The Evolution of Swarm Grammars— Growing Trees, Crafting Art and Bottom-Up Design

Sebastian von Mammen, Christian Jacob

Abstract— **We recently introduced swarm grammars (SGs) as an extension of Lindenmayer systems to model dynamic growth processes in 3D space through a large number of interacting (swarm) agents. Grammatical rewrite rules define different types of agents and their evolution over time. Sets of parameters determine specific interaction behaviors among the generated swarms.**

As we will show, swarm grammars lend themselves to creating an ecology of interacting entities and dynamic structures that are built by a multitude of agents. We will give examples of applications of swarm grammar structures in art and architecture.

In order to breed productive swarm grammars, we rely on three different approaches of genetic programming. First, a rather traditional approach of interactive genetic programming. Second, an immersive approach in which the designer takes on the role of a 'gardener', who is equipped with tools to influence and shape the on-going growth of SGs. Third, an automatically driven evolution that aims at the emergence of complex building processes.

I. Introduction

Parallel rewrite systems as a grammatical paradigm provide beneficial models to study and capture the formation of complex systems [1]. We specify agent interactions through swarm grammars [2] that are an extension of Lindenmayer systems [3], [4], which incorporate aspects of developmental design and morphogenesis. Both the rewrite rules and agent parameters are evolvable over time and help to breed structures in 3D space.

Evolution in nature is a competitive and distributed process. Agents have to compete for resources to secure their survival. Natural evolution also occurs in physical space, that is organisms as well as developmental and evolutionary processes are constrained by physical laws. Furthermore, evolution utilizes physical properties which naturally constrain the number of possible solutions. Our swarm grammar (SG) system incorporates several physical factors. However, instead of following the idea of an open-ended evolution, we drive evolutionary processes either manually to foster certain aesthetic features or automatically, if specific properties of the emerging structures are desired. In the long run, we want to merge both approaches in order to breed aesthetically pleasing and equally useful swarm grammar structures [2], [5]–[8]

In this article we describe three separately implemented breeding technologies of SGs, namely (1) interactive, (2) immersive and (3) automatic evolution. All of which have given rise to innovative constructions created for artistic as well as for architectural purposes.

During the traditional interactive processes a breeder rates a small population of SGs in accordance with his/her aesthetic impression, thereby guiding the course of evolution. The swarming agents act as self-organizing builders that compose three-dimensional structures, while they are interacting with each other, similar to termites or ants building their nests [9], [10]. The SG constructions grow in isolated virtual spaces so that no inter-species interactions take place.

The immersive breeding approach is realized in a co-evolutionary SG system that works with a multitude of agents, which we subdivide into swarms that exhibit certain properties shared by specific types of agents [11]. This creates the scenario of an emergent garden ecology, in which a gardener may arrange plants, takes care of them, and breed or re-seed plants over time. The ecology has its own dynamics, dependent on the physical properties of the simulated world and determined by the swarm agents' attributes, such as their speed, interaction dynamics, or separation and cohesion urges. Consequently, the breeder or designer is not in complete control of the overall evolutionary dynamics (which structures are built and where?), but can influence both the interaction processes as well as the evolutionary processes at any time and any location within the ecology. This is not unlike PolyWorld, a co-evolutionary virtual 2D world, in which agents, controlled by neural networks, evolve [12]. However, PolyWorld is a closed-world ecology, without any interactive external breeder.

The dynamic processes of construction as well as the final construction products of SGs can be observed and quantified, from which comparison measures can be derived. Upon the successful determination of such criteria, an automatic evolutionary process can be initiated to yield diverse, yet well-constrained SG structures.

After an overview of related work and a formal description of swarm grammars, we will discuss the three outlined breeding approaches. Finally, we will provide samples of successfully bred SG constructions for art and architecture.

II. Related Work

Our swarm grammar approach incorporates principles of morphogenesis, multi-agent systems, co-evolution, and inter-

Christian Jacob is a faculty member in the Department of Computer Science and the Department of Biochemistry & Molecular Biology, University of Calgary, Calgary, Alberta, T2N 1N4, Canada (email: cjacob@ucalgary.ca).

Sebastian von Mammen is a Ph.D. student in the Department of Computer Science, University of Calgary, Calgary, Alberta, T2N 1N4, Canada (email: s.vonmammen@ucalgary.ca).

active design. Therefore, we give a brief overview of related work in these respective areas.

A. Design through Development

Embryogenic and developmental approaches have been investigated for some time in the context of how designs can be grown instead of built [13], or how growth processes facilitate evolution [14]. The creativity that is facilitated by evolutionary systems to generate forms and functional designs [15]–[17] has led to interesting bridges between simulated design worlds and the automated manufacturing of physical and functional objects [18]–[20].

B. Design through Multi-agent Systems

Recently, promising multi-agent systems have been investigated to build and evolve virtual organisms [21] and embryogenic processes [22]. As an analogy to natural swarms, like ant colonies or schools of fish, systems of spatially interacting agents are often referred to as artificial swarms. Often, artificial swarms are associated with heuristic, parallel but neighborhood-dependend optimization methods (e.g. particle swarm optimization [23]). Such swarm models support construction and design tasks especially if their objectives are precisely phrased, as in the geometrical place problem [24], or shape optimization tasks [25]. However, none of these approaches has employed swarm intelligence to promote creative designs such as in [2].

It has also been shown that swarms of agents can be evolved to perform sorting tasks by arranging similar objects into clusters [26]–[28]. Evolutionary algorithms have been used to breed swarms of agents that display choreographed dynamics [29] or build structures in 3D space [30]. Similar reconstruction algorithms for 3D objects were also implemented with models of honey bees [31]. Computational models that combine morphogenesis and multi-agents show interesting analogies to embryogenetic processes in fruit flies [32]. Many evolutionary multi-agent systems exploit cooperation or competition in a coevolutionary environment, such as [33].

C. Interactive Evolution

Design is an iterative process. Human design, in particular, is an interactive process. Consequently, different techniques for engaging a system-external designer or evaluator into evolutionary computing have been studied. One of the early examples involves interactive evolution of procedural models for the creation of pictures and textures in computer graphics [34]. Especially interesting results were obtained with interactive techniques in the reproduction and analysis of natural evolutionary processes [35], [36]. A more formal representation of interactive evolution was proposed even before interactive evolutionary methodologies became more applicable due to the faster processing power of desktop computers [37]. Today, interactive evolutionary techniques are starting to become more sophisticated by applying machine learning techniques to adjust to user input and preferences [38].

D. Developmental Modeling and Lindenmayer-Systems

Our swarm grammars are extensions of Lindenmayer systems (L-systems) [4], which—quite successfully—have been used for the grammatical encoding of growth processes and generation of structures in two- and three-dimensional space. Plants have been modeled extensively with L-systems [39], [40], including simulated plants that interact with their environment [41]–[43]. Original work in genetic programming of L-systems [44]–[46] has led to several platforms for L-system evolution [47]–[49] and the breeding of virtual plants in a coevolutionary scenario, which even displays competitive arms-race situations [50]. Beyond plants, L-systems have also been used to evolve virtual creatures and their control networks [51], [52] and for the reconstruction of retina and blood vessel structures [53], [54].

III. Swarm Grammars

Following our previous work on swarm-based simulations [2], [55] and evolutionary swarms [29], we define a swarm grammar (SG) system as composed of two parts: (1) a set of *rewrite rules*, which determine the composition of agent types over time, and (2) a set of *agent specifications*, which define agent-type specific parameters that govern the agents' interactions.

A. Swarm Grammar Rewrite Rules

A swarm grammar system $SG = (SL, \Delta)$ consists of a rewrite system $SL = (\alpha, P)$ and a set of agent specifications $\Delta = \{\Delta_{a_1}, \Delta_{a_2}, ..., \Delta_{a_n}\}$ for n types of agents a_i. The rewrite system SL is a probabilistic L-system with axiom α and production rules P, as described in [4] and [48]. In the simplest form of context-free $0L$-systems, each rule has the form $p \xrightarrow{\theta} s$, where $p \in \Omega$ is a single symbol over an alphabet Ω, and $s \in \Omega^*$ is either the empty symbol (λ) or a word over Ω. The replacement rule is applied with probability θ. Each agent a_i is characterized by a set of attributes, Δ_{a_i}, which can include its geometrical shape, color, mass, vision range, radius of perception and other parameters such as separation or cohesion urges that determine its overall dynamics and interaction behavior as outlined in Table I.

