From Designing Objects to Designing Processes: Algorithms as Creativity Enhancers

Günter Barczik[1], Winfried Kurth[2]
Brandenburg Technical University Cottbus, Germany
[1]Department of Architecture, Chair of Contextual Building Design and Construction, Professor Inken Baller
[2]Department of Computer Science, Chair for Graphics Systems, Professor Winfried Kurth
[1]guenter.barczik@tu-cottbus.de, [2]wk@informatik.tu-cottbus.de

We discuss the creative potential of algorithmic design processes in architecture which use computer-based tools through presenting student work employing these techniques. We propose ways of further amplifying these creative possibilities by employing evolutionary design strategies.

Keywords*: Digital aids to design creativity; generative design; education & practice; CAAD curriculum.*

Interests, hypotheses and observations

As designing architects and educators, we believe we can enhance designers' creativity by not designing objects or buildings directly, but designing algorithmic processes that then can create buildings.

Through describing recent student design projects from our faculty of architecture we will argue that

a. Using algorithms in design can lead to designs that would not have been conceived of without the algorithms – especially in terms of the complexity of a design and its components, and of these components' combinations.
b. Using algorithms in design can allow for easier adjustment of a design to a given situation's specifics, especially once those start to change dynamically.
c. The effects described in (a) and (b) can be amplified by using the aid of computers in executing the algorithms.
d. Students can not only be more creative, but also much more structured, precise and articulate, as rationalizing a design process so far that it can be written as an algorithm requires unusually thorough analysis.
e. Students can develop their interpretative skills as the algorithms firstly often lead to unforeseen results that need to be interpreted in terms of possibilities and appropriateness and secondly produce three-dimensional structures that are a kind of proto-architecture requiring interpretation as to its possible development into architecture.
f. Students can enhance their understanding of the role of tools in design processes, especially how the choice of tools influences the designs.
g. Students can enhance their ability to bring specialists into the design process at an early stage to contribute special skills and knowledge.
h. The effects described in (a) and (b) can be amplified even more by using the aid of meta-algo-

rithms like evolutionary algorithms to conceive of the design algorithms themselves.

i. Radically rationalizing a design process can strengthen the role of intuition.

Background, motivation and educational and research program

Most plants and animals are not only much more complex, effective and economic than man-made structures, but also not so seldom more beautiful. One major difference between the man-made and the natural is that the former is the product of conscious conception while the latter is the product of unconscious and random growth and evolution. The recent advent of powerful tools allows us to artificially model these natural processes and to start exploring how architecture might benefit from semi-natural processes.

We have for several semesters now been carrying out design project classes in our architectural department using the 3d-modelling and programming environments Blender & GroIMP which work, respectively, with the programming languages Python & XL – all of which are public domain and open source. GroIMP & XL are being designed at our university's computer science department [Kniemeyer 2007], in order to create an intuitively usable language for the specification of models of growing plants. This new approach has also already shown its versatility and manifold uses [Kurth et al. 2005 and Kurth 2007]. XL, which stands for 'eXtended L-system language', unites the possibilities of rule-based model specification by Lindenmayer Systems [Prusinkiewicz & Lindenmayer 1990] and of object-oriented programming and, coupled with the software GroIMP (see www.grogra.de) offers considerable advantages for growing, handling and exporting 3-d structures and their relations.

