specification for a structure that the human operator wants them to build. When developing control strategies for swarms, engineers often take inspiration from biological phenomena like ant foraging or termite nest construction, but these natural systems certainly don't support external user input! The goal of swarm engineering is to develop approaches for designing desired swarm behaviour, which often involves abstracting and modifying biological mechanisms (Brambilla et al., 2013).

The work presented here falls under the swarm engineering umbrella but focusses mainly on human-swarm interaction, motivated by the question: How can a human 'conductor' influence the emergent behaviour of an autonomous swarm in order to elicit a desired outcome while retaining the robustness and adaptability of a self-organized system?

To that end, an interactive simulator was developed (see Methods) and a set of agent behaviours were implemented (see Results). The novelty of the work lies in how the biological concepts of stigmergic and template-based building (Perna & Theraulaz, 2017) were abstracted and applied. Combining various technologically feasible sensory modalities with user-input, the swarm performs several representative construction tasks.

6 Results

In line with most agent-based simulations, the model is an abstraction and simplification of actual swarm robotic construction. Its purpose is to develop and test organizational principles that could be translated to real robot builder swarms. Each experiment demonstrates a distinct mechanism that enables a group of agents, with only local sensing and decentralized control, to coordinate their efforts, generate large-scale structure, perform tasks that would be useful in a real construction context, and respond to human input. More details on the behaviours used in each of these experiments is provided in Methods.

6.1 Terrain levelling

This experiment, illustrated in Figure 1, represents a pre-construction phase where uneven terrain must be levelled before it can be built on.

In the simulation, agents can perceive a 3D depth map of their immediate surroundings. They assess the terrain's curvature at their current location (refer to Methods). If they identify a mound, they are inclined to collect material, while if they are in a depression, they might deposit material. In both scenarios, the likelihood of action aligns approximately with the local curvature, i.e., they're more likely to add or remove material if the mound or depression is more pronounced. Gradually, the terrain becomes progressively flatter. This mechanism is loosely inspired by research showing that local surface curvature influences soil displacement by termites (Calovi et al., 2019).

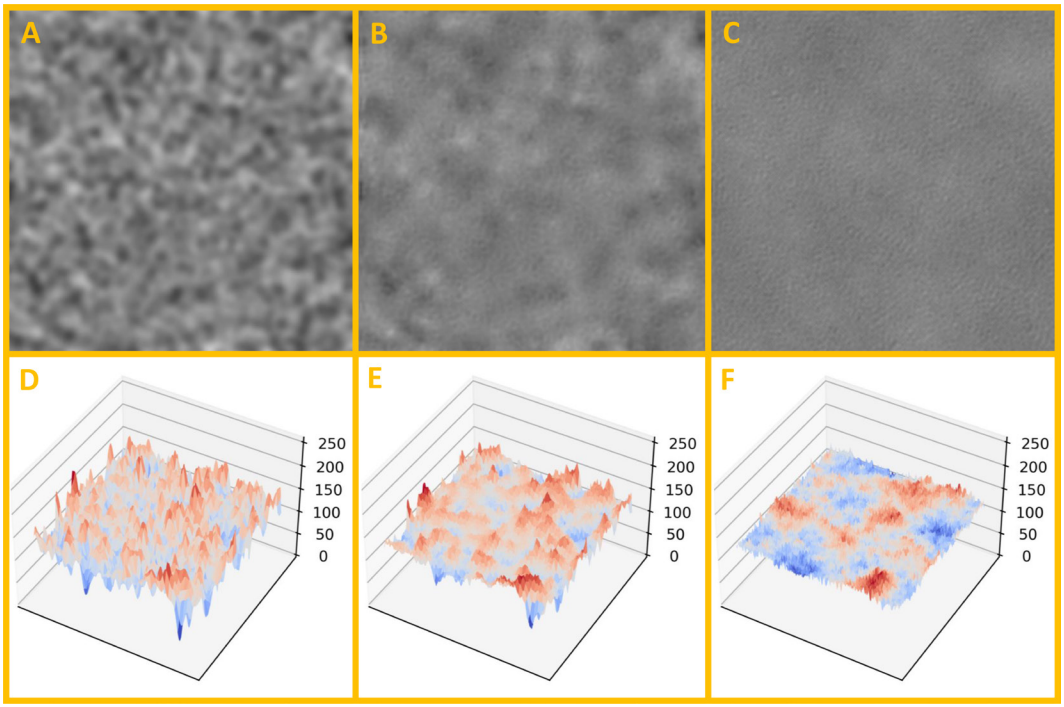

FIGURE 1 Initially random, lumpy terrain become increasingly smooth as the swarm fills depressions and excavates mounds. Panels A – C and D – F show the same snapshots in time (t = 0s, t = 30s and t = 180s), with the top row showing the 2D greyscale representation from the simulator itself (where lighter colours correspond to higher elevation), and the bottom row showing the same data as a 3D mesh plot.

6.2 Building a large-scale structure

This experiment, illustrated in Figure 2, represents the initial phase of constructing a building within a 'footprint' defined by a human architect. It addresses the challenge: how do small individual agents, relying on local sensing, ensure the large-scale geometric accuracy of their build? The mechanism deployed here mirrors the termite's 'Royal Chamber' construction, known to be facilitated by a queen-specific pheromone (Bonabeau et al., 1998).

The simulation employs a pheromone map (as described in Methods), which could be physically realized using a detectable chemical like ethanol, signal-emitting beacons (e.g., radiofrequency or audio), or GPS. In the experiment, the user designs the 'building footprint' via the software's UI. This design is translated into a 'blueprint pheromone' template, exhibiting a blurry transition between 'build' and 'no build' zones, representative of chemical diffusion or sensor uncertainty.

Agents conduct a random walk, stochastically depositing or picking up material based on sensed pheromone concentration (see Methods). Over time, a 'building' forms according to the plan. Despite the template's blurry boundaries and stochastic deposition, the structure emerges with defined edges due to the convergence of many random events on the probabilistically expected value.

