Teaching 3D Generative Virtual Architecture with VIPA CONSTRICTOR

Philipp Seifried, Jochen Hoog[1], Christoph Falkner[2]
Institute for Architecture and Design, Vienna Technical University, Austria
http://www.gbl.tuwien.ac.at
http://vipa.adm.at
[1]hoog@atelierprozess.net, [2]falkner@tuwien.ac.at

CONSTRICTOR is a novel teaching and learning tool to introduce students of architecture to the fundamentals of computational design. It was developed within an EC funded e-learning project, VIPA (Virtual campus for virtual space design Provided for European Architects). In this paper, we describe the general structure and technology of CONSTRICTOR and its application within the course "Synthetic Constructions II") that aims at familiarizing students with concepts like cellular automata, swarms, shape grammars, path finding etc. CONSTRICTOR allows students to observe simulations, navigate through them and change the parameters that define and control them, thereby developing an understanding for the rules that govern their virtual environments. They can also change the geometry used in the simulations and work with and export their results. Students can modify the simulations at code level, combine them or write entirely new simulations.
Finally some of the student work performed during and after the course is described, and an outlook for future work and further developments around the VIPA courses and the constrictor software will be given.

Keywords: *Blender; generative design; learning platform, virtual space design.*

The VIPA Project

The teaching and learning software Constrictor was developed within an EC funded e-learning project, VIPA (VIrtual campus for virtual space design Provided for European Architects). The objective of the last two years was the development of a virtual campus containing an e-learning and research platform for architecture universities. For the learning management aspects a Moodle environment was installed. It supports many of the course management and administration features needed, as well as standard courseware, assessment functionality, communication tools and a resource repository. The idea of a collaborative and immersive lab was the additional focus of the project.

In a first step the multiuser online application OpenCroquet seemed to be the best platform for the project. Crouquet and its scripting language Squeak are primarily targetted at the education sector. We built up some applications for generative design, i.e. cellular automata or autonomous steering behaviors, and used the plattform in a seminar with 15 students. In the end technical problems with the platform (release 0.3) and its instability stopped further developments

"A didactical difficulty encountered is interconnected with Croquets programming language Smalltalk/Squeak. Squeak, although suited for novice programmers, has not reached very broad distribution." So the consortium decided to develop a tool which allowed an easier access for using generative methods in an architectural design-process. In this case it was important to get a connection to a conventional 3d modelling software. The choice of Blender and Python made it possible to edit results of ongoing works in a 3d-modelling environment and export them to other applications. Another benefit was the greater stability and an ongoing development of the open source package. The VIPA Constrictor in its current version can be thought of as a stand alone application based on Blender. It is much easier in use and offers the students a graphical user interface, which makes it more attractive for students to work with scripts, as the varied results of the student projects show.

Anatomy of the VIPA CONSTRICTOR

VIPA Constrictor is a simple graphical scripting system, written on top of VIPA's 3D courseware framework, a collection of Python modules written for Blender which ease the development of generative applications.

Constrictor was designed to be a didactical alternative for architectural courses that aim to introduce students to fundamentals of computational design but cannot afford the time required to teach programming from the ground up. As such, it bridges a gap between basic demonstrations written for VIPA's courseware – which allow limited exploration of their subject matter by giving students an interface to change a running simulation's parameters – and full Python scripting, for which VIPA provides helpful modules, but which may be well outside of the scope of many courses on computational design.

Constrictor scripts are built by assembling instructions through a container-based graphical interface within Blender, rather than by writing code using a text editor. This provides for a very easy experience for novice programmers, by encapsulating syntactical and lexical rules in an easy to use graphical representation, making many common errors – e.g. syntax errors or infinite loops – impossible or very hard to achieve.

A Constrictor script can be thought of as a rule-set for one or more types of entities, such as cells in a CA or autonomous agents in flocking simulations. Rule-sets consist of statements and conditional statements, which are executed in sequence and which in combination define entities' behaviours.

A statement in VIPA Constrictor can be thought of as a single instruction to an entity, which is executed in every step of the simulation. Conditional statements consist of a block – i.e. a container – of expressions and a block of statements which are executed if the expression block evaluates to true. Within any given block statements and expressions can be added, removed or re-arranged.