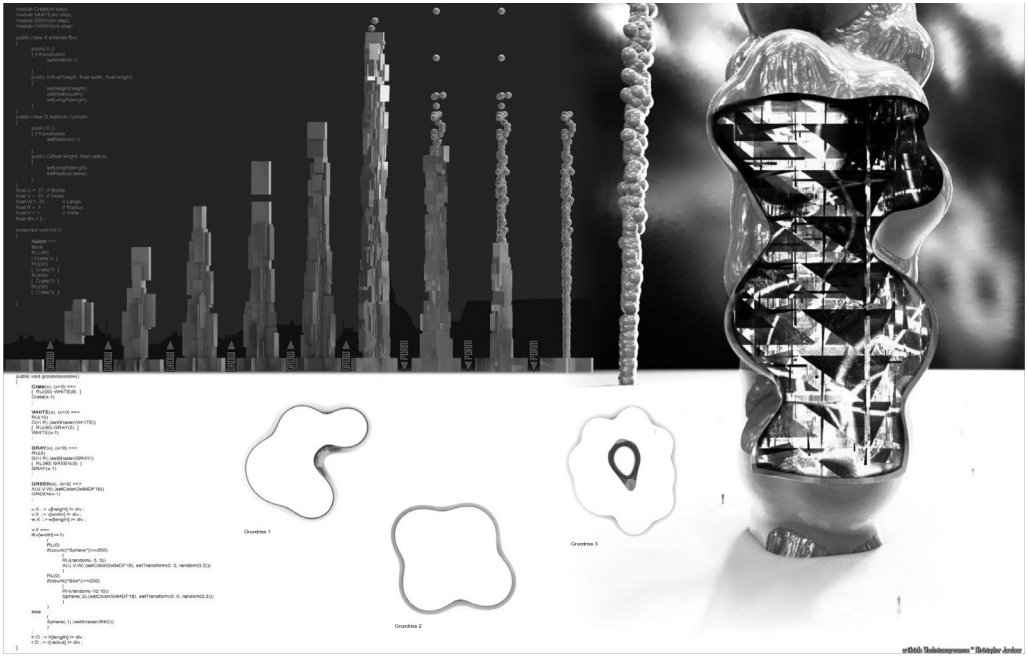

Figure 1
High-Rise study grown in GroIMP using XL. Student Project Winter 06/07. Christopher Jarchow. Cottbus University

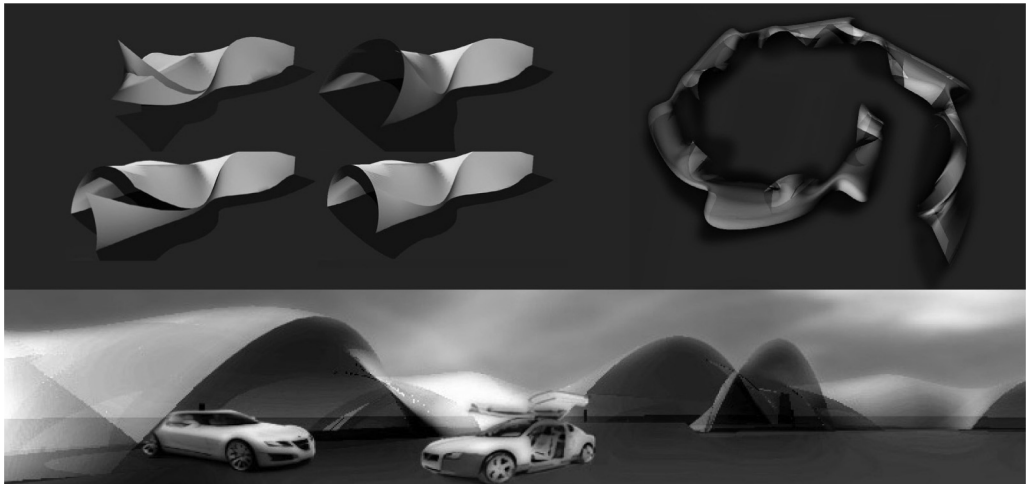

Figure 2
Envelope variations forming a spiral plan to create a car showroom grown in GroIMP using XL. Student Project Winter 06/07. Christopher Jarchow. Cottbus University

Designs that could not have been conceived of without design algorithms

Using fixed instructions – algorithms – to have surprising results be generated has been a fertile ground for experimentation throughout the history of modern art. To name but one example, the conceptual artist Sol Lewitt did not draw his Wall Drawings himself but thought up the instructions for doing so and had painter craftsmen execute those. In effect, Sol Lewitt programmed the craftsmen; this artist of the avant-garde was a programmer avant la lettre.

