Form finding and generative systems: A theoretical and applied research project

Gonçalo Castro HENRIQUES *1
Ernesto BUENO *2
Jarryer De MARTINO *3
Victor SARDENBERG *4
Daniel LENZ *5
(*1 e 5 Federal University of Rio de Janeiro, Brazil, LAMO - PROURB)
(*2 Positivo University, Curitiba, Brazil)
(*3) Federal University of Espírito Santo, Brazil)
(*4 Leibniz Universitaet Hannover, Germany)

LAMO seminar/workshop >> *Em busca da forma, sistemas generativos* (Form finding & generative systems) sought to deepen the theoretical and practical development of generative algorithms in design. It proposed to explore the advances of mathematics at the end of the 20th century, namely information theory and general systems theory. These advances provided a new set of techniques to find design solutions. Although these techniques have already been applied in areas such as engineering, graphic design and urbanism—given their mathematical and computational nature —their application in architecture is still scarce. This event sought to translate computational techniques, such as cellular automata, L-systems, genetic algorithms and shape grammars, for generative form finding tools. The idea was to understand through experimentation how these techniques can be applied to solve design problems. The event was preceded by a research and was supported by an international scientific committee, gathering a group of researchers experienced in the referred techniques. This article describes the methodology, studying the results, evaluating the potentialities and limitations of each technique, considering future developments.

Keywords: Generative Systems, Cellular Automata, L-Systems, Genetic Algorithms, Shape Grammar

1. INTRODUCTION, GENERATIVE SYSTEMS

The use of generative systems in design, unlike traditional methodology, implies an indirect relation with the final product. Production, rather than being done directly, with the "designer's own hands," is mediated by a "generative system," as Fisher and Herr point out [1]. A generative system is based on an abstract set of rules, capable of producing a set of variations that to have meaning must be able to translate a set of qualities in the domain they are applied. A generative system is a solution-seeking process that, when interacting with context, like other open systems, approaches natural evolutionary processes. Generative is an adjective that derives from the Latin word *generare*, it designates one who generates or has the property to generate; is relative to generation; in linguistics, is associated to the theory of grammar that establishes a model, structurally describing the generation of discourse from a set of rules.

Generative systems, by generating solutions beyond the imagined, allow expanding creativity. According to John Gero, the most common concept of creativity does not consider the ability to develop possibilities during the generation of results, only considering the ability to improve the quality of an end product [2]. To create multiple solutions is necessary to use a method that intensively explores the possibilities of solving a problem, to generate unexpected and less familiar results for its creator. In this sense, the concept of emergency, which is less valued in traditional project, should be more valued in generative processes. Irving Taylor [3], classifies creativity into five levels: expressive, productive, inventive, innovative and emerging. In the context of generative systems, it is important to emphasize beyond the "emergent creativity" the "productive creativity", that is, the creativity that is contextualized in the domain of a technique, which allows controlling the project in the environment where it is generated.

Since the 19th century, mathematics developed new models capable of supporting generative processes, even if they were not denominated as such. The formal discipline of mathematics has changed paradigm with the introduction of computational processes, based on probabilistic logic and evolutionary processes, like random search, swarm thinking, among others associated with the development of artificial intelligence. Generative design processes are now successfully used by professionals and researchers in areas such as graphic design, mathematical optimization and materials engineering. Although the use of distributed computing allows creating numerous cycles of design possibilities, there is the difficulty of defining mathematically how to choose the best solutions. Generative systems have been used to search for solutions according to explicit objectives in certain types of problems, such as formal mathematical problems. Implicit search is used in other type of problems such as those that are not computable and those that are not clearly definable due to their complexity, as is often the case with many architectural problems.

The application of generative systems in Architecture is still residual, incipient, and in most cases, limited to visual issues or to the large urban scale. However, the development and application of generative processes is fundamental to solve the increasing complexity of design problems. Casey Reas is an example of the application of generative processes in visual arts [4]. The application of generative processes at the urban scale is associated with the seminal research of José Duarte [5], advised by William Mitchell, one of the precursors of computational design at MIT. This research has been continued in Portugal by a new generation of researchers such as José Beirão, Nuno Montenegro, Sara Eloy, Alexandra Paio, among others. In Brazil, Gabriela Celani, who was also advised by Mitchell, developed a research on this area, followed by other researchers such as Benamy Turkienicz, José Cabral Filho, Daniel Cardoso, Fernando Lima, Robson Canuto, among others. Recent research at UFRJ has been developed by Maria Angela Dias and Margaret Chokyu. In architectural practice, Franklin Lee and Anne Save de Beaurecueil have been applying such techniques in professional practice. However, this urban research has found few applications at the *mesoscale* of project—that is, the scale of public squares or of temporary spaces like pavilions. The argument is that to transfer this methodology to the design processes it is necessary to study the context of application of the generative processes. Thus, we intend to map new generative techniques such as cellular automata, L-systems, genetic algorithms and shape grammars, and test their application in design.

Given that generative systems act indirectly on results, they need to be interpreted, which requires developing a greater knowledge about the process itself. This implies not only using algorithms, but also being able to modify them. Thus, the generative systems approach is more of an open process, which finds its parallel in evolution, and in natural *morphogenesis*. Both synthetic and natural generative processes determine the evaluation of the results and the ability to modify and improve the algorithms. This evaluation and improvement can only be done through the information feedback of each generation. The improvement process allows the algorithm to no longer be considering a black box—for which we only know the results—to become a more transparent box in which we can interfere in its internal process of operation. In the process of developing solutions, one of the most important references is the difference between computerization and computation processes, as referred to by Terzidis [6]. The combination of both processes constitutes *algorithmic design*, addressed in a previous article [7].