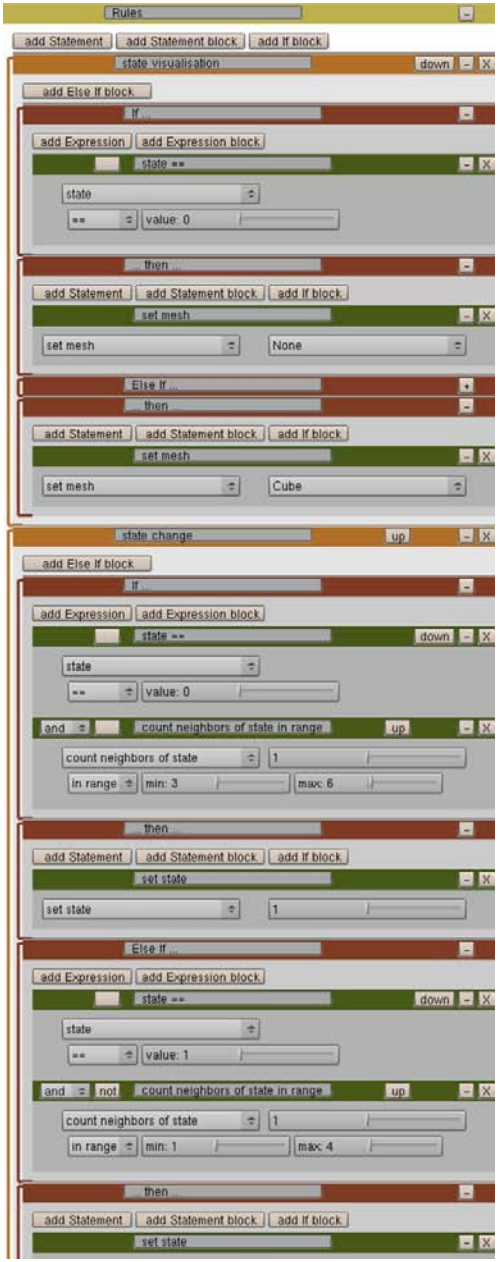
Figure 1 shows a complete Constrictor script which handles state change and state visualisation in a CA. As each block in Constrictor – be it a single statement or a container holding statements, expressions or other nested containers – can be collapsed and expanded, the script has been structured into two sections for better readability.

The first section of the script handles the visualisation of cell states. It can be read as

"If a cell's state equals zero, set its mesh to 'None', otherwise set its mesh to a 'Cube'".

The second section deals with state transitions and contains two conditions.

Figure 1
A complete Constrictor script



If a cell's current state is 0 and between 3 and 6 neighboring cells are in state 1, its state changes to 1. If the above condition is not met, if the cell's current state is 1, and less than 1 or more than 4 neighboring cells are also in state 1 (note the "not" toggle-button in the expression's header), then the cell's state changes to 0.

As Constrictor scripts' visual complexity quickly rises with the complexity of their rule-sets, students are encouraged to plan ahead before implementing their scripts. However, Constrictor offers a few features to make scripts more readable:

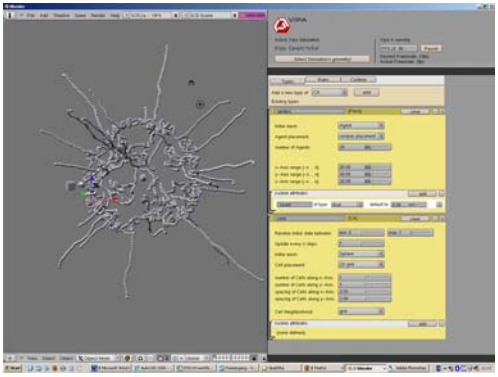
As stated above, every type of container can be collapsed to its header, hiding any attributes or containers nested within it. Nested containers are indented and drawn in different colors based on their indentation. For every type of container that can hold other containers – i.e. a block –, a border is also drawn, indicating the block's scope.

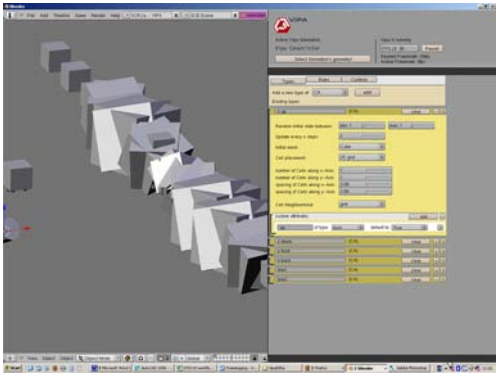
In addition, every element in Constrictor has a header indicating its function. This header can be changed by the user to specify the element's purpose more precisely. Indeed, it is possible to use this feature to place multi-line comments throughout a script by assembling several empty statements in series, collapsing them and then editing their headers.

CONSTRICTOR in use / Synthetic Construction 2

The course "Synthetic Construction II" was the second run of a prototypical course within the VIPA project. It is weighted with 2.4 ECTS points situated in the master studies. It is a blended learning course for the basics of generative design. Several generative systems, like cellular automata or steering behaviors for autonomous agents, are the main topics of the course.

15 students participated the course last winter semester at the vienna technical university. They where expected to implement small projects using the Constrictor software.