B. Controlling the Swarm Agents' Interactions

Graphically, a swarm agent is represented as a pyramid with its tip pointing in the direction of the agent's velocity vector (Fig. 1). Each agent is only aware of other flock mates (its neighbors) within its radial field of perception which is defined by a radius (r) and an angle (β). The velocity of an agent is constantly updated with an acceleration vector V_{acc} according to a simple 'boids' model [56]:

TABLE I
PARAMETER RANGES FOR A SWARM INDIVIDUAL

Symbol	Variable	Min	Max
r	Perception field radius	50	150
β	Perception field angle	2	6.28
		—	
$w_{x,y,z}$	x-y-z world center coordinates	-1000	1000
c_1, c_2, c_3	separation, cohesion, alignment	-2	2
c_4, c_5	world center attraction, noise	0	1
V_{max}	Maximum velocity	0	25
A_{max}	Maximum acceleration	0	40
I_e	Energy loss per iteration	0	0.25
I_b	Iterations until branching	20	150
I_d	Iterations until drawing	15	30
$Col_{r,g,b}$	Color range (for each r, g and b)	0	1
Cyl_E	Number of cylinder edges	3	13
Cyl_S	Cylinder scaling	0	2

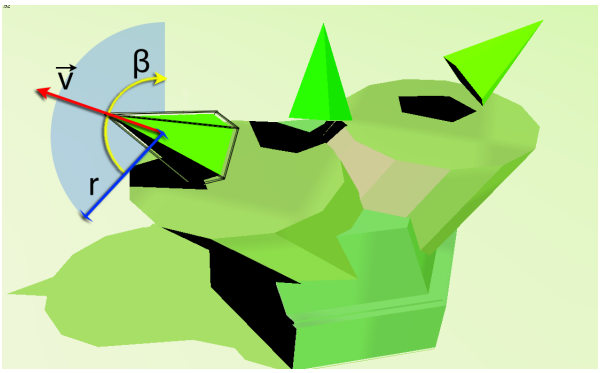


Fig. 1. The swarm agents are represented as pyramidal cones oriented towards their velocity.

$$V_{acc} = c_1 V_1(d) + c_2 V_2 + c_3 V_3 + c_4 V_4 + c_5 V_5. \quad (1)$$

Agents change their direction and adjust their speed according to three influential factors: (1) *separation* ($V_1(d)$), where an agent steers away from the collective of neighbors, given the minimum distance to other agents is smaller than a crowding radius d [29]; (2) *cohesion* (V_2), where the agent moves toward the average position of local flock mates; and (3) *alignment* (V_3), where the agent is oriented toward the average direction of its neighbors.

Vector V_4 points to the center of the simulated 3D world and V_5 represents a random unit-length vector to add some noise. The weights $c_1, ..., c_5$ determine how much influence each factor has on the agent. Each of these 'urges', c_j, is specified for an agent type as part of a swarm grammar (Table I). An agent stops applying the SL-system rules when it runs out of energy. Energy levels are inherited through replication.

The energy level also influences certain properties of the built 3D structures such as, for example, their size. The type of a swarm individual determines the visual representations of the construction elements it can leave behind on its journey through the simulated 3D scenario as illustrated in Figure 2.

Several values characterize these construction elements or building blocks: each scaled cylindrical object is placed in space at an agent's location after the swarm has flown for a certain number of iterations. The shorter these intervals I_d are, the smoother the appearance of the emerging construction. The color and the numbers of edges define the design of the cylindrical shapes (Table I).

For example, a swarm grammar $SG_a = (SL_a, \Delta_a)$ with

$$SL_a = (\alpha = A, P = \{A \rightarrow BBB, B \rightarrow A\}), \quad (2)$$
$$\Delta_a = \{\Delta_A, \Delta_B\} \quad (3)$$

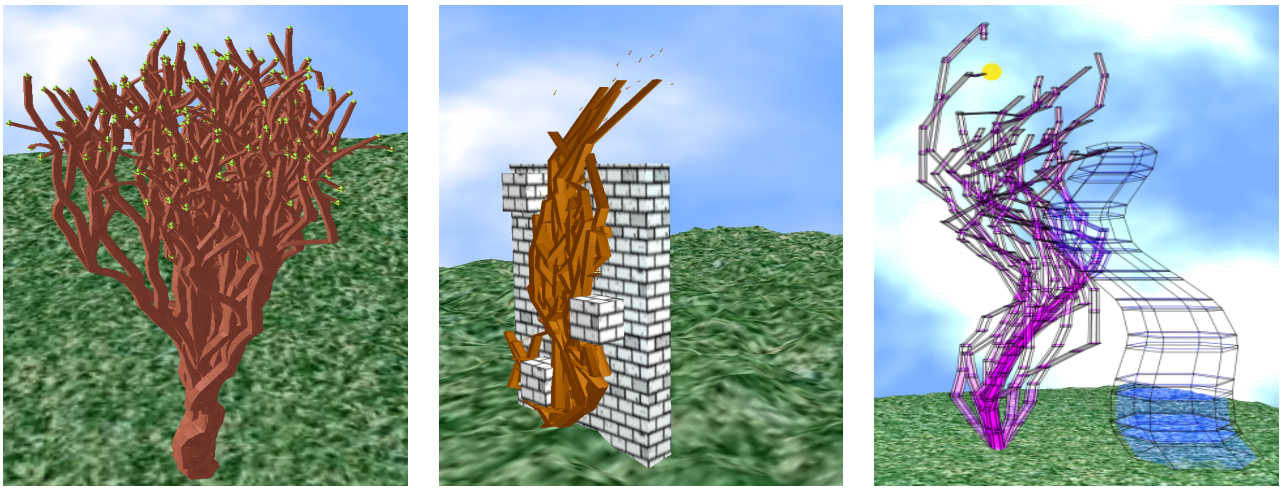
will generate a sequence of *swarm composition strings* A, BBB, AAA, $BBBBBBBBB$, etc. At each iteration step, either each type-A agent is replicated into three B agents, or agents change from type B to type A. If A agents have no separation urge ($c_1 = 0$), and B-type agents do separate ($c_1 = 1.0$), the generated swarm of agents creates a tree-like structure as in Figure 2(a). Note that here and in the following examples we assume $\theta = 1$, that is a matching rule is always applied.

In particular, Figure 2(a) displays 243 agents—which are visualized as pyramidal shapes at the branch tips. Both occurring agent types A and B have an upward urge, but since B-agents repel from each other, a bushy crown is emerging. Figure 2(b) shows a similar set of swarm grammar agents that is forced to climb up a wall. Once the agents reach to the top of the wall, they are drawn towards a fixed point above and behind the wall. The small flock of agents is visible just ahead of the top branches. In Figure 2(c) agents are attracted towards a rotating 'sun' object, which makes them follow a spiral during their upward path. The structure on the right is constructed by a single agent, whereas the left structure involves 20 agents which are repelling from each other.

Each step of applying the production rules (in parallel) represents a decision point for all agents within the system. Contrary to L-systems [4], where only a single 'turtle' is used to interpret a string, we employ a swarm of interacting agents. Neither do we need to add navigational commands for the turtles within the grammar strings, because the swarm agents navigate by themselves, determined by the agent specifications as part of the SG system. More detailed examples of swarm grammar rewriting that demonstrate further application aspects are given in [2].

IV. INTERACTIVE GENETIC SWARM GRAMMAR PROGRAMMING

Combining swarm systems with evolutionary computing has to our knowledge only been considered in the context of particle swarm optimization (e.g., [57], [58]) and in swarm-based music generating systems (e.g. [59], [60]). Emergence of collective behavior has been investigated for agents within a three-dimensional, static world [61], but this did not involve interactive evolution. Our *Genetic Swarm Grammar Programming* (GSGP) approach incorporates both interactive, user-guided evolution as well as the utilization of



$SL_a = \{A, \{A \rightarrow BBB, B \rightarrow A\}\}$ (a)

$SL_b = \{N, \{N \rightarrow NN\}\}$ (b)

$SL_c = \{I, \{I \rightarrow I\}\}$ (c)

Fig. 2. Swarm grammar agents interacting with their environment and their corresponding swarm rewrite systems.

emergent properties from interactions of a large number of agents.