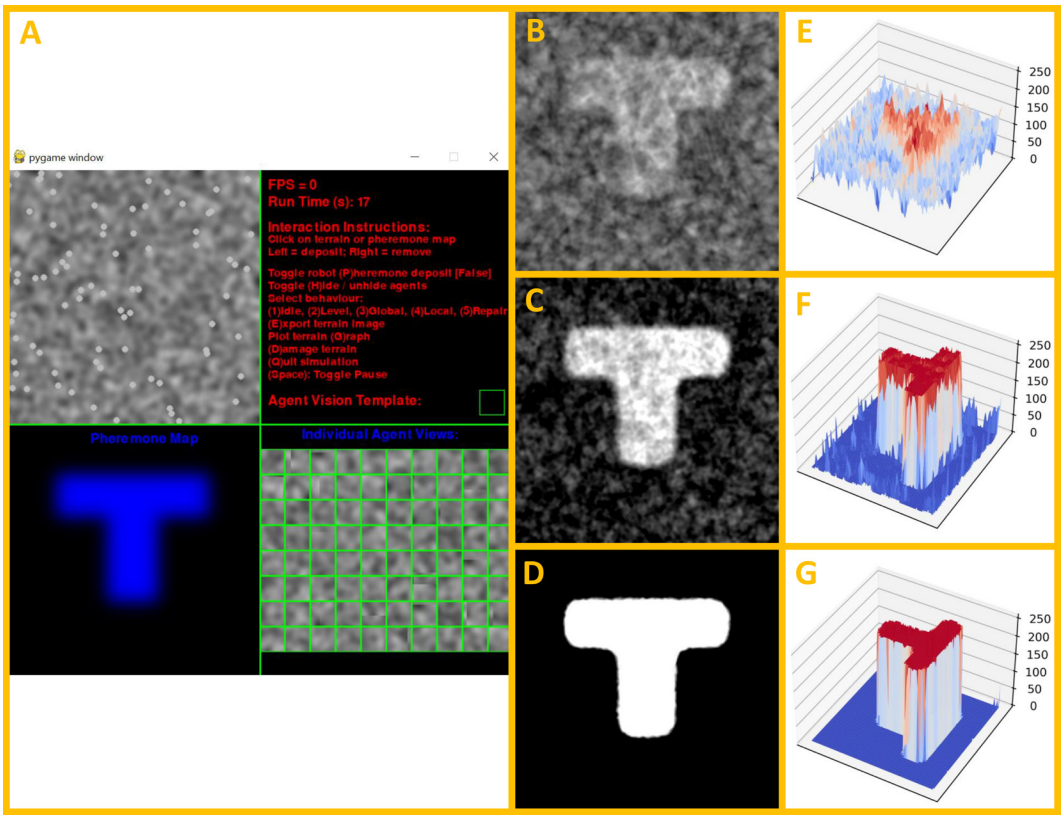

FIGURE 2 A 'building' defined by a blueprint pheromone template is constructed on initially random terrain. Panel A shows a screenshot of the simulator (top left, initial agent positions and terrain map; bottom left, pheromone template; top right, user instructions; bottom right, individual agent views). Panels B – D and E – G show the same snapshots in time (t =3 0s, t = 60s and t = 180s), with the left column showing the 2D greyscale representation from the simulator itself (where lighter colours correspond to higher elevation), and the right column showing the same data as a 3D mesh plot.

While the emergence of the building's flat top and surrounding ground appears as a simulation artefact due to a numerical range limit on the terrain map, in a physical setting, equipping robots with an altitude sensor to control the build and excavation heights would offer a practical way to achieve the same result.

6.3 Creating small-scale features

These experiments, illustrated in Figure 3 – 5, represent a later construction phase where small scale features must be added to a larger structure (like surface texture, ventilation holes or windows), or a repeating pattern needs to be applied across a large area (such as constructing support pillars, digging holes for plants, or creating drainage channels).

In this behavioural state, agents 'see' their surroundings (i.e., obtain a 3D depth map) and compare that view to a small-scale 'vision template' (inset in panel A of each figure), designed by the human user using the software's UI and representing the desired appearance of an equivalent small area. The agents perform a random walk and continually compare their observable area to the 'vision template'. They also 'envision' the outcome of adding or removing material and compare these two hypothetical cases to the 'vision template' as well (see Methods). If such actions improve the match to the template, they are executed. Gradually, the desired repeating pattern, albeit with some variability, manifests across the entire terrain map.

This variability results from stochastic agent behaviour and their localized, rather than global, perception. The degree of variability partially depends on the template's nature. Figure 3 shows a template representing an isolated feature surrounded by empty space (e.g., a pillar). Here, sensing range and template provide sufficient information for approximate pillar spacing but not for precise alignment. Figure 4 depicts a continuous feature template (e.g., rows of trenches), providing better alignment information. However, errors can still occur, primarily when agents independently initiate the pattern in different areas during early construction stages. Figure 5 depicts an experiment using the same template as in Figure 4, but this time the agents were initialised clustered in the middle of the world, and the human user guided the swarm by placing pheromone. This more methodical construction approach reduces improves overall pattern alignment[1].

1 Another repeat of the experiment shown in Figure 5, with user guidance of the swarm, is shown in Supplementary Video 1 (https://youtu.be/p9uJOsB9LCg).