1.1. On the application of generative systems

Mathematical optimization has a precise definition and is applied strictly. This optimization is often used in a final design stage, after the form is defined, to improve its performance. For example, a roof structure can be developed using the traditional design process and then the end form can be optimized to improve structural performance. The search for design form beyond strict criteria also obeys implicit criteria belonging to ill-defined problems. These type of problems, because they are not easily translatable into logical parameters and criteria, are often avoided in the mathematical reasoning of engineering. However, these criteria are implicitly considered in architectural design processes. In order to find solutions, experience with the tools and their application are important for evaluating the solutions. Thus, it is necessary to study the development of algorithms, acting on the system generation and not directly on the results as in the traditional processes. In this sense, it is fundamental to study, through experimentation, how the mentioned techniques—cellular automata, L-systems, genetic algorithms and shape grammars—act in the development of the form. Namely, to study with each type of algorithms, what is possible to develop and to what kind of design problems can be applied.

Problem solving depends on multi-causal manner of the processes and techniques used, but also on the nature and type of the problems. Therefore, it was considered relevant to identify examples of success using the techniques mentioned, to develop its application to concrete design problems. Identifying the type of problems and how they were solved in a particular context is important to solve new problems. Thus, the interest in learning and experiencing these techniques is justified, to validate them empirically. It enables the researchers to become familiar with these types of processes, stimulating knowledge through action learning process.

1.2. Problem/Project Context

The preliminary research intended identifying simple solutions, using the referred generative techniques to solve design problems in a small/medium scale. The goal was to use generative design to solve a design problem, for an intermediate scale space between $3 \times 3 \times 3$ m (min.) and $10 \times 10 \times 10$ m (max.). The idea was to map the existing techniques to solve problems for this space, considering few variables. It is also intended to identify how to interact with generation through feedback. The idea was to develop a process of interaction of the algorithm with the environment, instead of using an inaccessible black box. This adaptation allowed us to approach generative design problems specific to architecture. The preliminary research to the seminar studied this problem, trying to identify similar situations, variables and contexts already tested. The application of these techniques was studied, among others, in public

square projects, shelters, pavilions and constructive systems. The solutions depend on the tools and on the computational processes used. Thus, we sought to identify the search space and factors that interfere in this search, starting from generic to applied processes, in specific contexts.
The need to work with algorithms in a workshop of a diverse audience—graduation and post-graduation students, teachers and professionals from different regions—led us to choose *Grasshopper*, a parametric 3D modeller by means of visual programming. It runs as a plug-in of the NURBS modeller Rhinoceros. The algorithmic nature of Grasshopper allows implementing generative algorithms without the drawbacks of learning a programming language; while its open policy enables the access to a great number of software add-ons for experimentation [8].

2. CELLULAR AUTOMATA

Cellular Automata (CA) are systems that operate locally, which means that the state of each cell in a *n*-dimensional grid is defined by its neighbours. CA was invented and developed in the 40's by Stanislaw Ulam and John von Neumann and it has been applied in a broad number of fields, like Computer Graphics and Cryptography. However, besides its capacity of creating forms, it has not been so popular in the field of Architecture.
According to our traditional intuition, a system with simple input using simple rules would produce simple behaviour. However, CA is able to produce, from simplicity, emergent complexity, as it was demonstrated by Stephen Wolfram [9]. Given our historical moment's attraction to complexity, we can use CAs to generate it successfully.
The CA unpredictability: The workshop exercises introduced the students to the idea of *Form Making*, instead of *Form Finding*. The idea is to create an alternative for contemporary rational trends that advocate that architectural form can be undermined by simple quantitative criteria, like environmental performance, material use and cost reduction [10]. For each exercise with CA, the participants were invited to analyze the spatial and aesthetic qualities of each form and therefore interact with the system. One characteristic of CA is that it is impossible to predict its behaviour—there's no feasible way for foreseeing how a change in its rules would affect its product. Therefore, the way the designer interacts with the system differs from Albertian total control of the drawing. The participants were invited to approach the *form making algorithm* in a playful and horizontal way. *Estranging The Context* and *The Virtual Cocoon* are projects developed using CA based in a previous research about how computational morphogenesis can foster a new intuition to solve complex phenomena [11].

2.1. CA Workshop development

In the act of *form making*, there's an intention to not only create geometric patterns, but also geometry with an architectural meaning. In this way, during all stages of the process, architectural elements as slabs, windows, façades or ramps were introduced or produced by the CA system in such a manner to give architectural meaning.
Two groups of students produced two projects. The first group *Estranging the Context* used the agile form making characteristic of CA to produce architectural elements that inhabit movies. Each movie, according to the mood of its genre (Sci-Fi, Comedy, Horror, i.e.), received an architectural installation that responded to it and at the same time altered it. The second group, *The Virtual Cocoon* used a three-dimensional CA to produce a four-dimensional object that can be explored in Virtual Reality. Usually, a 3D CA is presented as a series of cubes side by side. With VR, it's possible to experience it as a malleable virtual object. In addition, the inhabitant of the cocoon can interact with it according to its position in space [Figure 1].