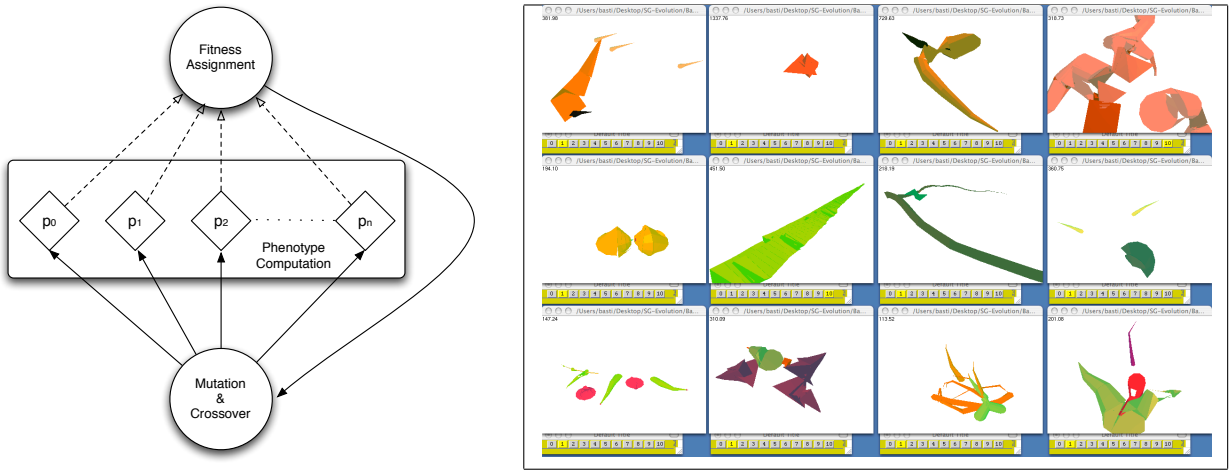
The rewrite rules and agent parameters are represented as symbolic expressions, so that GP can be used to evolve both the set of rules as well as any agent attributes. This follows our framework for evolutionary programming, *EVOLVICA* [48], where all rewrite rules and agent parameters are encoded as symbolic expressions [29]. For the examples we present here, only context-free rules with a maximum string length of three ($|s| = 3$) are applied. We allow at most five rules and up to three different types of swarm individuals per SG-genotype. Again, each agent type is described by the coefficients listed in Table I.

In our evolutionary swarm grammar experiments conducted within the *EVOLVICA* framework, standard GP tree-crossover and subtree mutations are the only applied genetic operators [48]. We use an extension of Inspirica [29], one of our interactive evolutionary design tools, to explore the potential of the described swarm grammar systems. Figure 3(a) depicts the procedure of a standard interactive evolutionary approach with manual fitness assignment. The phenotypes (p_0 to p_n) are computed separately, inspected manually, and rated. The assigned fitness values determine the probabilities of the genetic operators: selection, mutation and crossover. Arrows illustrate the flow of information. Dashed lines represent visual inspection. Accordingly, Figure 3(b) displays a screenshot of the Inspirica [29] user interface that helps to interactively evolve swarm grammars. All windows display the construction process as it occurs. All designs are true objects in 3D space, hence can be rotated, zoomed and inspected in various ways. After assessment of the presented (twelve) structures, the swarm designer assigns fitness values between 0 and 10 to each solution, and proceeds to the next generation. By means of this approach, one can easily—within only a few generations—create structures as illustrated in Figure 4.

The impact of the inter-breeding process, accomplished through crossovers of the SL-system grammars and their associated agent parameters, is illustrated in Figure 5. The replication of an agent (as determined by the grammar) and its associated constructions cease as soon as a swarm agent runs out of energy. Since the energy level of an agent is linked to the radius of the built cylindrical shape, the structures tend to look like naturally grown, with smaller tips at the ends. If the agents' energy loss, I_e, is very low, however, the radii of the cylindrical objects hardly decrease. Since the energy level is one possible termination criterion, constructions that keep their radii approximately constant often appear in tandem with vivid growth. These effects are illustrated in Figures 2 and 4.

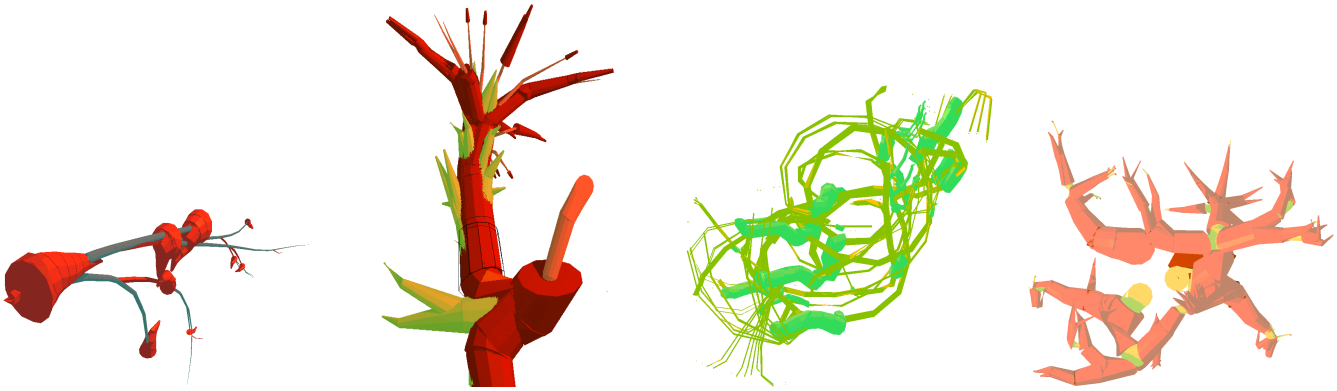
V. Immersive Swarm Grammar Evolution

In the previous examples the phenotypes are grown in separate spaces, whereas the subsequent fitness assignment is realized through a two-dimensional user interface. The isolated swarm grammar phenotypes, as depicted in Figure 3, are completely independent of each other, that is there is no interaction among the growing structures of different swarm grammars. In a co-existing and co-evolutionary setup, the encountered phenotypes can be the result of massive interactions of swarm agents. In an immersive design ecology, however, one can identify robust swarm grammars that generate stable phenotypes, whether they are isolated or put into highly populated environments. Figure 6(a) schematically depicts the processes in an immersive breeding environment. It integrates the computation of the phenotypes (p_0 to p_n, etc.) as well as the evolutionary manipulation of the underlying genotypes. In the diagram, arrows depict the flow of genetic material, induced by spatial breeding operators.



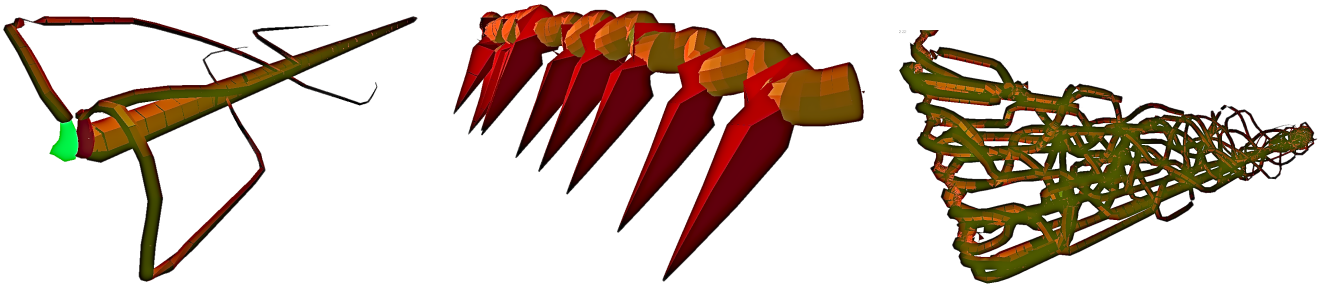
(a) (b)

Fig. 3. Interactive Swarm Grammar Evolution: (a) Diagram depicting the standard procedure of interactive evolutionary algorithms. (b) Screenshot of the Inspirica GUI that enables interactive evolution based on Mathematica in combination with its genetic programming extension Evolvica.