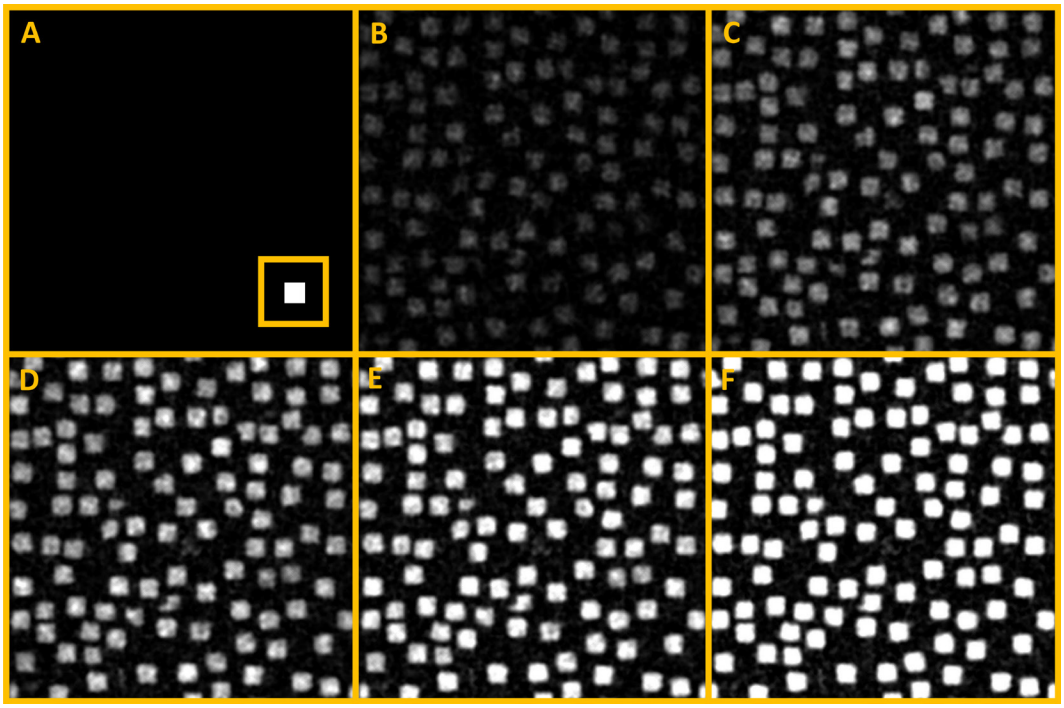

FIGURE 3 Construction of 'pillars' based on a local vision template (inset in Panel A). Panels A to F show snapshots of the construction process in increments of 30s.

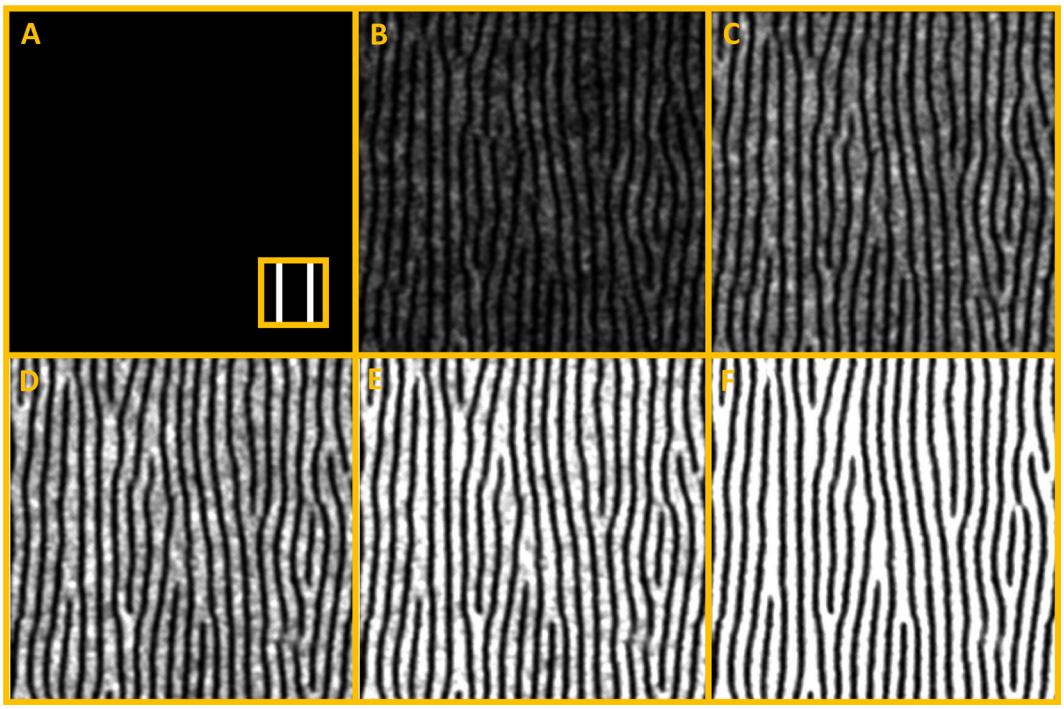

FIGURE 4 Construction of 'trenches' based on a local vision template (inset in Panel A). Panels A to F show snapshots of the construction process in increments of 30s.

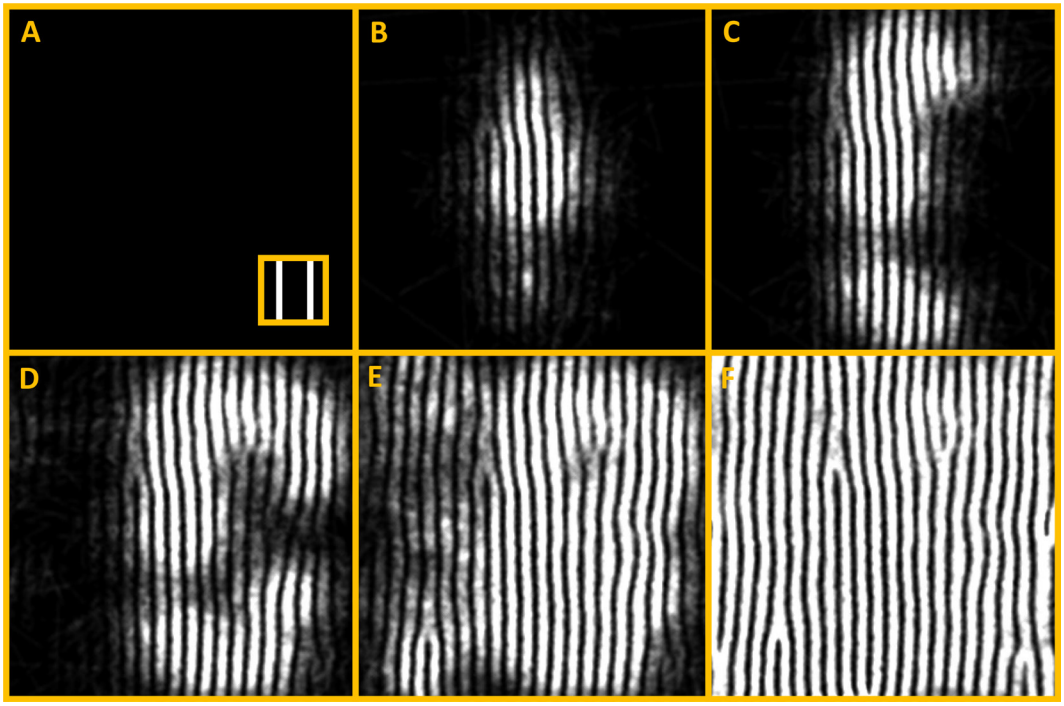

FIGURE 5 Construction of 'trenches' based on a local vision template (inset in Panel A), with the building activity of the swarm guided by a human overseer. Panels A to F show snapshots of the construction at 0s, 30s, 60s, 90s, 150s and 240s. Note how construction progresses through space as the swarm moves.