2.2. CA Workshop conclusions

CA is a very fast method to create complex and intricate form if there is a playful interaction with the designer. However, it is always necessary to have a qualitative understanding from an architectural point of view. There's a difference in between creating *geometric patterns* and *architectural drawings*. The first can easily be created by generative computational systems and be optimized according to quantitative criteria. The latter requires a cultural understanding of the spatial and aesthetic geometrical arrangement, being a very specific type of drawing that can only be understood through qualitative criteria offered by the history of Architecture as a discipline.

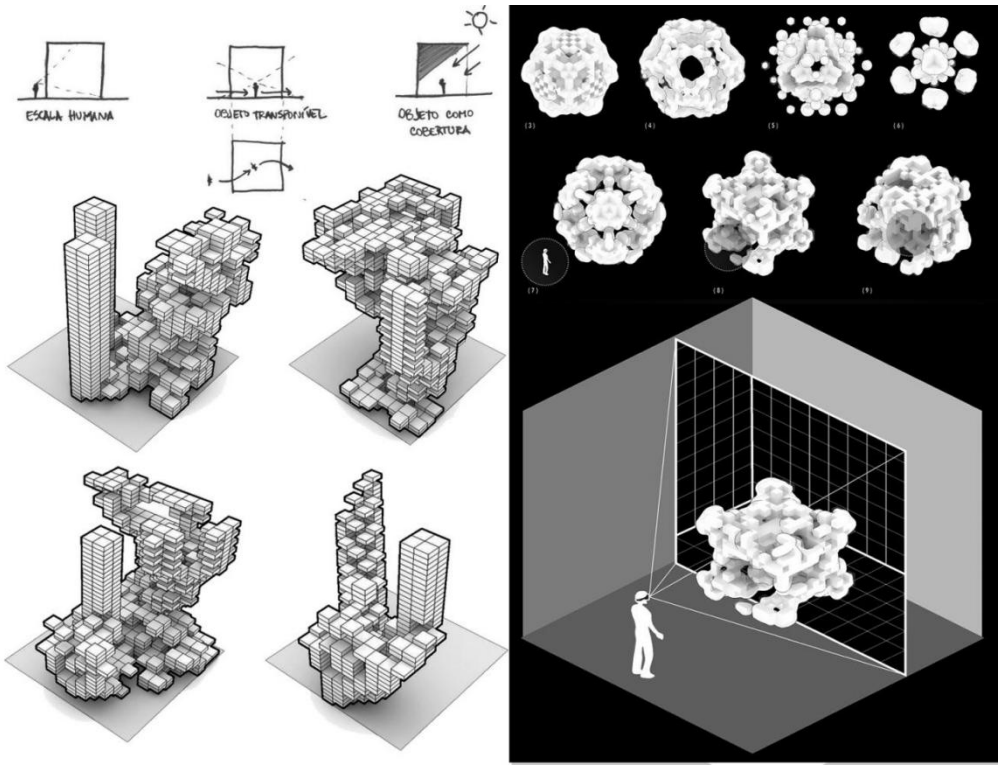

Figure 1. CA proposals, Estranging The Context, architectural elements that inhabit movies, group:Thatilane Loureiro, David Mendonça, Eugênio Moreira (Left), and The Virtual Cocoon a three-dimensional CA that produces a four-dimensional object to be explored in Virtual Reality, group Nicolle Prado, Isadora Tebaldi, Emilio Marostega (Right)

3. L-SYSTEMS

This research studied the application of L-systems to solve design problems, by looking in the literature on the theme, but especially, by working on the technical implementation.

L-systems are symbolic systems capable of generating growth structures, based on the ability of rewriting rules recursively. They were discovered by biologist Aristid Lindenmayer (1925-89) to describe the growth of simple species such as bacteria and algae. The technique is based on three concepts: (i) axiom, also referred to as seed, that is the initial string of symbols, which represents the initial state of the system; to this seed are applied (ii) production rules, through (iii) recursion, a repetitive computational method, in which each generation is executed from the previous one.

Originally, L-systems, are formal and deterministic systems that does not consider the interaction of rules with the context. An L-system is deterministic, when there is one, and only one, substitution rule for each letter of the alphabet [12]. In order to extend the application of this generative technique, we searched for methods using visual programming to develop context-sensitive L-systems. With the possibility of reacting to the context, L-systems evolve from a characteristic representation of a single artificial plant to the ability of generating differentiated species over time.

Among the generative design references that are relevant to L-systems, there are Fisher [1] and Agkathidis [13]. A more practical reference is the book by Prusinkiewicz and Lindenmayer [14]. There were few applications in architectural design, so we searched for papers, identifying the methods such as those by Paul Coates et al. [15].

While working with Grasshopper, we evaluated different software add-ons to generate L-systems, like *Rabbit* [16]. This add-on has an interpreter that translates the code into turtle graphics. It was found that the interpreter only accepts as input, letter symbols and is deterministic. An application was sought to apply functions with recursive loops, enabling non-deterministic L-systems. We tested *Hoopsnake*, used by Maycon Sedrez [17]. However, this application was unintuitive and presented some limitations. We tested *Loop* [18] and finally, we tested *Anemone* [19]. The latter is more intuitive, data order is maintained during loops and has an expandable list of data paths, enabling to change parameter values that affect each generation of form. It was thus possible to change the generation according to external factors and to implement turtle movement.