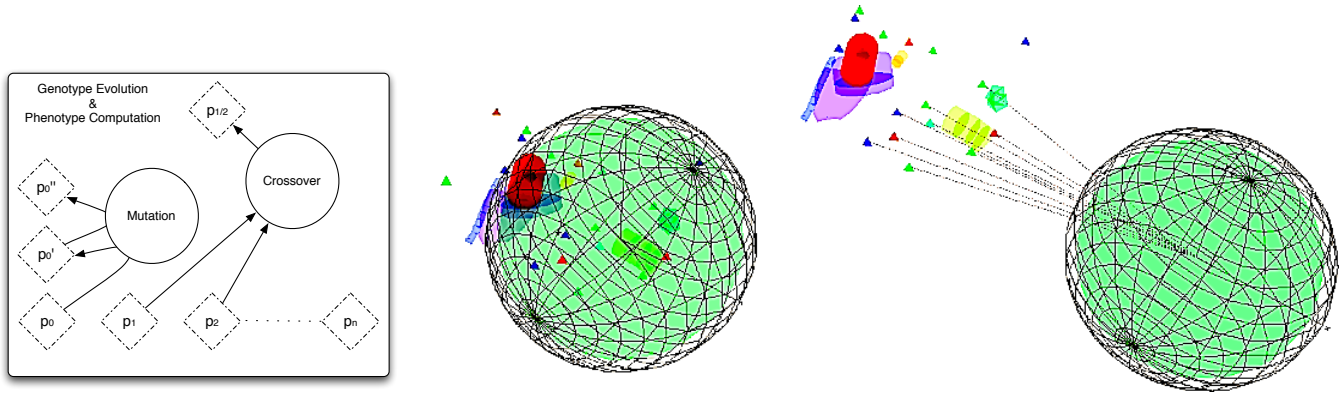
(a) (b) (c) (d)

Fig. 4. Examples of Evolved Swarm Grammar Phenotypes: (a) Pointy yet smooth nodes connect with long thin branches. (b) A flower-like structure created by a single mutation. (c) Spinning and whirling groups of swarm agents create a woven 3D pattern. (d) An organismic structure with growing tips.



(a) (b) (c)

Fig. 5. Examples of the impact of interactive breeding: (a) and (b) show two phenotypes that were interbred and whose offspring (c) successfully acquired characteristics of both parent structures. Investigation of the genotypes confirms that a recombinational transfer of a recursively applicable grammatical rule leads to the complex mesh of ramifications seen in (c).



(a) (b) (c)

Fig. 6. Immersive Evolution: (a) Diagram depicting the approach schematically. (b) By means of volumetric tools and (c) additional visual aids, the immersed breeder can manually select and tinker with the present specimen.

A. Spatial Breeding Operators

Our immersive user interface integrates two aspects: visual representation and intuitive manipulation by an external breeder or designer. The latter mechanism is realized by the already mentioned spatial breeding operators, or breeder volumes. Figure 6(b) shows a breeder volume that encloses several swarm grammar agents. Swarm agents that pass through a volume (a sphere in this case) can be influenced in various ways. We use breeder volumes for the crossover and mutation operators, for moving and copying swarm agents, and for boosting their energy levels. Analogous to the watering of plants in a garden, fitness evaluations are only given implicitly by providing more energy to selected groups of agents. In order to facilitate the selective evolutionary intervention, breeder volumes can be placed at fixed positions to perform operations on temporary visitors with predefined frequencies. Additional visuals allow to keep track of previous agent selections. Figure 6(c) depicts how previously enclosed agents remain associated with the according breeder volume. This relationship is visualized by the connecting lines.

The visualization interface enables the moving, rotating, and zooming of the camera, or the saving and restoring of specific views and scenario settings (Fig. 7). Most of these procedures are already incorporated in the agent software environment *BREVE* which we use as our display and simulation engine [61]. In addition to aspects of visualization, the supervising breeder is equipped with tools to select, group, copy, and move swarm grammar agents, thus being able to influence the course of evolution within the emerging scenario. The set of possible manipulations also includes mutation and crossover operators to manually trigger changes of the genotypes that encode the swarm grammar rules and the agent parameters.

B. The Swarm Grammar Gardener

Figure 7 illustrates how a breeder can influence the emerging building processes within a simple ecology of swarms. In Figure 7(a) two swarm agents have built a cylindrical structure with a small side branch. Both agents, which have run out of energy, are still visible at the top left and to the right of this construction. In the next step (Fig. 7(b)) a breeder sphere is introduced so that it encloses the agent on the right. Through a contextual menu, this agent is 'revived' by replenishing its energy reservoir. Subsequently, the agent resumes its building process, generates an additional side branch and extends the overall structure further to the right (Fig. 7(c)). A similar procedure is applied to the agent on the left. It is captured by the breeder sphere and triggered to first replicate, i.e., make copies of itself, and then resume construction (Fig. 7(d,e)). This generates further expansions of the structures and—after further energy boosts (Fig. 7(f))—results in the structure depicted in Figure 7(g). The pattern continues to grow until the agents run out of energy again.

This is only an example of how external manipulation by a breeder, the 'gardener', can influence the agent behaviors, the building or developmental processes. Their evolution as agents can change their respective control parameters during replication. Agents of a specific type share a swarm grammar, but agent groups can be copied as well, so that they inherit a new copy of their own swarm grammar, which may also evolve over time, either automatically or through direct influence from the gardener. Figure 8 gives a few examples of evolved swarm grammar ecologies and extracted structures at different stages during their evolution.

VI. Swarm Constructions in the Arts

In the provided examples, the aesthetic judgement of a breeder drove the artificial evolution of swarm grammars. This objective suits well for endeavors in which an artist searches for innovative expressions of certain artistic themes. Swarm grammar constructions are special in that the dynamics of their construction processes are captured in the emerging structures in peculiar ways. Local interactions determine the placement of construction elements and the flight formations of the swarm. Inherent in any swarm system, the tandem of actions and reactions can result in