Our interest lies in the unexpected outcomes an author of instructions might be confronted with: albeit precise prescriptions, the result is something unconsidered. In a design course, students were introduced to XL and GroIMP and in playfully exploring the possibilities of thereby growing three-dimensional structures quickly encountered comparable 'nice surprises' (figs. 1-3). The development of growth algorithms and grown structures oscillated between precise intentionality and creative interpretation of unforeseen results.

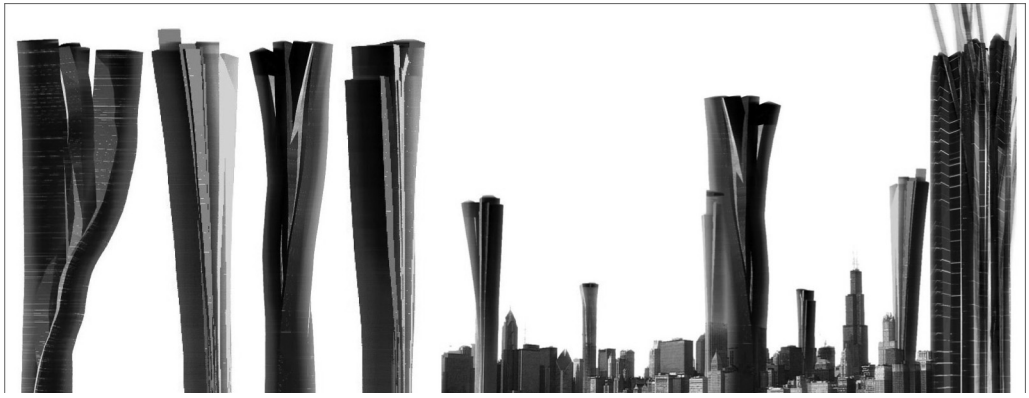

Figure 3
High-rise variations grown in GroIMP using XL, inserted into an existing cityscape. Student project winter 06/07 Christopher Jarchow. Cottbus University

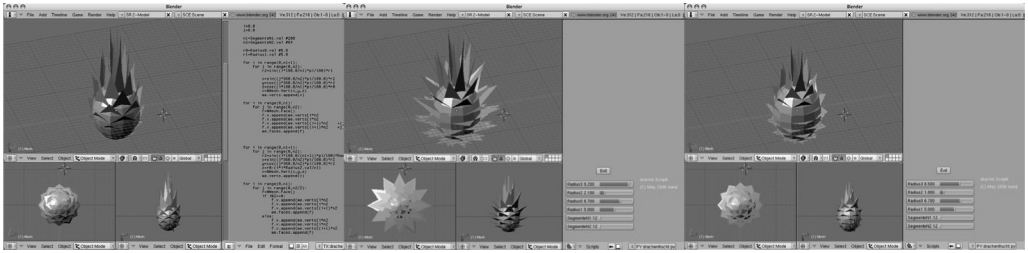

Figure 4
Dragon fruit created in BLENDER through PYTHON code. Student project summer 06 Manja Arnold. Cottbus University

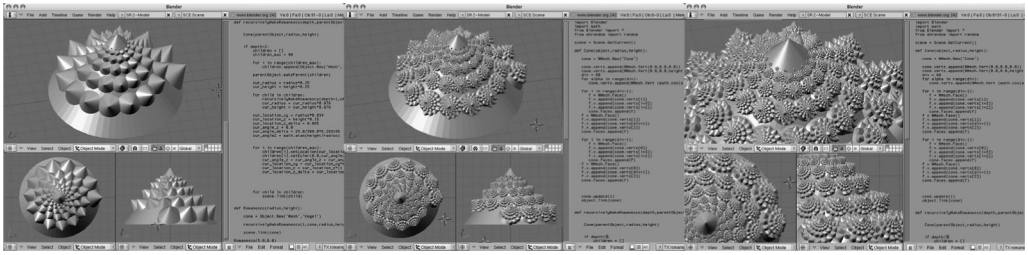

Figure 5
Romanesco created in BLENDER through PYTHON code. Student project summer 06. Asja Kasdorf. Cottbus University

With computers, the complexity and intricacy of designs can be increased enormously, quickly leading to structures that would have been very difficult, if not impossible, to conceive, let alone subsequently handle, manually.