3.1. LS Workshop development

Two groups developed L-systems, one designated *Menger Revisited: Interactive Black-Box* and the other, *L-Aquele (A)braço (*What could be translated freely to the hugging trees*)*. The possibilities explored by the two groups are complementary. The first revisits the Menger sponge, proposing laws of recursive

interactive transformations, in this case a fractal that changes according to the presence of the user, in a virtual topological space [Figure 3]. It goes beyond the classical Menger sponge by breaking the symmetry in unpredictable, yet relational ways, without losing self-similarity. The second revisits the development of a tree-like structure, but in which its branches are re-configured according to human presence, changing the angles and planes of rotation according to this presence as an attractor [Figure 4]. The system is in equilibrium when the user is in the center of the space, and then is embraced by the trees. The first solution is virtually interesting, but difficult to be applied in the physical world. The second, despite following some stereotypes of a tree, has the type of mechanism that could be applied in an articulated physical structure.

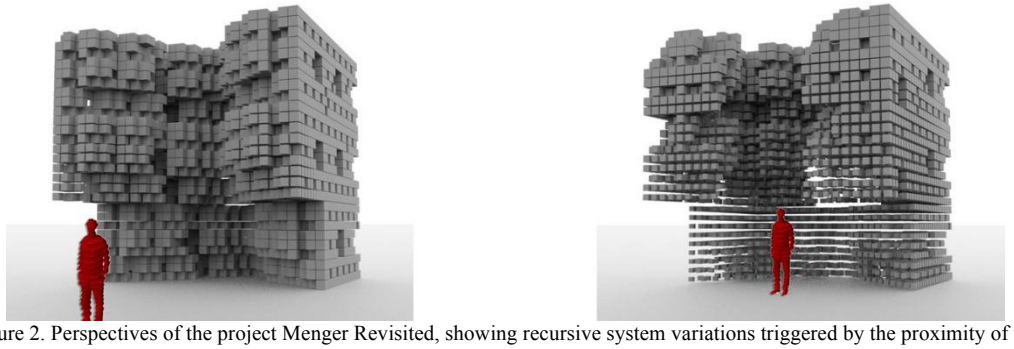

Figure 2. Perspectives of the project Menger Revisited, showing recursive system variations triggered by the proximity of the user.Group Fernando Lima, Aurélio Wijnands, Maria Eloisa.

Figure 3. Perspectives of the project L-Aquele (A)braço, showing the movement of the fractal trees as the user passes by. Group: Núbia Gremion, Erick Bromerschenckel, Daniel Wyllie

3.2. LS Workshop conclusions

Cecil Balmond designates “formal” as a platonic Ideal, reduced to a set of rules [20]. By contrast it designates “informal” as having non-linear design features. It also states that “the nature of reality is linked to probability, and that order is only a small local space, in a static state of a much greater random order.” In this sense, the projects developed confirms the possibility of (de)formalizing a technique or a (re)naturalization. This way both projects can continue to advance in the virtual world, as in the real analogue world. It is hoped that in the future some formalistic clichés may be lost, so that we will approach artificial-life systems, in-vitro.

4. GENETIC ALGORITHMS

The Genetic Algorithms (GA) are defined by a set of finite rules and operations that simulate the combination of the characteristics of the individuals of the same species, in order to select those that best satisfy the demands of the environment in which they are inserted. The algorithms are structured considering the main mechanisms present in the evolution of the species: genetic inheritance, random variation (crossover and mutation) and natural selection. Its application is related to processes of optimization solutions, being widely used by engineering to optimize topologically structures, parts, among others. In architecture, the GA has been used for space planning, optimization and generation of forms. Although it is a method of optimization, it is possible to use it as a generator mechanism capable of assisting the designer in the process of exploration and investigation of the space of solutions, being possible to obtain the emergence of unexpected and creative results. In this context, the algorithm must be configured to obtain favourable solutions independent of the lower or higher level of optimization obtained in the whole process, guaranteeing the flexibility of choice among the solutions generated. The main authors adopted as reference for development of the workshop were Holland [21], Bentley [22] and Mitchell [23].

Grasshopper has its own generic solver, *Galapagos*, with offers GA functionality. However, it is limited in terms of possible design problems it can tackle [24]. The Grasshopper add-on *Biomorpher* [25] was used because it allows the designer to interact with the algorithm during its execution. The designer selects the "parent individuals" from objective criteria (numerical values related to optimization) and subjective (evaluation of the composition aesthetics), directing the evolutionary process, once the "parents individuals" will be crossed to generate the "individual offspring".

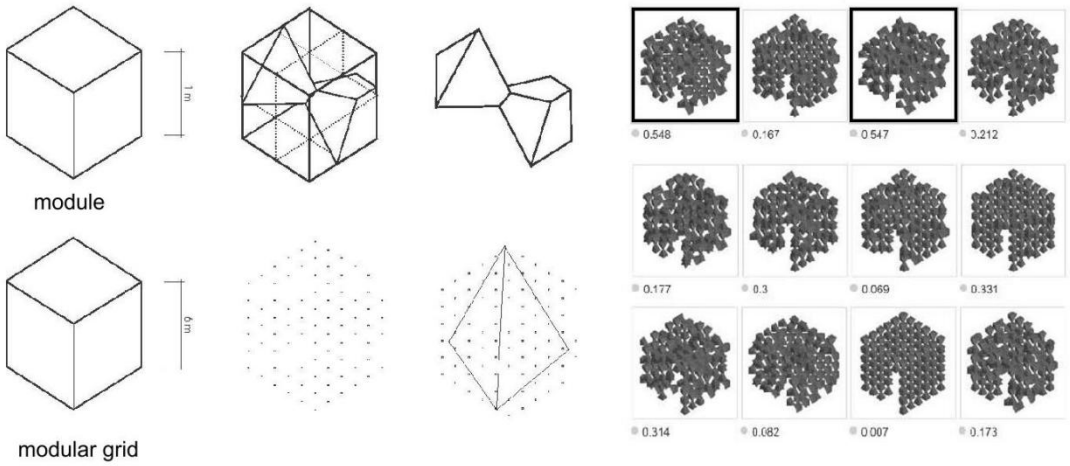

Figure 4. Group EA strategy and solutions. Group: Luciana Gronda, Igor Machado, Monique Cunha, Wellida Coelho