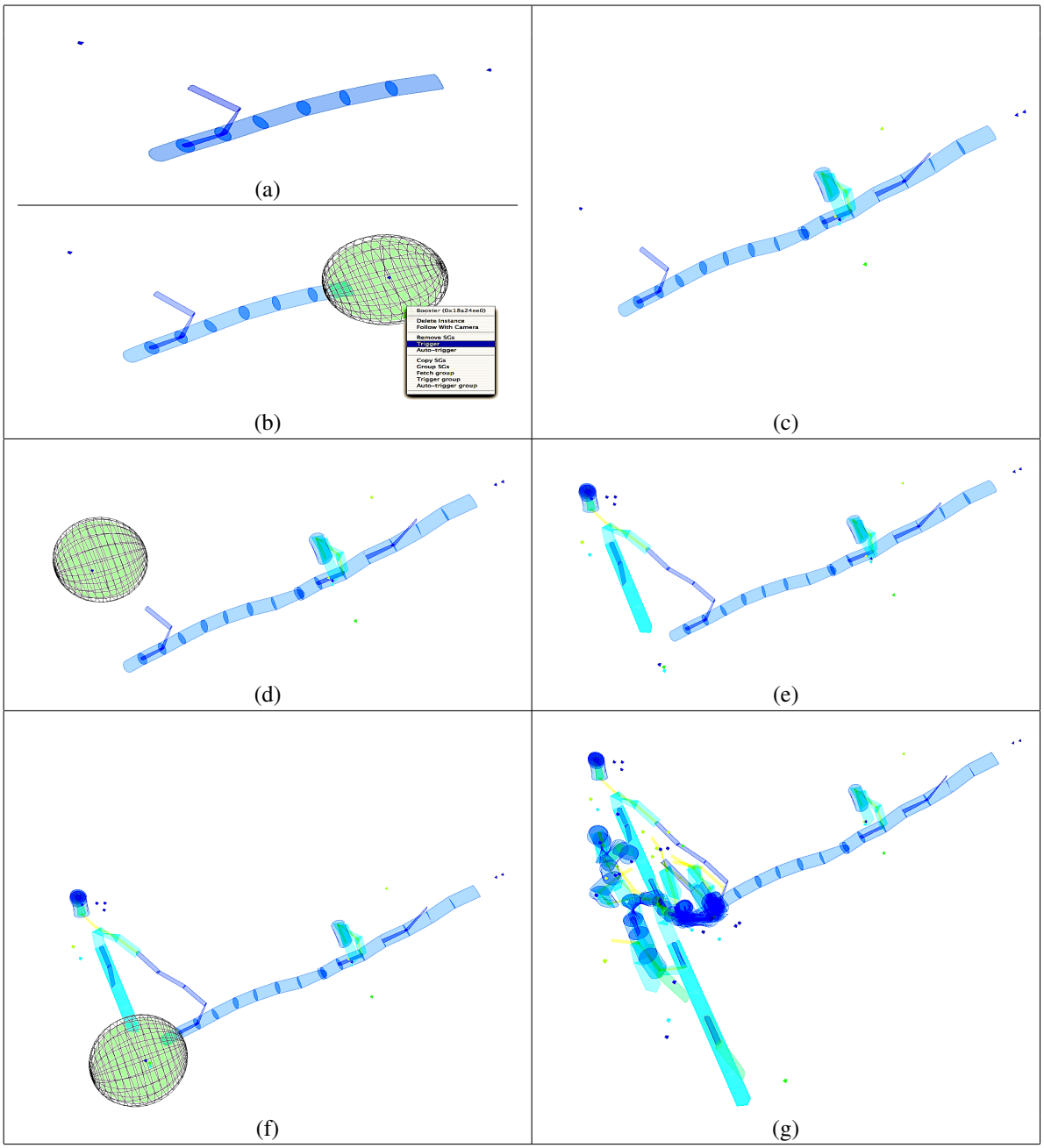


Fig. 7. Illustration of Interactive Manipulation of Swarm Grammar Agents by an External Breeder. (a) Two agents create an initial structure. (b) A breeder sphere locally infuses energy. (c) Further growth is initiated by the additional energy. (d-e) Replication of an agent triggers further parallel construction. (f-g) Expansion of the structure is continued after another energy influx.

a feedback loop of interdependencies [62]. The diagram in Figure 9 hints at the complex relationships that arise in boid systems, without even considering indirect communication beyond the ever changing neighborhood relations between the swarm individuals: A swarm agent i perceives a set of neighbors that determine its acceleration. Its changed location, in turn, affects those swarm mates that perceive i as a neighbor. The emerging dynamics are captured in structures that exhibit liveliness and spontaneity, contrasting themes, rhythmic movements, tension, organic looks, and rigid forms.

Accordingly, the artistic interpretation of SG structures can support artistic work in several ways, for example by composing pieces of computer-generated SG structures and traditionally painted motives, or by inspiring themes and concepts of artistic works as a whole [6]. As an example, Figure 10(a) displays the diptych [Outlining Blues] by the Canadian artist Joyce Wong. It comprises two oil paintings on 12' x 24' metal plates that were preprocessed with rusting agents. It was inspired by the interplay of two different SG systems (Fig. 10(b))—one swarm grammar leaving a black thin trace of cylindrical objects, the other one building up

Fig. 8. Collage of Designs Generated by Swarm Grammars. The figure in the centre illustrates a swarm grammar garden ecology, within which the surrounding designs were created.

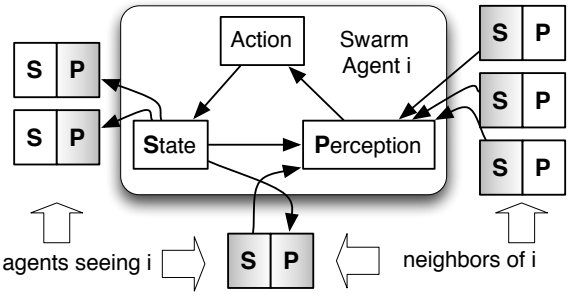
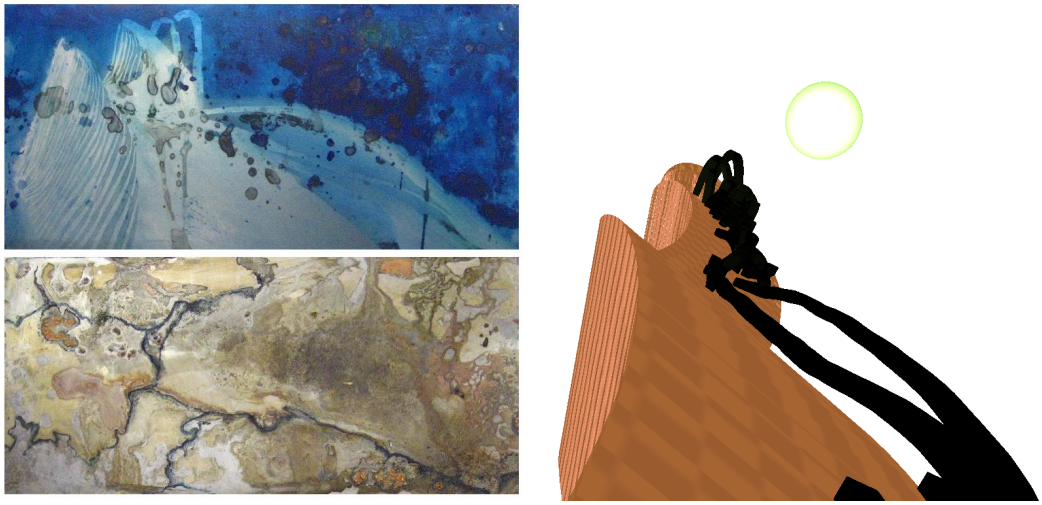


Fig. 9. The black arrows in the upper box show the direction of influence between perception, action and state of a swarm agent i. The S-P tuples stand for the state and perception modules of other agents that interact with agent i.

waves of pyramidal layers.

VII. Automatic Swarm Grammar Evolution

Once a concrete design task is chosen for a swarm grammar system, certain constraints on the growing structure can be formulated. Afterwards, the power of innovation of evolutionary computation can be harnessed to automatically create assortments of SG designs. In addition to the analysis of the genotype of a swarm grammar, two things can be subjected to the fitness assignment of an evolutionary algorithm: (1) the construction processes and (2) the emerging structures. Structural analysis is either very course grained, considering for example the overall volume and the proportions, or computationally very costly, for instance when attempting to identify hierarchies and re-occurring modules. Therefore,



(a) (b)

Fig. 10. Swarm grammar structures are inspirations for art. (a) An artistic realization of swarm grammar renderings. (b) The underlying SG structures.

we put an emphasis on the observation and classification of the construction processes.

In particular, in our SG examples used for architectural designs, we promote productivity, diversity and collaboration and prevent computational outgrowth of the generated structure using the fitness formula:

$$f_{SG}^{simple} = r_p + g_n + \frac{g_c + g_a}{\sqrt{max(t,1)}} \quad (4)$$

$$g_n = sin(\pi * r_n) \quad (5)$$

$$g_c = sin(\pi * 0.005 * min(n_c, 200)) \quad (6)$$

$$g_a = sin(\pi * 0.005 * min(n_a, 200)) \quad (7)$$

In order to measure productivity the SG constructions are compared with pre-defined structures, yielding the ratio r_p. Diversity is traced in the total number of expressed agent genotypes n_a, as well as in the according value n_c of deployed construction materials or construction mechanisms. In order to foster collaboration between the SG agents, we protocol the average ratio of perceived neighbors r_n. Too low values of r_n imply that no interactions are taking place, whereas too great values mean that the agents are trapped within small spaces. As the basic swarm grammar model can be extended in accordance with the respective application task, SG agent collaboration may also be reflected in the amount of effective communication between the agents. Randomly initialized swarm grammar systems can quickly exhaust the provided computing power: Fast, possibly unconditional sequences of SG rule applications may result in exponential agent reproduction rates. Temporarily such explosions of activity could be beneficial, for example in designs integrating large numbers of ramifications. In the long run, however, such an overwhelming demand on computing requirements has to be avoided. As a simple means to prevent destructive outgrowth, yet allowing for temporary leaps of activity, we set a time limit of 100 real seconds for the construction process that are limited to 8 simulated seconds. Thus, we filter inefficient SGs during the evolutionary experiments.

A desired value for the neighborhood ratio r_n is somewhere close to 0.5, hence the normalization by a sine function (Eqn. 5). Accordingly, the numbers of deployed construction elements and expressed agents are rewarded only within a certain interval (Eqn. 6 and 7, respectively). Additionally, the SG fitness function yields a greater value with a shorter simulation time t.