Designs that adjust to situational specifics

Architecture is mostly designed for a specific set of parameters – a building's program, its size, budget, site etc. – usually frozen at the start of the design process. Change of parameters means redesign. If instead of a fixed object a parameterized procedure is designed that generates the object, the procedure can be re-executed with the changed parameters, and an adjusted object be generated.

This was studied in a design class where students were firstly familiarized with algorithms by analyzing fruits and formulating a detailed set of instructions for re-creating an abstract form of them. Parameters were introduced into the algorithms and set to 'un-natural' values to create variations of the fruits that do not exist in nature. These programs were written first in natural language and then translated into Python and executed in Blender (fig. 4 & 5).

The students went on to write algorithmic descriptions and instructions that would generate not fruits but proto-architectonic spatial structures that could then manually be further developed into buildings. We describe two examples:

In the beach sports centre project a formation was generated that not only contained spaces for beach sports, but that in itself conveyed 'beach-ness' (fig. 6).

The process starts with placing a number of predefined playfields on a pre-defined site, according to specifics resulting from an analysis of sand dunes but with an amount of randomness. The playfields are developed into the minimum volumes necessary for practicing the desired sports. In compliance with the 'dune-ness' specifics, parallel trusses follow and are enclosed in a partially translucent skin on which

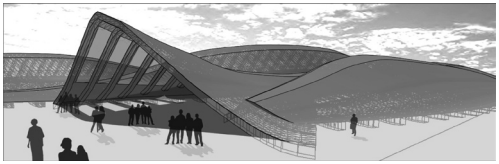

Figure 6
DuneScape - centre for beach sports. View towards entrance. Student project summer 06 Kirsten Galow. Cottbus University

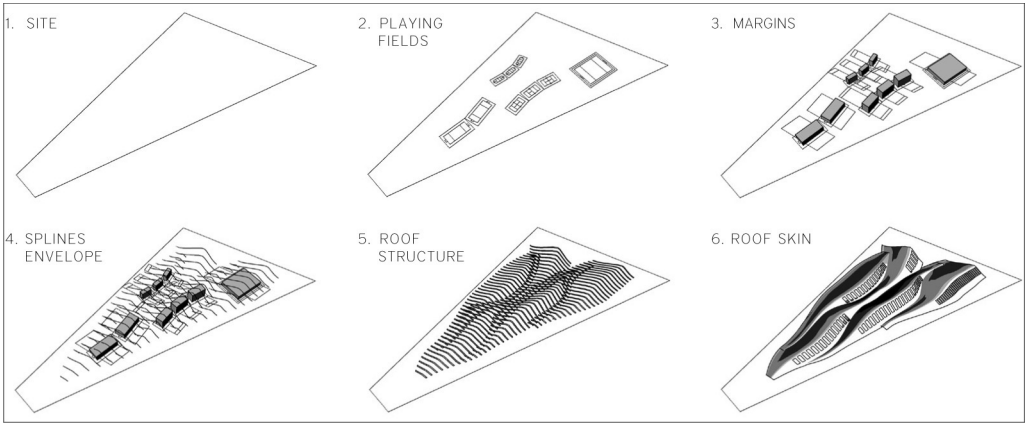

Figure 7
Algorithm for DuneScape creation. Student project summer 06 Kirsten Galow. Cottbus University