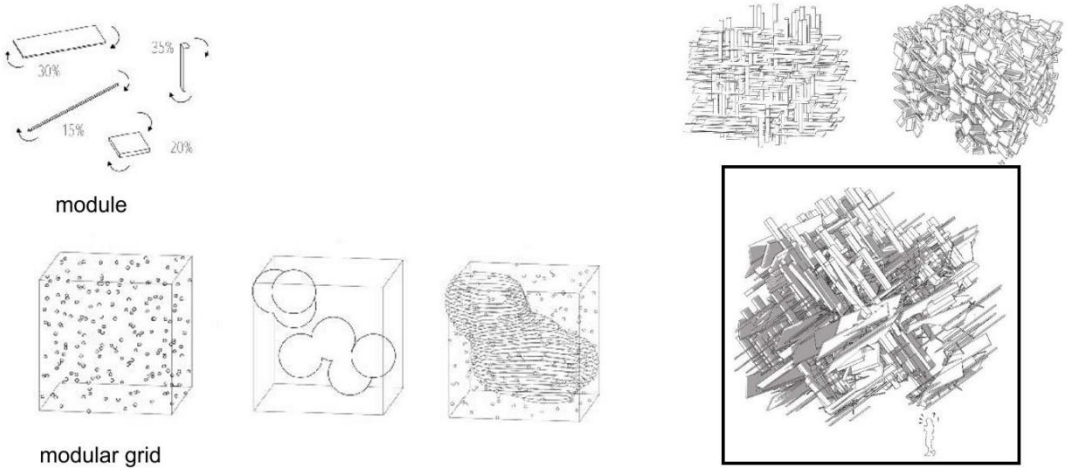

Figure 5. Group DD strategy and solutions. Group: Anael Alves, Loan Tammela, Felipe Lannes

4.1. GA Workshop development

Two groups developed GA, one designated *Evolutionary Aggregation* (EA) and *Dwell Debris* (DD). They adopted as criteria the shading, the contact generated by the grouping of components and the formal composition. As a strategy for structuring the GA, the EA group opted for a regular three-dimensional modular grid for the distribution of modules, ensuring regularity and orthogonality to the project [Figure 4]. The DD group opted for a cloud of randomly distributed points in space, giving irregular arrangements for the overall form [Figure 5]. In both strategies, the void inside the structure was obtained by subtracting points that intersect a given solid located at the center of the structure. The modules used by each group were quite distinct from each other. In EA, a single geometry capable of rotating on its own axis has been defined, generating different options of modules to be positioned in the grid of points. DD defined four geometries that also rotated in their own axis, generating a diversity of modules to be grouped and distributed in the cloud of points. The EA group programmed the GA to find solutions that had the largest contact surface between the modules and the largest projected shadow area. On the other hand, the objectives of the DD group sought to find the greatest number of intersections among the modules, greater volume in the lower dimensions and less shading.

4.2. GA Workshop conclusions

The difficulties encountered by the groups during the workshop were not related to the use of the GA, but to the manipulation of the Grasshopper components and the structuring of the modelling algorithm. The first algorithm generated unwanted results. In this way, the groups needed to re-formulate both the strategy and the algorithm. This review contributed to improve the understanding of the design problem and also the recognition of the need to create mechanisms to control the generative system. In summary, the genetic generative system not only automated the process of generating forms, but contributed to obtain more knowledge about the problem proposed by each group.

5. SHAPE GRAMMARS

As there was no established software to run shape grammars in our context, we decided to develop a definition for Rhino/Grasshopper to emulate them. To do so, we got back to some Stiny´s initial texts [26], [27], [28], looking for elements to structure the definition. We identified a recurrent description that we organized into the triple ordinate [Equation (1)]:

$$G = (v, r, d) \tag{1}$$

Where G is the grammar, v the vocabulary, r the set of rules, and d a set with the derivation, i.e., the sequel of rules to be applied. In Stiny´s classics, G, v and r are always mentioned. However, d appears only in a latent form, never explicit, be as markers, or as different set of rules for each stage of development, or as rules applicable to specific shapes that only appear after a certain rule. Thus, we understood it better to assign an independent set to the derivation sequel for a more coherent general structure, with the benefit of also allow a more straightforward implementation.

5.1. SG Workshop development

Our algorithm is understood as a protocol of concatenation between the three sets, while each set is allowed independence in its inner definition, as long as the protocol of coordination is observed. It is a loop with v as input, applying the rule from r indicated in d, accumulating the result in v. While v and d are simple lists, r is more complex. It takes two shapes, indicated as initial and final, memorizes their relations of scaling and other Euclidean transformations, and then apply them to the input shape. This relation rule can be established either through Grasshopper coding or by drawing in the Rhino canvas, with initial and resulting shapes as Brep inputs to this algorithm. There are some holdbacks to point out. The definition presented did not have any ability to recognize the shape it was dealing with, so it could operate a triangle related rule to a square, for instance. However, as work in progress this is a next step, along with the recognition of emerging forms.