6.4 Repairing a damaged structure based on local curvature sensing

This experiment, illustrated in Figure 6, is a simplified representation of a post-construction phase where agents maintain an existing building. The concept involves agents initially exploring the intact structure with local sensing to learn what 'normal' looks like. Here, that involves memorizing the structure's maximum (most positive) and minimum (most negative) curvature. In a real-world application, training a neural network to recognize a range of 'normal' local features through computer vision techniques would enable a more complex version of this.

Once familiar with the intact structure, the agents are switched to 'maintenance mode'. They start investigating the structure for anomalies that diverge from their learned experience. In this simulation, anomalies are characterized by local curvature surpassing previously encountered limits. 'Damage' to the structure, represented by random lumps and pits, can be introduced by the user. On detecting these anomalies, agents add or remove material to 'repair' the structure.

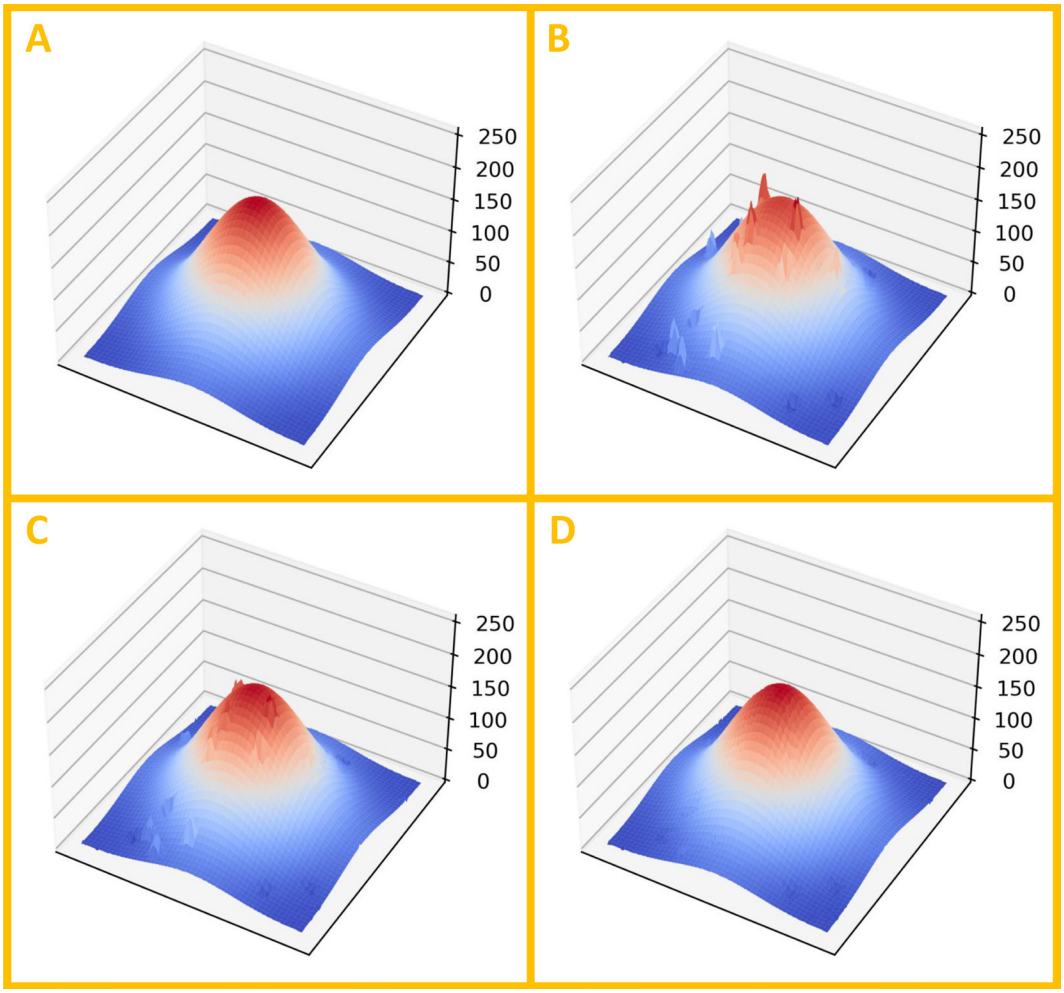

FIGURE 6 Repair of a damaged structure by the swarm. In this case only 3D mesh plots are shown, because these provide a better visualisation. A) The original, undamaged structure, which the swarm 'learns' the features of. B) Damage is introduced, in the form of localised lumps and holes in the surface. C) The swarm has begun recognising these anomalies and repairing them. D) Repair complete.

7 Conclusions

This work presents promising algorithms that combine sensing and user input to achieve human-directed construction by a simulated robot swarm. A user-defined global spatial template encoded in the intensity of a detectable quantity (e.g., chemical concentration, light intensity, electromagnetic field), sampled locally by the agents and used to modulate the probability of picking up or depositing material, was shown to successfully impose large-scale structure. Short-range sensing of 3D shape around the agent (achievable with laser scanning, time-of-flight cameras or structured light sensors) compared to a 'desired' local shape (learned or user-specified) and used to influence pickup and deposit probabilities was also shown to be effective. As agents move through space, continuously applying these 'local templates', order can be imposed at scales much larger than the sensing radius.