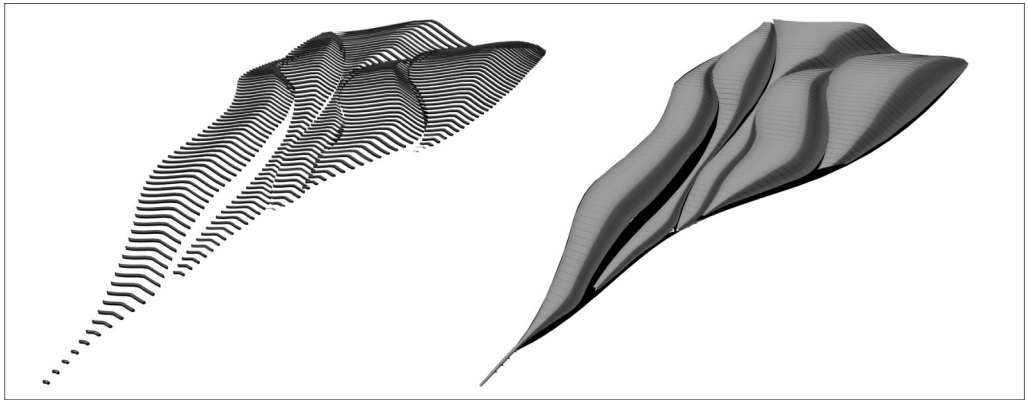

Figure 8
Finished DuneScape Ripple Structure & Skin. Student project summer 06 Kirsten Galow. Cottbus University

it is still possible to walk (fig. 7) - resulting in an artificial dune scape which serves both as a sports building and as a public park (fig. 6 & 8).

The aim for the public square was to generate public outdoor urban space and to manipulate the relief so that uses like a café, a restaurant, a theatre/performing space, sport fields etc. could be integrated (fig. 9). Additionally, parallel strips should be readable in the relief as reminiscence to past uses of the site.

To generate such a topography, various uses were assigned sectional profiles (fig. 10). These were then connected to generate long strips which were in a next step extruded horizontally into a relief topography. Several of those strips were created and placed alongside each other according to pre-set parameters like type, number and position of use. In several following steps, the strip relief topography was smoothened out, resulting in the desired public square (fig. 11 - 13).

Figure 9
Urban relief topography. A public outdoor space with various uses inserted onto a relief topography. Student project summer 06 Hubert Kopiec. Cottbus University

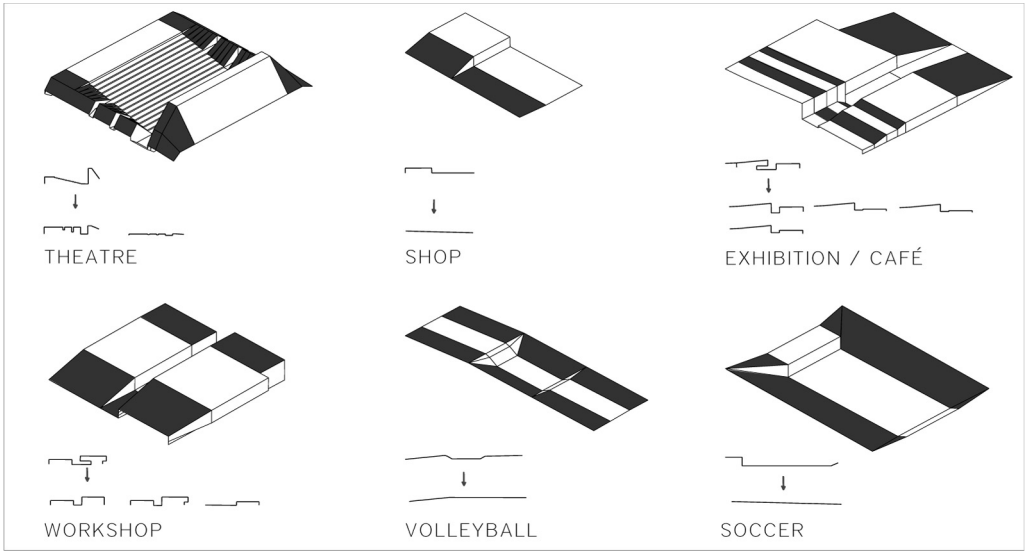

Figure 10
Urban relief topography. Assigning sections and their variations to different uses. Student project summer 06 Hubert Kopiec. Cottbus University

In both projects, the creational processes, the algorithms, will always create the desired type of building even with altered initial parameters like number, kind and positioning of use, and shape and configuration of the site.