This overview structure turned out simple enough to encouraged the teams to develop their own definition. After the initial presentation and exercises, one of the teams developed a Shape Grammar relating music to façade [Figure 6], while the other focused on designing a shelter [Figure 7]. At the end, they either turned back to using our definition changing the rules, or to a very similar shape/rule dedicated definitions. In either case, many designs were explored before deciding for the final one. As speculations over this work frame, we identified a few potentials, technically and conceptually.

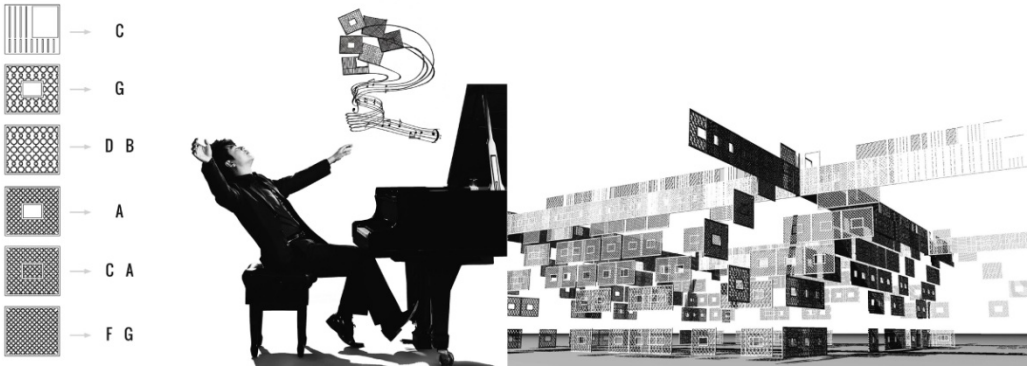

Figure 6. Shape Grammars, re-scribing compositions. Group: Daniel de Sá, Rodrigo Scheeren, Leonardo Prazeres

Figure 7. Shape Grammar, sheltering grammars. Group: Lais Kaori, Rebecca Borges, Davi Ramalho

5.2. SG Workshop conclusions

Recognition of emergence could be easier done inside the operation of the rule, for which the result is known. The work frame definition with a set of shapes containing the initial and resulting shapes makes this easier.

The derivation ordinate plays a fundamental part in Shape Grammars. Usually they are taught as shapes and rules, and derivation remaining rather obtuse or unclear, as if it could be just chance. All the randomness has to be incorporated at the derivation. Therefore, full randomness probably leads to meaningless design. Some amount of randomness can be used and still generate some meaningful design. In this case, the derivation set must be treated as a decision tree, with different paths encoded, and the randomness used to change between them.

6. CONCLUSIONS

6.1. Roundtable discussion

The final roundtable discussion was intended to reflect on the process conducted by tutors, seeking feedback from all participants. It allows us to reflect on the discoveries that preceded and happened during the event, valuing the association between theory and experimentation. The seminar began with tutorials, and then techniques were empirically tested during the workshop, evidencing the application of knowledge in design. The acquired practical knowledge was validated in design, expanding the initial theoretical knowledge.

The roundtable was triggered by questioning: how do the discovered tools interfere in the “form” found? This question was posed to each tutor, to reflect on the technique he used, identifying the advantages and limitations of each one.

Ernesto Bueno (EB): The presentations we have just seen from the projects are interesting and quite different. Groups explored generative processes differently. The eight projects have in common the fact that they are interactive, all based on simple rules, that in a first interaction are not complex, but that explore the potential of the computer, to take us to levels in which our capacity of natural and geometric comprehension would not take us. What is interesting, and has already been discussed since previous LAMO events, is how to use the computer as a design tool, which goes beyond the simple CAD drawing board. Parametric drawing can be very advanced for certain tasks, but it is still an automation of a series of processes, which ultimately depend too much on end-user software skills. Perhaps generative design does not have these limitations.

Gonçalo Castro Henriques (GH): What are the limits and advantages of L-systems?

EB: The main limitation was how to use the turtle graphics. We can see this in both L-system projects. The one of the tree that embraces us, made a more literal reading, although progressively got out of the rules of turtle graphics and created trees that open and close with an attractor, with rules that can be considered L-systems rules, but they do not exist in the classic turtle movement. The Menger sponge-based project is a recursive system which in essence is an L-system, but it was conceived from the beginning using different techniques. That made possible to deepen the exploration on what can be done with a fractal. This is fantastic.

The limitations of the software are the ones we have always have. That's why I'm a fervent adept at learning to program, to develop our own tools and not be limited to what the industry offers to us.

GH: When we started to prepare exercises on L-systems, the definitions we found did not predict interaction with the environment. This was a limitation that might explain why this technique, although old, is currently less used. We saw the possibility of changing the application of the technique through stochastic algorithms. However, the examples that we found resorted in programming only, which was difficult to implement in such an event. So we look for and tested several applications for recursive cycles in Grasshopper with the ability to interfere in the growth with external factors. Both groups were able to interfere with growth cycles and create interaction with the environment. The results, as Ernesto commented, may be somehow literal—with difficulty being applied to the project—but I believe we have opened a new path, introducing new possibilities for research on the subject.

Victor Sardenberg: I found very interesting that usually in a workshop we start from a concept or an intention to generate a product and then the students discover what tools they need for designing this product. This time, we first found out what these tools can generate as a product, we explored them and we let the students speculate on what they can do.