For readers who want to explore these behaviours, experiment with modifications, or create new behaviours, the expandable open-source simulator (GitHub link in Methods) is complex enough to yield intriguing insights while remaining accessible to those with only intermediate Python programming skills.

8 Methods

This paper provides only a concise description of the implementation due to space limitations. For further details, consult the source code[2]. The simulation is built in Python (3.11.3) and uses several standard libraries along with **numpy**, **scipy**, **matplotlib** and **pygame** (which you will need). It runs in real time, offering user control over various simulation features and behaviours. Numerical constants in the code can be easily modified. The presented experiments utilized the provided values. Individual components of the software are described below, and a high-level overview of the architecture is shown in Figure 7.

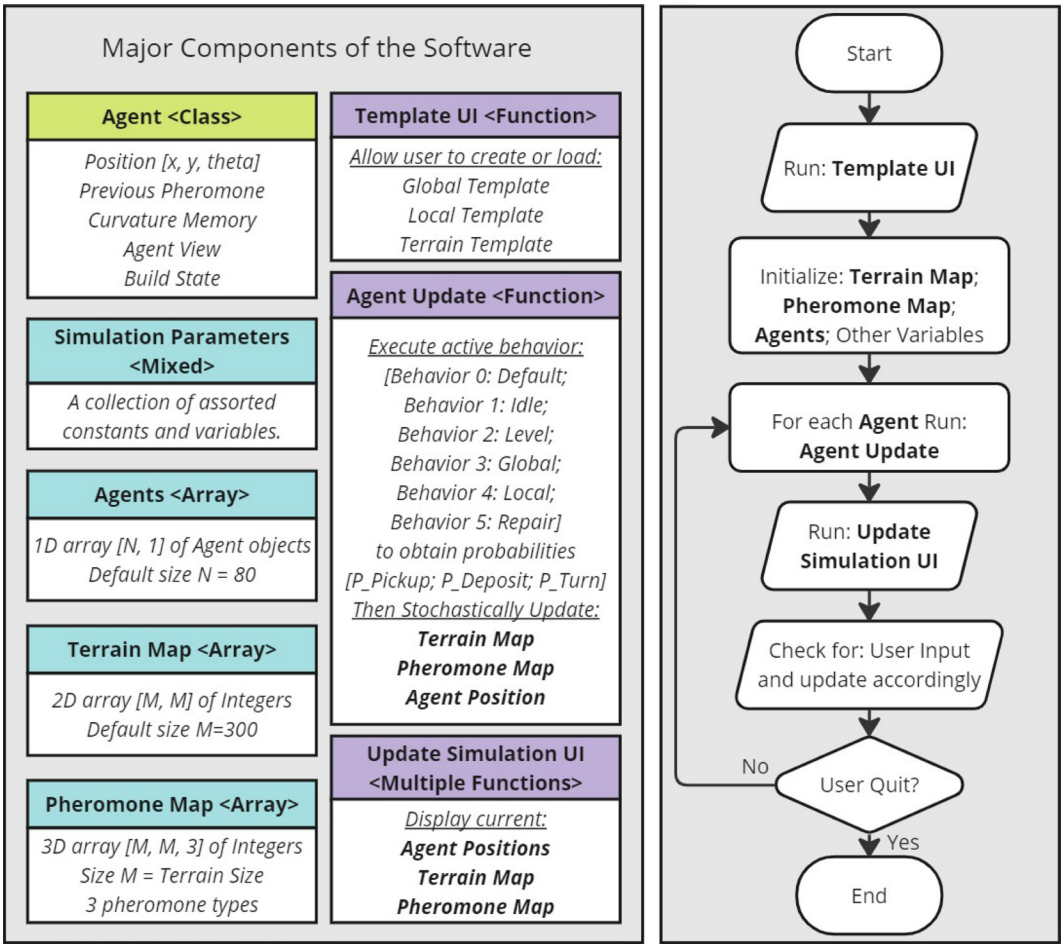

FIGURE 7 Main software components (left) and high-level program flow (right) of the simulator.

2 https://github.com/DrJordanBoyle/Spool_Builder_Swarm_Simulator/tree/main

8.1 World

The *world* is defined by a *terrain map* and a *pheromone map*, which are square grids of square cells. The size of the *world* is specified by a parameter *width* (*width* = 300 here). Cells of the *terrain map* each store one integer in the range [0 – 255], where this value represents the amount of material (i.e. 'height' of the terrain) in that cell. During the simulation the evolving *terrain map* is visualised as a 2D greyscale image where black = 0 and white = 255. At any time, the user can export a snapshot of this 2D image or generate a 3D mesh plot using Python's **matplotlib** library (if you are running the code in an IDE that supports this, like 'Spyder').

Cells of the *pheromone map* each store three integers in the range [0 – 65,535] representing the concentrations of three different pheromone types. When developing the simulation, it was found that the pheromones benefitted from a higher resolution representation than the terrain, hence the different scale.

The boundaries of the world can either be *reflecting* (as if enclosed by a wall) or *wrapping* (as if the world was toroidal), as selected by the user.

8.1.1 World: Initial conditions

The user can select between three options for the initial conditions of *terrain map: empty* (no initial terrain); *random* ('lumpy' terrain created through a combination of random number generation and Gaussian smoothing); *template based* (created by smoothing a binary template drawn by the user and then converting this to terrain values). The user can also choose to create a *blueprint pheromone* template (see Agent Behaviours). Finally, the user can choose whether *agents* are initialised *spread out* (at random positions in the *world*) or not (all grouped in the centre).

8.1.2 World: Updates

The *terrain map* and *pheromone map* are both modified during simulations. *Agents* can *pickup* or *deposit* terrain. When they do so, they decrease or increase (respectively) values of the *terrain map* in a rounded 'blob' (height = 5 terrain units; radius = 5 cells) centred on their current position. The resulting terrain values are clipped to the range [0 – 255] which is admittedly unrealistic but necessary for practical reasons.