The automatic evolution allowed us to start a systematic comparison between different implementations of genetic operators. In the described setup, for instance, a crossover on single agents showed better results than on the level of single genes, such as single flocking parameters or behavioral rules of construction and reproduction. This result is rendered plausible by the fact that a SG agent is a higher-order module that encapsulates a broad variety of interwoven characteristics and behaviors. However, further investigations are required to diligently develop phases of modular evolution and to improve on the effectiveness of the deployed genetic operators in general.

VIII. Ecological Swarm Architecture

A detailed evolutionary approach to automatic SG evolution is presented in [7]. Here, the implementations of the constraints mentioned above led to a variety of aesthetically innovative architectural design models. In order to reward productivity of a swarm grammar, we measured the emerged architectural structure against a solid cube. Construction elements built inside the pre-defined cubic shape contributed positively to an SG's fitness, whereas constructions outside the cube decreased it, similar to an approach we used in [30]. For the task of architectural design generation an extension of the basic swarm grammar concept has been implemented. Instead of the continuous placement of construction elements

and instead of a regularly timed application of the SG's reproduction rules, these activities became conditional on either chance, agent-internal timers, or the perception of specific 'smells' or visual stimuli. An example of the rule-based genotype extension of swarm grammars is illustrated in Figure 11. Perception-based rule execution allows for indirect, so-called *stigmergic* communication by which social insects, such as ants, termites and some wasp and bee species coordinate their construction efforts [9], [11], [63]. In fact, we count the events of successful stigmergic communication in SGs to foster coordination of the agents' activities. In particular, we consider it a successful communication event whenever a construction or reproduction rule of an agent is triggered by a stigmergic stimulus.

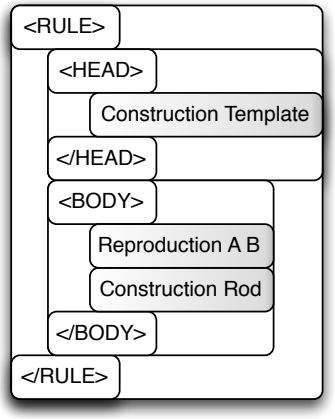


Fig. 11. An example of a behavioral rule of a swarm grammar agent that considers internal timers, chance and stigmergic stimuli.

For architectural swarm grammar models we capture the swarm dynamics as in the SG art experiments. The flowing and organic shapes built by the bio-inspired, generative SG representation promise to support the design efforts of architects [8], [64]. Several examples of architectural SG models are displayed in Figure 12. Utilizing the extended swarm grammar model to breed architectural designs is not only interesting from a creative and innovative perspective on aesthetics, but also bears the potential of 'optimizing' architectural designs. In addition to an evolutionary promotion of productivity, coordination and diversity, ecological and economical aspects of the emerging architectural models could drive the evolutionary processes. Such ecological criteria could be temperature regulation and ventilation [65], [66], adaptation of building structures to the surrounding landscape, utilization of sun exposed structures for electric power generation, and other evolvable and measurable features.

IX. Conclusion and Future Work

We presented swarm grammars as an extension of Lindenmayer systems. Instead of applying a single ('turtle') agent to convert linear strings into 3D structures, we use a swarm of agents which navigate in 3D space and—as a side effect—place structural building blocks into their environment. The swarm grammars are used to specify how the setup of agent types changes over time. Additional agent parameters determine the agents' behaviors and their interaction dynamics. Both the grammar rules and the agent parameters are evolvable and can change over time—either automatically at replication and collision events among the agents, or triggered by external 'tinkering' from a supervising breeder. When swarm grammars are applied to concrete problems, constraints on the developmental processes as well as on the emerging structures may provide the basis for an automatic evolutionary algorithm.

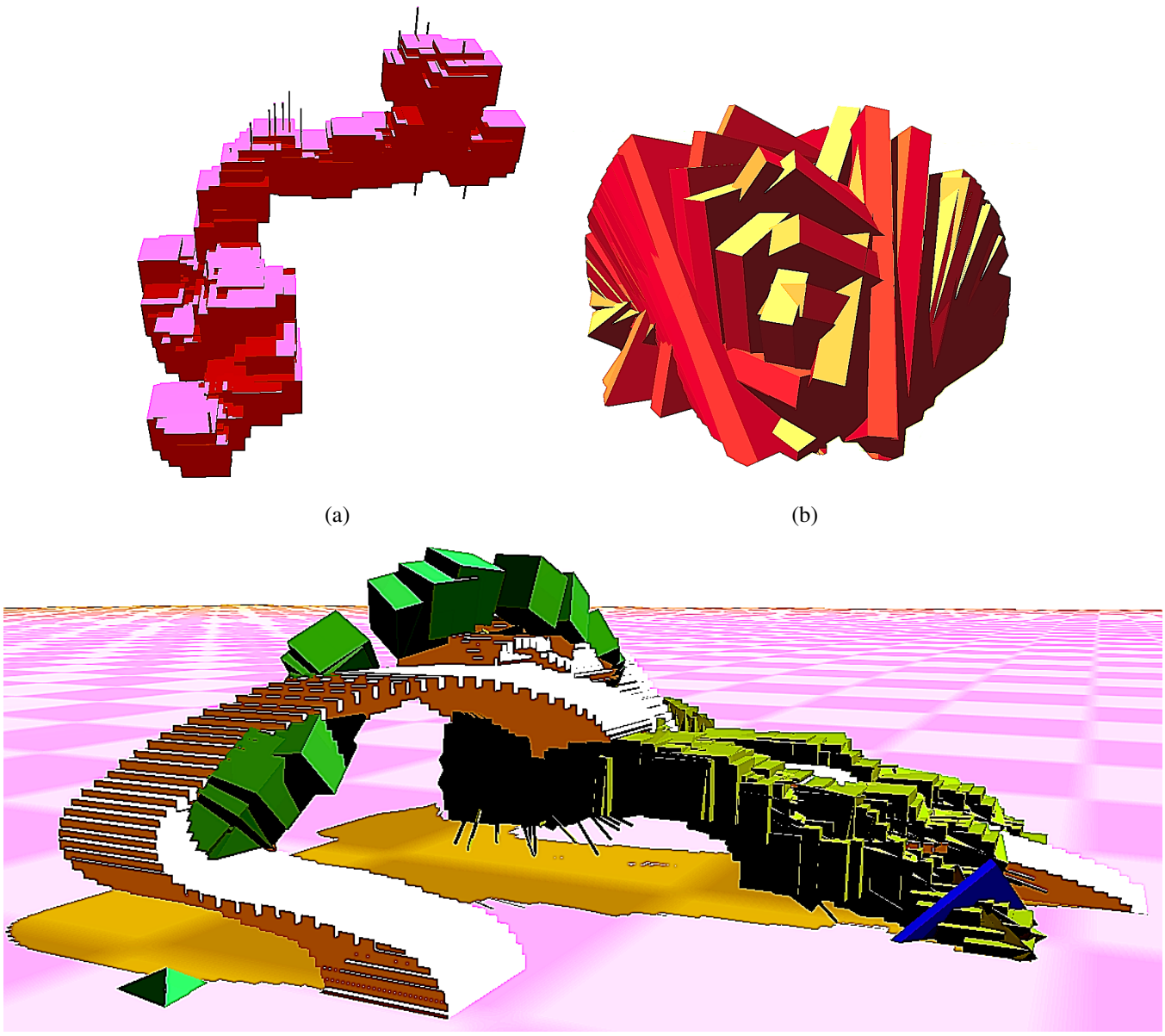
Several examples show how we utilize the interaction dynamics of swarm agents to generate structural forms in virtual 3D space. Since the characteristics of the developmental construction processes are reflected in the built structures, swarm grammar systems are interesting for artistic works. This property can be exploited for other design applications as well, as illustrated by several of our architectural idea models. For the generation of architectural models we extended the basic swarm grammar system with the means of event-driven construction and reproduction behavior, which in turn allows for indirect (stigmergic) communication between the swarm individuals. The combination of swarm grammars as bio-inspired generative representation and evolutionary exploration of innovative designs opens up an array of possibilities to develop (post-)modern ecological architectural designs.

Swarm grammar agents can also be used to investigate and evolve other dynamic processes, such as in gene regulatory processes [55], [67], immune system interactions [68], swarm behavior choreographies [29], or in interactive SwarmArt installations [69].