Regarding the cellular automata, I have the impression that is almost a *cellular autonomous*, which is very difficult to make and to interact with. It works more like a form generation from which one tries to find an architectural sense. Thus, according to the process used, most projects ended up being close to a representation of that process.

Perhaps the genetic algorithm, by not forcing such a specific form, brought the most freedom to produce results, which were not so expected. So, I would say that the advantage of the automata is the ability to generate many shapes, very fast; and the biggest difficulty is the lack of a clear architectural sense and the difficulty of interacting with it.

Jarryer De Martino: Regarding genetic algorithms, the result was good to show the possibility of using these algorithms which—although being optimization tools—can be used as generative tools. It was reinforced what we presented about generative design and this possibility, of being used as a shape-generation engine. Reinforcing what I had already foreseen in my PhD thesis, about this possibility of not only generating forms, but generating knowledge during the creative process, generating information, being able to review the process and its feedback. Genetic algorithms itself, reflecting on what Victor mentioned, works very much as a mechanism. I think that, because it does not have this link with the mapping of form, or modulation, it targets forces that are acting on that mechanism. Maybe it gives that greater freedom, because one defines genes and rules. It is a mechanism of automation of selection within that process. So this workshop gave a good idea of the possibility of generating forms using genetic algorithms.

GH: Daniel, how was the experience of Shape Grammar? Not being your field, you decided to explore it in what it seemed like a rich experience.

Daniel Lenz: Shape grammars have been a little strange to me, because they dictate a set of rules, but this can be done blindly. We needed to have an object that was already coherent, so that this generated grammar could produce new coherent results. Other than that, this is speculation and just like in the cellular automata, you're going to have to cross your fingers and wish for it to work to get somewhere. However, some aspects were cleared, including the supervision that is needed for establishing the sequence of rules. I think that the interaction with the grammar is much out there.

I found it excellent that in both groups, after establishing the working mechanism, each of them worked to develop their mechanism, or to customize it. It was a very interesting experience, mainly because we started with a certain disadvantage. In other techniques there was already a machine working in Grasshopper, but in grammar there was not. So, there is this great merit of the teams. I think that it was very clear that what will make the architectural coherence possible is the supervision at the moment that the rules are defined. Therefore, it is important when we decide that we are going to stop here and start on other side. An interesting research to be done is whether this supervision process can be coded, or whether it can be automated in something that envelops this grammar. This is the question that remains from this workshop about grammars.

6.2. Final considerations

The event was an investigation that began many months ago with the aim of form finding using generative tools. In the opening presentation was mentioned Kostas Terzidis [6], which distinguishes processes and design tools using the computer. In research in general the computer is being used as a process of computation. But it was important to show, and discover, the other side of the tool, of computerization in the artistic sense, and therefore considered more archaic—contrary to what many think—the tooling or computerization can coexist with the computation process, and this junction brings a new meaning to the form. The tool is somehow more tied to the context and limitations and potentialities of the project, albeit implicitly. But this process is not blind, benefiting from previous

experiences, of trial and error, that even though they cannot be coded, they work as heuristics, which can help to define the computational mechanisms that we need to use in design. So this use means rescuing the design intuition prior to the use of computation.
In this event we got lost well, experimenting with these form-finding tools, that may be useful in the future. The tools influence the processes, they are not neutral: they influence the final result and the solutions found. We thank and congratulate everyone for the excellent result. We thank all the organization, the tutors, the lecturers, the participants and volunteers and everyone present. It is undoubtedly a joint effort and we are all to be congratulated.

Figure 8.. >> Em busca da forma, sistemas generativos: group photo by Nico Batista

Research Credits

Coordination of the event: Gonçalo Castro Henriques, Andrés Passaro, Maria Elisa Vianna. Preliminary applied research: Cellular automata: Victor Sardenberg, L-systems: Gonçalo C. Henriques, Ernesto Bueno, Genetic algorithms: Jarryer De Martino, Shape Grammars: Daniel Lenz; Interaction: Lucas de Sordi, Laura Lago. This text is the result of a joint research that was initiated at LAMO. In the prospective research on generative systems applications, participated: Andrés Passaro, Caio Cavalcanti, Cintia Mechler, Elisa Vianna, Erick Bromerschenckel, Gabriel Gaspar, Giordana Pacini, Isadora Tebaldi, Julia Nodari, Lais Kaori, Loan Tammela, Nicolle Prado, Rebecca Borges, Roberto Costa Matta, Pedro Engel, Thiers Freire, Vinícius Lucena, among others. This first research fed the initial text, and vice versa. The text began to be written by Henriques, and later commented and reviewed with the contribution of Sardenberg, Martino, Lenz and Bueno. This joint paper helped to define the theme, the methodology and to search for the tools. Scientific Committee: Andrés Passaro, Daniel Lenz, Denise P. Machado, Eliane Bessa, Gonçalo Castro Henriques, Guto Nóbrega, Maria Angela Dias, Rodrigo Cury: Federal University of Rio de Janeiro, Brazil; Ernesto Bueno: Positivo University, Curitiba, Brazil; Gabriela Celani: Campinas State University, Brazil; Jarryer De Martino: Federal University of Espírito Santo, Brazil; José Duarte: University of Pennsylvania, United States; José Pedro Sousa: University of Porto, Portugal; Mauro Chiarella: National University of the Coast, Argentina; Rebeca Duque Estrada: University Stuttgart, Germany; Victor Sardenberg: Gottfried Wilhelm Leibniz Universität Hannover, Germany. Photography: Caio Carvalho, Luciana Willi, Nicolas Batista. Video: Diogo de La Vega, collaboration Laura Basile.