The three pheromone types are used differently in the current implementation, though the same capabilities exist for all of them. *Terrain pheromone* is actually only used transiently as a computationally convenient way to achieve *template-based* initialisation of *terrain map*, so it won't be discussed further here. *Blueprint pheromone* underlies one of the agent behaviours described later. It is user generated (through the UI) and the associated parameters have been set such that it does not change while the simulation runs. *Build pheromone* is more complex. If the *agents deposit pheromone* option is active, then whenever an *agent picks up* or *deposits* material, it also adds a circular 'cloud' (peak value at centre = 20 pheromone units; drops off linearly to zero over a radius of 50 cells) of *build pheromone* to the *pheromone map* centred on its current location. In addition, the concentration of *build pheromone* in each cell decays every time step, reducing by an amount proportional to the current concentration. As such, it will gradually disappear in any given location if not replenished. Finally, one of the ways the user can influence the behaviour of the

swarm is by manually adding (peak value at centre ≈ 16,000; drops off linearly to zero over a radius of 100 cells) or removing (set = 0 within radius of 100 cells) *build pheromone* from a specific area by left or right clicking with the mouse.

8.2 Agents

The simulator supports an arbitrary number of *agents* (representing robots). The value of N=80 used here was found to be the maximum number for which the desired frame rate of 30FPS could be consistently achieved on the author's laptop. Agents are stored as an array of instances of an *AGENT* class. *Agent* state is represented by set of class variables:

$$position = \{(x, y) \in \mathbb{R}\}$$

$$orientation = \{\theta \in \mathbb{R}\}$$

$$build\ state = \{idle, level, global, local, repair\}$$

$$previous\ pheromone = \{(Ph_{build}, Ph_{blueprint}, Ph_{terrain}) \in \mathbb{Z}\}$$

$$curvature\ memory = \{(max, min) \in \mathbb{R}\}$$

$$agent\ view = \left\{ \begin{bmatrix} t_{1,1} & \cdots & t_{1,N} \\ \vdots & \ddots & \vdots \\ t_{M,1} & \cdots & t_{M,N} \end{bmatrix} \middle| t_{i,j} \in \mathbb{Z} \right\}$$

The software limits x,y to the range [0 → width] and θ to the range [0 → 2π]. The size of *agent view* is also a parameter that could be changed, but M=N=30 is what the author has always used.

8.2.1 Behaviour 0: Default

This is the default behaviour, and is always active. *Agents* perform a random walk, biased by *build pheromone* and may *pickup* or *deposit* material. First the change in *build pheromone* concentration since the last time step is computed:

$$\Delta Ph_{build} = sense(Ph_{build}) - previous\ pheromone(Ph_{build})$$

Then the turn probability of reorienting is computed based on that change:

$$P_{turn} \begin{cases} 0.2, & \Delta Ph_{build} \leq -5 \\ 0.002, & \Delta Ph_{build} \geq 5 \\ 0.02, & otherwise \end{cases}$$

Then a random number, *X*, is generated from a uniform distribution in the range [0→1] and compared to the turn probability to determine whether the agent reorients. If it does, a turn angle is drawn from a normal distribution with $\mu=\pi$ and $\sigma = \frac{\pi}{4}$ such that:

$$\theta = \begin{cases} \theta + \Delta\theta, & X < P_{turn} \\ \theta, & otherwise \end{cases}$$

Then the *agent* updates its position based on its speed and orientation:

$$x = x + Speed \times \Delta t \times \cos\theta$$

$$y = y + Speed \times \Delta t \times \sin\theta$$

Where *Speed=100 cells/sec* and $\Delta t = \frac{1}{30}$

Finally, two random numbers U and V are generated from a uniform distribution in the range [0→1] and compared to probabilities P_{pickup} and $P_{deposit}$ to determine whether the *agent* executes a *pickup* and *deposit* of material (as described in World: Updates). By default, these probabilities are both zero, but they can be modified by other behaviours, of which exactly one is always active (selected by the user through the UI).

8.2.2 Behaviour 1: Idle

This is a minimal overlay on top of Behaviour 0, during which P_{pickup}=*0* and $P_{deposit}$=*0*. The only addition is that *agents* constantly monitor the local terrain curvature, which is obtained by computing the Laplacian of *agent view*:

$$\mathcal{L} = \nabla^2(agent\ view)$$

And then taking the average of the four elements in the 'middle' of the matrix (which correspond roughly to the *agent's* current position). Assuming the size of *agent view* is 30x30 as used here:

$$Curv = \left(\mathcal{L}_{14,14} + \mathcal{L}_{14,15} + \mathcal{L}_{15,14} + \mathcal{L}_{15,15}\right)/4$$

Finally, this value is compared to the current *max* and *min* values of *curvature memory* and these are updated if necessary:

$$max = \begin{cases} Curv, & Curv > max \\ max, & otherwise \end{cases}$$

$$min = \begin{cases} Curv, & Curv < min \\ min, & otherwise \end{cases}$$

8.2.3 Behaviour 2: Level

First, each *agent* computes the local terrain curvature Curv based on *agent view* as described in Behaviour 1. Then, probabilities P_{pickup} and $P_{deposit}$ are calculated as follows:

$$P_{pickup} = \begin{cases} Curv, & Curv > 0 \\ 0, & otherwise \end{cases}$$

$$P_{deposit} = \begin{cases} 0, & Curv > 0 \\ -Curv, & otherwise \end{cases}$$

8.2.4 Behaviour 3: Global

Each *agent* senses the value of *blueprint pheromone* at its current location and normalises it to the range [0 → 1] by dividing by the maximum possible pheromone value of 65,535. This value is then passed through a sigmoid function:

$$Ph_{norm} = sense(Ph_{blueprint})/65{,}535$$

$$Ph_{sigmoid} = \frac{1}{1 + e^{(-\sigma(Ph_{norm} - \mu))}}$$

Where values of σ=*50* and μ=0.4 are used here. Finally, the probabilities P_{pickup} and $P_{deposit}$ are calculated as follows:

$$P_{deposit} = Ph_{sigmoid}$$

$$P_{pickup} = 1 - P_{deposit}$$

8.2.5 Behaviour 4: Local

Each *agent* samples a local portion of *terrain map*, centred on its current location *(x, y)* to obtain *agent view* which is an *MxN* (30x30) matrix. It then creates two copies of this matrix, which are modified by performing a *pickup* and *deposit* in exactly the same way as is done when modifying *terrain map*. This yields two additional matrices *pickup view* and *deposit view*. Each of these three matrices becomes a *sample* for comparison with the user-generated *view template*. This is achieved using forward and inverse Fast Fourier Transforms from the **numpy.fft** library as follows:

$$templateFFT = FFT2(view\ template)$$

$$sampleFFT = FFT2(sample)$$

$$correlation = iFFT2(conjugate(templateFFT) * sampleFFT)$$

$$match = \max\left(abs(correlation)\right)$$

Note that the match is quantified based on the maximum absolute value within the correlation matrix, which makes the process insensitive to any spatial offset between the sample and template as desired. After doing this for *agent view*, *pickup view* and *deposit view* to obtain *match, pickup match* and *deposit match*, the probabilities are calculated as follows:

$$P_{pickup} = \begin{cases} 1, & pickup\ match > deposit\ match\ AND\ pickup\ match > match \\ 0, & otherwise \end{cases}$$

$$P_{deposit} = \begin{cases} 1, & deposit\ match > pickup\ match\ AND\ deposit\ match > match \\ 0, & otherwise \end{cases}$$

8.2.6 Behaviour 5: Repair

Each agent computes the local terrain curvature *Curv* as per the method described in Behaviour 1, and compares this to the *max* and *min* curvatures learned while executing Behaviour 1. The probabilities P_{pickup} and $P_{deposit}$ are calculated as follows:

$$P_{pickup} = \begin{cases} 2 \times (Curv - 1.1 \times max), & Curv > 1.1 \times max \\ 0, & otherwise \end{cases}$$

$$P_{deposit} = \begin{cases} -2 \times (Curv - 1.1 \times max), & Curv < 1.1 \times min \\ 0, & otherwise \end{cases}$$

User Interface

Upon launching the program, a pre-simulation menu is displayed which allows for the creation or loading of Global (for Behaviour 3), Local (for Behaviour 4) and Terrain (for *template-based* world initialisation) templates, as well as selection of some initial conditions.

Once choices are confirmed, the main screen appears, with the simulation initially paused to allow the user to change settings before agents start moving. This screen is subdivided into four windows: Top left shows the *agents* (they can be hidden to see the terrain better) and current *terrain map*; Bottom left (separate for visual clarity) shows the current *pheromone map* (*build pheromone*: Red, *blueprint pheromone*: Blue, *terrain pheromone*, Green). Top right shows the FPS currently being achieved (this is capped at the target of 30FPS) and simulation time, along with user instructions and some *agent* status information. Bottom right shows a grid of the *agent view* for all *agents* (if a behaviour that uses these is active).

References

Agirbas, A. (2019). Façade form-finding with swarm intelligence. *Automation in Construction*, *99*, 140–151. https://doi.org/10.1016/j.autcon.2018.12.003

Allwright, M., Bhalla, N., El-faham, H., Antoun, A., Pinciroli, C., & Dorigo, M. (2014). SRoCS: Leveraging Stigmergy on a Multi-robot Construction Platform for Unknown Environments. In M. Dorigo, M. Birattari, S. Garnier, H. Hamann, M. Montes De Oca, C. Solnon, & T. Stützle (Hrsg.), *Swarm Intelligence* (Bd. 8667, S. 158–169). Springer International Publishing. https://doi.org/10.1007/978-3-319-09952-1_14

Allwright, M., Zhu, W., & Dorigo, M. (2019). An open-source multi-robot construction system. *HardwareX*, *5*, e00050. https://doi.org/10.1016/j.ohx.2018.e00050

Augugliaro, F., Mirjan, A., Gramazio, F., Kohler, M., & D'Andrea, R. (2013). Building tensile structures with flying machines. *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 3487–3492. https://doi.org/10.1109/IROS.2013.6696853

Bier, H., Hidding, A., & Galli, M. (2020, Oktober 14). *Design-to-Robotic-Production and -Assembly for Architectural Hybrid Structures*. 37th International Symposium on Automation and Robotics in Construction, Kitakyushu, Japan. https://doi.org/10.22260/ISARC2020/0207

Bonabeau, E., Dorigo, M., & Theraulaz, G. (1999). *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press. https://doi.org/10.1093/oso/9780195131581.001.0001

Bonabeau, E., Theraulaz, G., Deneubourg, J., Franks, N. R., Rafelsberger, O., Joly, J., & Blanco, S. (1998). A model for the emergence of pillars, walls and royal chambers in termite nests. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, *353*(1375), 1561–1576. https://doi.org/10.1098/rstb.1998.0310

Brambilla, M., Ferrante, E., Birattari, M., & Dorigo, M. (2013). Swarm robotics: A review from the swarm engineering perspective. *Swarm Intelligence*, *7*(1), 1–41. https://doi.org/10.1007/s11721-012-0075-2

Buus, D. P. (2006). *Constructing Human-Like Architecture with Swarm Intelligence. Thesis, Aalborg University Department of Computer Science*. https://projekter.aau.dk/projekter/files/61068317/1154483391.pdf

Calovi, D. S., Bardunias, P., Carey, N., Scott Turner, J., Nagpal, R., & Werfel, J. (2019). Surface curvature guides early construction activity in mound-building termites. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *374*(1774), 20180374. https://doi.org/10.1098/rstb.2018.0374

Camere, S., & Karana, E. (2018). Fabricating materials from living organisms: An emerging design practice. *Journal of Cleaner Production*, *186*, 570–584. https://doi.org/10.1016/j.jclepro.2018.03.081

Chiang, Y.-C., Bier, H., & Mostafavi, S. (2018). Design to Robotic Assembly: An Exploration in Stacking. *Frontiers in Digital Humanities*, *5*, 23. https://doi.org/10.3389/fdigh.2018.00023

Debrah, C., Chan, A. P. C., & Darko, A. (2022). Artificial intelligence in green building. *Automation in Construction*, *137*, 104192. https://doi.org/10.1016/j.autcon.2022.104192

Deng, Y., Hua, Y., Napp, N., & Petersen, K. (2019). A Compiler for Scalable Construction by the TERMES Robot Collective. *Robotics and Autonomous Systems*, *121*, 103240. https://doi.org/10.1016/j.robot.2019.07.010