Currently we are investigating several possible improvements of swarm grammar systems and their applications in arts and architecture. A discretized lattice system of swarm grammars promises enhanced measurements of the interaction and construction processes, thereby increasing the control of the evolutionary runs. First analyses of swarm systems as complex interaction networks have shown to be supportive in coordinating different phases of construction: Phase transitions in the exhibited flight patterns, for instance, directly influence possible emerging structures [62].

Furthermore, we are continuing an interdisciplinary collaborative project between computer scientists and artists. Here, a strong emphasis is placed on the integration of new and traditional media. The developmental processes are attempted to be caught from different angles: SG structures egress the 2D screen mapping when plotting them in 3D on prototype printers or through artistic sculpturing. Additionally, we are exploring means to capture the swarm dynamics and construction processes through their projection and to integrate them with artistic work on conventional canvas.

Future work includes honing the architectural modeling approach in ecological and economic regards. A hybrid system of interactive, automatic and possibly immersive breeding technologies would strengthen these efforts. The creation



(a) (b)

(c)

Fig. 12. Three Architectural Idea Models Created in an Automated Process of Swarm Grammar Evolution. As construction elements the SG agents could rely on cubic bodies, layers and rods. (a) represents a body-based idea model that occasionally integrates rod elements. (b) arose through the dynamic placement of layers. (c) shows a rather complex construction utilizing all basic construction elements.

of a swarm simulation environment that is distributed onto computer clusters would further enhance the means to evolutionarily explore the interaction and construction patterns of artificial swarm systems.

Up-to-date details about swarm grammars and other agent-based simulation examples from our *Evolutionary & Swarm Design Lab* can be found at: `http://www.swarm-design.org` .

References

[1] J. Collado-Vides, "Towards a grammatical paradigm for the study of the regulation of gene expression," in *Theoretical Biology. Epigenetic and Evolutionary Order from Complex Systems*, B. Goodwin and P. Saunders, Eds. Baltimore, ML: Johns Hopkins University Press, 1992, pp. 211–224.

[2] C. Jacob and S. von Mammen, "Swarm grammars: growing dynamic structures in 3d agent spaces," *Digital Creativity*, vol. 18, no. 1, 2007.

[3] P. Prusinkiewicz and J. Hanan, *Lindenmayer Systems, Fractals, and Plants*, ser. Lecture Notes in Biomathematics. New York: Springer, 1989, vol. 79.

[4] P. Prusinkiewicz and A. Lindenmayer, *The Algorithmic Beauty of Plants*. New York: Springer, 1990.

[5] S. von Mammen and C. Jacob, "Genetic swarm grammar programming: Ecological breeding like a gardener," in *2007 IEEE Congress on Evolutionary Computation*, ser. IEEE Press, D. Srinivasan and L. Wang, Eds., 2007, pp. 851–858.

[6] S. von Mammen, J. Wong, and C. Jacob, "Virtual constructive swarms: Compositions and inspirations," in *Applications of Evolutionary Computing, Proceedings of EvoWorkshops 2008*, ser. Lecture Notes in

Computer Science, vol. 4974. Berlin-Heidelberg: Springer-Verlag, 2008, pp. 491–496.

[7] S. von Mammen and C. Jacob, "Evolutionary swarm design of architectural idea models," in *Genetic and Evolutionary Computation Conference (GECCO) 2008*. New York, NY, USA: ACM Press, 2008, pp. 143–150.

[8] ——, "Swarm-driven idea models - from insect nests to modern architecture," in *Eco-Architecture 2008, Second International Conference on Harmonisation Between Architecture and Nature*, C. Brebbia, Ed. Winchester, UK: WIT Press, 2008, pp. 117–126.

[9] S. Camazine, J.-L. Deneubourg, N. R. Franks, J. Sneyd, G. Theraulaz, and E. Bonabeau, *Self-Organization in Biological Systems*, ser. Princeton Studies in Complexity. Princeton: Princeton University Press, 2003.

[10] S. Johnson, *Emergence: The Connected Lives of Ants, Brains, Cities, and Software*. New York: Scribner, 2001.

[11] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*, ser. Santa Fe Institute Studies in the Sciences of Complexity. New York: Oxford University Press, 1999.

[12] L. Yaeger, "Computational genetics, physiology, metabolism, neural systems, learning, vision, and behavior or polyworld: Life in a new context," in *Artificial Life III*, C. G. Langton, Ed. Addison-Wesley, 1994.

[13] P. J. Bentley and S. Kumar, "Three ways to grow designs: A comparison of embryogenies for an evolutionary design problem," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-1999)*, 1999.

[14] S. Kumar and P. J. Bentley, "Implicit evolvability: An investigation into the evolvability of an embryogeny," in *GECCO 2000: Genetic and Evolutionary Computation Conference — Late Breaking Papers*, 2000.

[15] T. Broughton, A. Tan, and P. S. Coates, "The use of genetic programing in exploring 3D design worlds," in *CAAD Futures 97*, R. Junge, Ed. Technical University Munich, Germany: Kluwer Academic Publishers, 4-6 Aug. 1997, pp. 885–917. [Online]. Available: http://www.uel.ac.uk/ceca/cad/lsystems/lsystemsgp.htm

[16] P. Bentley and D. Corne, Eds., *Creative Evolutionary Systems*, ser. Artificial Intelligence. San Francisco, CA: Morgan Kaufmann, 2001.

[17] S. Kumar and P. Bentley, Eds., *On Growth, Form and Computers*. London: Elsevier Academic Press, 2003.

[18] H. Lipson and J. B. Pollack, "Automatic design and manufacture of robotic lifeforms," *Nature*, vol. 406, pp. 974–978, August 31 2000.

[19] J. Rieffel and J. Pollack, "Automated assembly as situated development: using artificial ontogenies to evolve buildable 3-d objects," in *GECCO '05: Genetic and evolutionary computation conference*. New York, NY, USA: ACM Press, 2005, pp. 99–106.

[20] M. Hemberg, U.-M. O'Reilly, A. Menges, K. Jonas, M. da Costa Gonçalves, and S. R. Fuchs, *The Art of Artificial Life: A Handbook on Evolutionary Art and Music*, ser. Natural Computing Series. Springer, 2008, ch. Genr8: Architects' Experience with an Emergent Design Tool, pp. 167–188.

[21] G. Beurier, F. Michel, and J. Ferber, "Towards an evolution model of multiagent organisms," in *ECCS'05: European Conference on Complex Systems, Workshop on Multi-Agents for Modeling Complex Systems (MA4CS)*, 2005.

[22] S. Kumar and P. J. Bentley, "Biologically inspired evolutionary development," *Evolvable Systems: From Biology to Hardware*, pp. 99–106, 2003. [Online]. Available: http://www.springerlink.com/content/v5gyw4hvv0gd1ref

[23] J. Kennedy and R. C. Eberhart, *Swarm Intelligence*, ser. The Morgan Kaufmann Series in Evolutionary Computation. San Francisco: Morgan Kaufmann Publishers, 2001.

[24] C. Grosan, A. Abraham, S. Han, and A. Gelbukh, "Hybrid particle swarm – evolutionary algorithm for search and optimization," in *MICAI 2005: Advances in Artificial Intelligence. Lecture Notes in Artificial Intelligence*. Springer-Verlag, 2005, pp. 623–632.

[25] A. Adly and S. Abd-El-Hafiz, "Using the particle swarm evolutionary approach in shape optimization and field analysis of devices involving nonlinear magnetic media," *IEEE Transactions on Magnetics*, vol. 42, no. 10, pp. 3150–3152, October 2006.

[26] M. Resnick, *Turtles, Termites, and Traffic Jams: Explorations in Massively Parallel Microworlds*, ser. Complex Adaptive Systems. Cambridge, MA: MIT Press, 1997.

[27] V. S. Colella, E. Klopfer, and M. Resnick, *Adventures in Modeling: Exploring Complex, Dynamic Systems with StarLogo*. New York: Teachers College Press, Columbia University, 2001.

[28] V. Hartmann, "Evolving agent swarms for clustering and sorting," in *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*. New York, NY, USA: ACM Press, 2005, pp. 217–224.

[29] H. Kwong and C. Jacob, "Evolutionary exploration of dynamic swarm behaviour," in *Congress on Evolutionary Computation*. Canberra, Australia: IEEE Press, 2003.