References

[1] Fisher, Thomas; Herr, Christiane. “Teaching generative design”, in: Proceedings. Fourth International Conference on Generative Art, 2001. Available:papers.cumincad.org/data/works/att/c78f.content.pdf
[2] Gero, John, “Creativity, emergence and evolution in design”, *Knowledge-Based Systems*, vol. 9, no. 7, pp. 435-448, 1996. doi:10.1016/S0950-7051(96)01054-4
[3] Taylor, Irving “The nature of the creative process”, in: *Creativity: An examination of the creative process*, P. Smith, Ed., New York: Books for Libraries Press, 1972, pp. 51-101.
[4] Reas, Casey., “Process/Drawing”, *Architectural Design*, vol. 76, no. 4, pp. 26-33, 2006. doi:10.1002/ad.290 See also: How to Draw with Code, 2012: https://youtu.be/_8DMEHxOLQE
[5] Duarte, José Pinto, “A Discursive Grammar for Customizing Mass Housing: the case of Siza's houses at Malagueira”, *Automation in Construction*, vol. 14, no. 2, pp. 265-275, Mar. 2005. doi:10.1016/j.autcon.2004.07.013
[6] Terzidis, Kostas. *Algorithmic Architecture*. Oxford: Architectural Press, 2006.
[7] Henriques, Gonçalo Castro, “Algorithmic Architecture: Techniques, processes and fundamentals”, presented at: IV ENANPARQ, Session 39. Digital design and fabrication in architecture: Teaching, research and challenge, PROPAR - UFRGS, Porto Alegre, Jul. 2016. Available:www.researchgate.net/publication/305827549
[8] Bueno, Ernesto. “Grasshopper”, in: *101 Conceitos de Arquitetura e Urbanismo na Era Digital*, Braida, F. et al., Orgs., São Paulo: ProBooks, 2016, pp. 118-119.
[9] Wolfram, Stephen, *A New Kind of Science*, Wolfram Media, 2002.
[10] Carpo, Mario. “Digital Darwinism: Mass Collaboration, Form-Finding, and The Dissolution of Authorship”, *Log Magazine*, 2012.
[11] Sardenberg, Victor. “Processos Emergentes em Territórios Informais”, B.Arch. Diploma monograph, Mackenzie Presbyterian University, São Paulo, 2013.
[12] Martins, João Manuel dos Santos, “Sistemas de Lindenmayer: Modelação de árvores com recurso ao Maple”, M. S. thesis, University of Porto, Mar. 2008.

[13] Agkathidis, Asterios, “Generative Design Methods - Implementing Computational Techniques in Undergraduate Architectural Education”, in: Proceedings. the 33rd eCAADe Conference, vol. 2, 2015. pp. 47-55.
Available:papers.cumincad.org/cgi-bin/works/paper/ecaade2015_18
[14] Prusinkiewicz, P. and Lindenmayer, A. *The Algorithmic Beauty of Plants*. New York: Springer-Verlag, 1990. Available:http://algorithmicbotany.org/papers/abop/abop.pdf
[15] Coates, P., Appels, T., Simon, C. and Derix, C. “Current work at CECA”, In: Proceedings. Fourth International Conference on Generative Art, 2001.
[16] Dimitrova, G., and Mandov, K. “Intro to L-systems,” *MORPHOCODE*, 17 June, 2014. [Online].
Available:http://morphocode.com/intro-to-l-systems/
[17] Sedrez, M. and Meneghel, R. “Projeto paramétrico com fractais no detalhamento de uma fachada,” *PARC Pesquisa em Arquitetura e Construção*, vol. 4, no. 2, pp. 22-29, Dez., 2013. doi:10.20396/parc.v4i2.8634548
[18] Turiello, Antonio. “Loop”, *Food4Rhino*, 27 Nov., 2013. [Online]. Available:www.food4rhino.com/app/loop
[19] Zwierzycki, Mateusz, “Anemone”, *Food4Rhino*, 14 Dec., 2015. [Online]. Available:www.food4rhino.com/app/anemone
[20] Balmond, Cecil., Jannuzzi, M. and Smith, R., *Informal*. London: Prestel Verlag, 2002.
[21] Holland, John, *Hidden order: how adaptation builds complexity*. New York: Basic Books, 1995.
[22] Bentley, Peter, *Evolutionary Design by Computers*. San Francisco: Morgan Kaufmann Publishers Inc, 1999.
[23] Mitchell, William, *An introduction to genetic algorithms*. Cambridge - MA: The MIT Press, 1999.
[24] Rutten, David. “On the logic and limitations of generic solvers”, *Architectural Design*, vol. 83, no. 2. pp. 132-135, 2 Apr. 2013. doi:10.1002/ad.1568
[25] Harding, John. *Bioz* on the generation of the Chinese lattice designs”, *Environment and Planning B*, vol. 4, no. 1, pp. 89-98, 1977. doi:10.1068/b040089
[27] Stiny, G. and Mitchell, W., “The Palladian Grammar”, *Environment and Planning B*, vol. 5, no. 1, pp. 5-18, 1978. doi:10.1068/b050005
[28] Stiny, George. “Introduction to shape and shape grammars”, *Environment and Planning B*, vol. 7, no. 3, pp. 343-351, 1980. doi:10.1068/b070343