Dias, P. G. F., Silva, M. C., Rocha Filho, G. P., Vargas, P. A., Cota, L. P., & Pessin, G. (2021). Swarm Robotics: A Perspective on the Latest Reviewed Concepts and Applications. *Sensors*, *21*(6), 2062. https://doi.org/10.3390/s21062062

Farrugia, C., Borg, R. P., Ferrara, L., & Buhagiar, J. (2019). The Application of Lysinibacillus sphaericus for Surface Treatment and Crack Healing in Mortar. *Frontiers in Built Environment*, *5*, 62. https://doi.org/10.3389/fbuil.2019.00062

Guamán Rivera, R., García Alvarado, R., Martínez-Rocamora, A., & Auat Cheein, F. (2021). Workspace Analysis of a Mobile Manipulator with Obstacle Avoidance in 3D Printing Tasks. *Applied Sciences*, *11*(17), 7923. https://doi.org/10.3390/app11177923

Heinrich, M. K., Von Mammen, S., Hofstadler, D. N., Wahby, M., Zahadat, P., Skrzypczak, T., Soorati, M. D., Krela, R., Kwiatkowski, W., Schmickl, T., Ayres, P., Stoy, K., & Hamann, H. (2019). Constructing living buildings: A review of relevant technologies for a novel application of biohybrid robotics. *Journal of The Royal Society Interface*, *16*(156), 20190238. https://doi.org/10.1098/rsif.2019.0238

Heveran, C. M., Williams, S. L., Qiu, J., Artier, J., Hubler, M. H., Cook, S. M., Cameron, J. C., & Srubar, W. V. (2020). Biomineralization and Successive Regeneration of Engineered Living Building Materials. *Matter*, *2*(2), 481–494. https://doi.org/10.1016/j.matt.2019.11.016

Munaro, M. R., Tavares, S. F., & Bragança, L. (2020). Towards circular and more sustainable buildings: A systematic literature review on the circular economy in the built environment. *Journal of Cleaner Production*, *260*, 121134. https://doi.org/10.1016/j.jclepro.2020.121134

Oskam, P., Bier, H., & Alavi, H. (2022). Bio-Cyber-Physical 'Planetoids' for Repopulating Residual Spaces. *SPOOL*, *9*(1), 49–55. https://doi.org/10.47982/spool.2022.1.04

Paolini, A., Kollmannsberger, S., & Rank, E. (2019). Additive manufacturing in construction: A review on processes, applications, and digital planning methods. *Additive Manufacturing*, *30*, 100894. https://doi.org/10.1016/j.addma.2019.100894

Perna, A., & Theraulaz, G. (2017). When social behaviour is moulded in clay: On growth and form of social insect nests. *Journal of Experimental Biology*, *220*(1), 83–91. https://doi.org/10.1242/jeb.143347

Petersen, K. H., Napp, N., Stuart-Smith, R., Rus, D., & Kovac, M. (2019). A review of collective robotic construction. *Science Robotics*, *4*(28), eaau8479. https://doi.org/10.1126/scirobotics.aau8479

Sustarevas, J., Butters, D., Hammid, M., Dwyer, G., Stuart-Smith, R., & Pawar, V. M. (2018). MAP - A Mobile Agile Printer Robot for on-site Construction. *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2441–2448. https://doi.org/10.1109/IROS.2018.8593815

Tang, J., Liu, G., & Pan, Q. (2021). A Review on Representative Swarm Intelligence Algorithms for Solving Optimization Problems: Applications and Trends. *IEEE/CAA Journal of Automatica Sinica*, *8*(10), 1627–1643. https://doi.org/10.1109/JAS.2021.1004129

Tiryaki, M. E., Zhang, X., & Pham, Q.-C. (2019). Printing-while-moving: A new paradigm for large-scale robotic 3D Printing. *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2286–2291. https://doi.org/10.1109/IROS40897.2019.8967524

Von Mammen, S., & Jacob, C. (2008). Swarm-driven idea models – from insect nests to modern architecture. *Eco-Architecture II*, *I*, 117–126. https://doi.org/10.2495/ARC080121

Werfel, J., Petersen, K., & Nagpal, R. (2014). Designing Collective Behavior in a Termite-Inspired Robot Construction Team. *Science*, *343*(6172), 754–758. https://doi.org/10.1126/science.1245842

Wiesenhuetter, S., Wilde, A., & Noennig, J. R. (2016). Swarm Intelligence in Architectural Design. In Y. Tan, Y. Shi, & B. Niu (Hrsg.), *Advances in Swarm Intelligence* (Bd. 9712, S. 3–13). Springer International Publishing. https://doi.org/10.1007/978-3-319-41000-5_1

Willmann, J., Augugliaro, F., Cadalbert, T., D'Andrea, R., Gramazio, F., & Kohler, M. (2012). Aerial Robotic Construction towards a New Field of Architectural Research. *International Journal of Architectural Computing*, *10*(3), 439–459. https://doi.org/10.1260/1478-0771.10.3.439

Xu, Z., Song, T., Guo, S., Peng, J., Zeng, L., & Zhu, M. (2022). Robotics technologies aided for 3D printing in construction: A review. *The International Journal of Advanced Manufacturing Technology*, *118*(11–12), 3559–3574. https://doi.org/10.1007/s00170-021-08067-2

Zhang, K., Chermprayong, P., Xiao, F., Tzoumanikas, D., Dams, B., Kay, S., Kocer, B. B., Burns, A., Orr, L., Alhinai, T., Choi, C., Darekar, D. D., Li, W., Hirschmann, S., Soana, V., Ngah, S. A., Grillot, C., Sareh, S., Choubey, A., … Kovac, M. (2022). Aerial additive manufacturing with multiple autonomous robots. *Nature*, *609*(7928), 709–717. https://doi.org/10.1038/s41586-022-04988-4

Zhang, X., Li, M., Lim, J. H., Weng, Y., Tay, Y. W. D., Pham, H., & Pham, Q.-C. (2018). Large-scale 3D printing by a team of mobile robots. *Automation in Construction*, *95*, 98–106. https://doi.org/10.1016/j.autcon.2018.08.004