[30] S. von Mammen, C. Jacob, and G. Kókai, "Evolving swarms that build 3d structures," in *CEC 2005, IEEE Congress on Evolutionary Computation*. Edinburgh, UK: IEEE Press, 2005.

[31] G. Olague and C. Puente, "Parisian evolution with honeybees for three-dimensional reconstruction," in *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*. New York, NY, USA: ACM Press, 2006, pp. 191–198.

[32] G. Beurier, F. Michel, and J. Ferber, "A morphogenesis model for multiagent embryogeny," in *Proceedings of the 10th International Conference on the Simulation and Synthesis of Living Systems (ALIFE X)*, 2006.

[33] A. Bucci and J. B. Pollack, "On identifying global optima in cooperative coevolution," in *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*. New York, NY, USA: ACM Press, 2005, pp. 539–544.

[34] K. Sims, "Interactive evolution of equations for procedural models," *The Visual Computer*, vol. 9, no. 8, pp. 466–476, 1993.

[35] J. Graf and W. Banzhaf, "Interactive evolution in simulated natural evolution," in *Artificial Evolution*, vol. LNCS 1063, 1995, pp. 259–272.

[36] ——, "An expansion operator for interactive evolution," in *Proceedings of the IEEE International Conference on Evolutionary Computation*. IEEE Press, 1995, pp. 798–802.

[37] W. Banzhaf, "Interactive evolution," in *Handbook of Evolutionary Computation*, T. Bäck, D. B. Fogel, and Z. Michalewicz, Eds. Bristol, New York: Institute of Physics Publishing and Oxford University Press, 1997, pp. C2.9:1–6.

[38] X. Llorà, K. Sastry, F. Alías, D. E. Goldberg, and M. Welge, "Analyzing active interactive genetic algorithms using visual analytics," in *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*. New York, NY, USA: ACM Press, 2006, pp. 1417–1418.

[39] P. Prusinkiewicz, M. Hammel, J. Hanan, and R. Mech, "Visual models of plant development," in *Handbook of Formal Languages*, G. Rozenberg and A. Salomaa, Eds. New York: Springer, 1997.

[40] J. Yu, "Evolutionary design of 2d fractals and 3d plant structures for computer graphics," Master's Thesis, Department of Computer Science, University of Calgary, 2004.

[41] R. Mech and P. Prusinkiewicz, "Visual models of plants interacting with their environment," in *SIGGRAPH'96*. New Orleans, Louisiana: ACM SIGGRAPH, New York, 1996, pp. 397–410.

[42] M. T. Michalewicz, Ed., *Plants to Ecosystems: Advances in Computational Life Sciences*. Collingwood, VIC, Australia: CSIRO Publishing, 1997.

[43] O. Deussen, P. Hanrahan, B. Lintermann, R. Mech, M. Pharr, and P. Prusinkiewicz, "Realistic modeling and rendering of plant ecosystems," in *SIGGRAPH 98, Computer Graphics, Annual Conference Series*. ACM SIGGRAPH, 1998, pp. 275–286.

[44] C. Jacob, "Genetic l-system programming," in *PPSN III - Parallel Problem Solving from Nature*, ser. Lecture Notes in Computer Science, vol. 866. Jerusalem, Israel: Springer, 1994, pp. 334–343.

[45] ——, "Evolving evolution programs: Genetic programming and l-systems," in *Genetic Programming 1996: First Annual Conference*, J. R. Koza, D. E. Goldberg, D. B. Fogel, and R. Riolo, Eds. Stanford University, Palo Alto, CA: MIT Press, Cambridge, MA, 1996, pp. 107–115.

[46] ——, "Evolution and co-evolution of developmental programs," *Computer Physics Communications, Special Issue, Modeling Collective Phenomena in the Sciences*, 1999.

[47] K. J. Mock, "Wildwood: The evolution of l-system plants for virtual environments," in *IEEE Conference on Evolutionary Computation*. Anchorage, AL: IEEE Press, New York, 1998, pp. 476–480.

[48] C. Jacob, *Illustrating Evolutionary Computation with Mathematica*. San Francisco, CA: Morgan Kaufmann Publishers, 2001.

[49] F. Michel, G. Beurier, and J. Ferber, "The turtlekit simulation platform: Application to complex systems," in *Proceedings of Workshop Sessions*

at the 1st International Conference on Signal & Image Technology and Internet-Based Systems (IEEE SITIS05). IEEE Press, 2005, pp. 122–128.

[50] M. Ebner, "Coevolution and the red queen effect shape virtual plants," *Genetic Programming and Evolvable Machines*, vol. 7, no. 1, pp. 103–123, 2006.

[51] G. S. Hornby and J. B. Pollack, "Evolving l-systems to generate virtual creatures," *Computers & Graphics*, vol. 25, pp. 1041–1048, 2001.

[52] ——, "Body-brain co-evolution using L-systems as a generative encoding," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, L. Spector, E. D. Goodman, A. Wu, W. B. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. H. Garzon, and E. Burke, Eds. San Francisco, California, USA: Morgan Kaufmann, 7-11 2001, pp. 868–875.

[53] G. Kókai, R. Ványi, and Z. Tóth, "Parametric l-system description of the retina with combined evolutionary operators," in *Genetic and Evolutionary Computation Conference, GECCO-99*, Orlando, Florida, USA, 1999.

[54] G. Kókai, Z. Tóth, and R. Ványi, "Modelling blood vessel of the eye with parametric l-systems using evolutionary algorithms," in *Artificial Intelligence in Medicine, Proceedings of the Joint European Conference on Artificial Intelligence in Medicine and Medical Decision Making, AIMDM'99*, W. Horn, Y. Shahar, G. Lindberg, S. Andreassen, and J. C. Wyatt, Eds., vol. 1620, 1999, pp. 433–443.

[55] C. Jacob and I. Burleigh, "Biomolecular swarms: An agent-based model of the lactose operon," *Natural Computing*, vol. 3, no. 4, pp. 361–376, December 2004.

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

[57] M. Settles, P. Nathan, and T. Soule, "Breeding swarms: a new approach to recurrent neural network training," in *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*. New York, NY, USA: ACM Press, 2005, pp. 185–192.

[58] M. Settles and T. Soule, "Breeding swarms: a ga/pso hybrid," in *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*. New York, NY, USA: ACM Press, 2005, pp. 161–168.

[59] T. Blackwell, "Swarming and music," *Evolutionary Computer Music*, pp. 194–217, 2007.

[60] D. Jones, "Atomswarm: A framework for swarm improvisation," *Applications of Evolutionary Computing*, pp. 423–432, 2008.

[61] L. Spector, J. Klein, C. Perry, and M. Feinstein, "Emergence of collective behavior in evolving populations of flying agents," in *Genetic and Evolutionary Computation Conference (GECCO-2003)*, E. C.-P. e. al., Ed. Chicago, IL: Springer-Verlag, 2003, pp. 61–73.

[62] S. von Mammen and C. Jacob, "The spatiality of swarms — quantitative analysis of dynamic interaction networks," in *Proceedings of Artificial Life XI*. MIT Press, 2008, pp. 662–669.

[63] B. Hölldobler and E. O. Wilson, *The Ants*. Berlin-Heidelberg: Springer-Verlag, 1990.

[64] S. Van der Ryn and S. Cowan, *Ecological Design*. Island Press, 2007.

[65] K. Gowri, "Green building rating systems: An overview." *ASHRAE Journal*, vol. 46, no. 11, pp. 56–60, 2004.

[66] G. Farmer and S. Guy, "Visions of Ventilation: Pathways to Sustainable Architecture," *Department of Architecture, University of Newcastle upon Tyne, Newcastle upon Tyne,(UK)*, 2002.

[67] C. Jacob, A. Barbasiewicz, and G. Tsui, "Swarms and genes: Exploring λ-switch gene regulation through swarm intelligence," in *CEC 2006, IEEE Congress on Evolutionary Computation*, 2006.

[68] C. Jacob, S. Steil, and K. Bergmann, "The swarming body: Simulating the decentralized defenses of immunity," in *Artificial Immune Systems, ICARIS 2006, 5th International Conference*. Oeiras, Portugal: Springer, September 2006.

[69] C. Jacob, G. Hushlak, J. Boyd, P. Nuytten, M. Sayles, and M. Pilat, "Swarmart: Interactive art from swarm intelligence," *Leonardo*, vol. 40, no. 3, 2007.