Although in both the XL/GroIMP and the Python/Blender based design projects the analytical and executional rigour and discipline were not easy for the students - not all were able to write algorithmic descriptions - once such algorithms had been

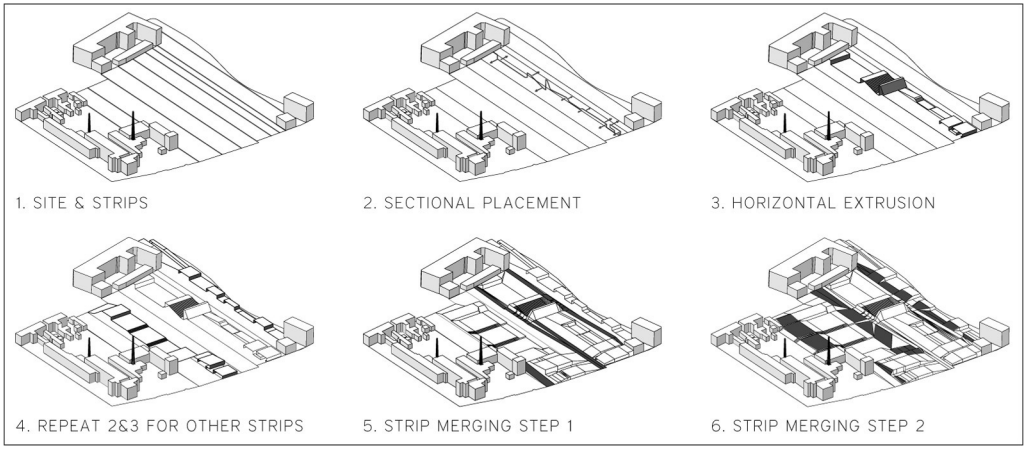

Figure 11
Urban relief topography. Algorithm for the creation of the relief topography. Student project summer 06 Hubert Kopiec. Cottbus University

Figure 12
Urban relief topography. Longitudinal section showing different uses. Student project summer 06 Hubert Kopiec. Cottbus University

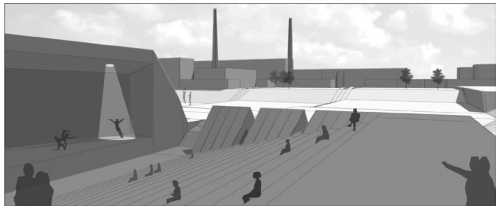

Figure 13
Urban relief topography. View onto outdoor performance space. Student project summer 06 Hubert Kopiec. Cottbus University

formulated, they resulted in more creative possibilities, especially when transposed into a programming language and executed by computers.

The complex algorithms for the sports building and for the public square were executed by hand, as their translation from a graphical into a computer language unfortunately exceeded the programming skills acquired in one semester.

These descriptions and explanations should illustrate the validity of our hypotheses (a) to (f). As to (g): the students themselves quickly started enlisting professional help: the actual process of translating natural-language programs into computer languages was rather simple for students of computer science.

Outlook: Applying genetic algorithms to evolve the design algorithms

For the future, we aim to artificially evolve the generative processes instead of designing them manually. This technique of evolutionary algorithms is widely used in engineering for the optimization of processes.

The application to architectural design necessitates measuring how successful a three-dimensional structure generated by the random algorithm is as a building of the specified use. Unfortunately, this is not at all trivial, as it requires evaluation of size, structural stability, air and temperature movement in and through the structure, its overall appearance in terms of proportion and shape, to name only a few criteria. While the first evaluations might be manageable, the latter ones are a matter of continuous debate even with conventionally designed architecture.

We propose to leave the crucial selection to humans who monitor the evolutionary process, that is to have them pick the ones they consider the most successful – even if that choice might be based on intuition and only be partly rationalisable. Although this may sound somewhat weird, quite a number of contemporary architectural practices design in almost such a way – though they would probably refrain from describing their methods that drastically. Consider, for example, the Christian Dior building in Tokyo (fig. 14). The final design would by no means have been foreseeable at the start. With only the building's envelope given, internal organization and facade underwent changes upon changes. Selecting a generation's survivors happened very much according to 'I will know it when I see it'.

Such human-controlled selection of mutated structures was also used by the biologist Richard Dawkins to create his virtual 'biomorphs' [Dawkins 1986] who described in pathetic words his feelings when he first had let evolve these simple but organic-looking patterns: 'Nothing in my biologist's intuition, nothing in my 20 years' experience of programming computers, and nothing in my wildest dreams, prepared me for what actually emerged on the screen' [Dawkins 1986].

Would not applying the principle of 'I'll know it when I see it' to evolve algorithms that create buildings enable the fitness criteria themselves to

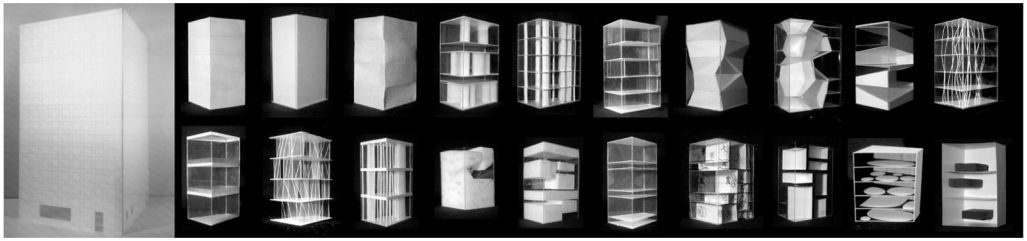

Figure 14
Christian Dior building in Tokyo by Kazuyo Sejima & Ryue Nishizawa (SANAA). Final model & selection of study models

evolve as the designers acting as human monitors of the evolutionary process would come to learn more about the specifics of a design task and the requirements for its solutions from generation to generation of both rationally and intuitively selected design proposals, ever improving on their observations, interpretations, critiques and selections while discovering ever more space for their intuitions' playfulness?

As the random mutation would ensure that unforeseen options would be brought into play, the creative capabilities mentioned earlier ensured by the use of algorithms to create buildings would be further amplified – as stated in our hypotheses (h) and (i).

So far, they remain hypotheses only. We have just begun to put into place the framework necessary for testing them. Although we very much want to refrain from making unjustified predictions: artificially evolving algorithms that grow three-dimensional structures that might be interpreted as buildings is what we would predict for the future of CAAD.

Acknowledgements

We very much thank the students Manja Arnold, Kirsten Galow, Christopher Jarchow, Asja Kasdorf and Hubert Kopiec as well as all other students involved in the design classes and to Dr. Riklef Rambow of Professor Eduard Führ's Chair for Architectural Theory at the Department of Architecture of Cottbus University and Reinhardt Hemmerling at Professor Winfried Kurth's Chair of Graphical Systems at the Department of Computer Science at the University of Cottbus for all their contributions and continously productive critique.

References

Dawkins, Richard: 1986, The Blind Watchmaker. Norton, New York.

Kniemeyer, Ole: 2007, Design And Implementation Of A Graph Grammar Based Language For Functional-structural Plant Modelling. Btu Cottbus, Doctoral Thesis (forthcoming).

Kurth, Winfried: 2007, Specification Of Morphological Models With L-systems And Relational Growth Grammars. Image - Journal Of Interdisc. Image Science, 5, Special Issue (Submitted).

Kurth, Winfried; Kniemeyer, Ole; Buck-Sorlin, Gerhard: 2005, Relational Growth Grammars - A Graph Rewriting Approach To Dynamical Systems With A Dynamical Structure. In: Banâtre, J.p.; Fradet, P.; Giavitto, J.-l.; Michel, O. (Eds.), Unconventional Programming Paradigms. Lecture Notes In Computer Science 3566, Springer, Berlin, 56-72.

Prusinkiewicz, Przemyslaw; Lindenmayer, Aristid: 1990, The Algorithmic Beauty Of Plants. Springer, Berlin.