Figures 2 and 3 Screenshots: VIPA Constrictor

The first unit contained an introduction into the blender software.

Blender is a modelling application rather than a three-dimensional environment but it offers good ways to use scripting in Python. It provides a wide spectrum of modelling tools to create and manipulate three-dimensional objects, similar to Maya, 3DS Max or Cinema 4D.

Objects created in Blender can be exported to prevalent file formats. This fact allows students to use the software also for their designs in general and to work on with their results with other programs.

In the following units the students received an introduction on how to use and work with Constrictor. After that it was their turn to define a project theme and to use a suitable algorithm to get good results for their design project. In contrast to the previous course (prototypical run I) the barrier to work with code and to use algorithms to generate objects was smaller. Right from the beginning the students "played around" with the tool, and tested different outcomes. During the period of the course the VIPA Constrictor was constantly improved and advanced.

During the course different results were presented and discussed to define, improve or correct the design approaches of the individual projects. The students tried different ways how to use code and how the code affects the outcome. The graphical interface to work with predefined code fragments and the possibility to combine different fragments of code increase the quality of the outcome. To get three dimensional results – rendered by Blender in real-time - encouraged the students to work with code, even if they just started working with scripts. So they had more time to really work on the concept and rework and rethink the outcome in architectural aspects. That means to alternate between working on the script and testing the three-dimensional result.

Compared to the prototypical run I (Synthetic constructions I) the results show a deeper level of working intensity among the students. The acceptance of Constrictor was good.

The VIPA Constrictor provides for students a tool to generate three-dimensional results right from the beginning. That encourages the class to learn using code as designers and minimises concerns about programming. (which is not instructed to architects at the TU Vienna).

At the end of the course the VIPA consortium decided to test the online multiuser platform Second Life as an immersive three dimensional collaborative environment.

Despite the fact that Second Life is not an open source platform, and has limitations in the amount of objects (prims) and users, it offers stable in-world tools to create and communicate within the virtual environment. VIPA has founded a virtual online

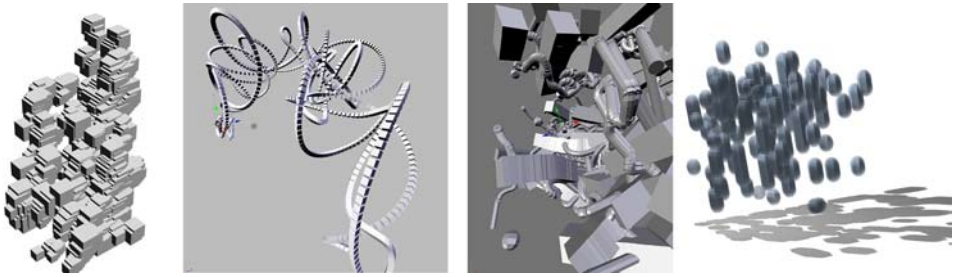


Figure 4
Student works: H. Schreckensberger, W. Grossmann, J. Haunschmidt, S. Windischbauer

group in Second Life. The group VIPA provides some basic information about the project and can be entered within the virtual world.

Second Life was tested as a collaborative place and communication platform. The students have to present their projects on a virtual whiteboard and had to communicate via chat. The students as well as the lecturers and some guests were standing around the screen visualised through their avatars. The presentation went well, even though the communication via chat was not good enough to really talk about the projects.

Future work

After running the VIPA project some new technologies should make it possible to establish a virtual 3D lab. One future aspect will be the combination of Constrictor with a collaborative multiuser plattform like Second Life or the new OpenCroquet release 1.0. Porting the Python modules on which Constrictor is built to these technologies will be relatively straightforward, but also its scripting paradigm can be emulated within these platforms with moderate effort.

The Institute of Architecture and Design – TU Vienna is going to test the social multiplayer platform Second Life as a 3d immersive environment.

The first step will be an organisation and presentation of an exhibition of student works, interlaced with a "real" exhibition shown in Vienna (Archdiploma2007 in October 2007). Based on those experiences a design studio is planned to use Second Life as a tool to work with and as a collaborative environment for the design process.

It seems to be a necessary step to test an existing 3d virtual environment and to gain experiences using it as a tool and a possible future design domain.

In terms of Constrictor's current state as a Blender based solution, improvements can be made in terms of streamlining the interface and making the system more flexible by allowing to link the output of expressions to various statement's parameters. However, care must be taken to balance the system's versatility against its essentially "fail-safe" nature, as we consider its inherent improbability of creating show-stopping errors within rule-sets to be a strong asset.

With Constrictor as a tool for generative systems and second life as 3d environment we have two components to offer an enlarged design-course for "virtual space design"

Acknowledgements

We would like to thank the members of the VIPA Consortium: The architecture departments at the universities of Aalborg (DK), East London (UK), Lubljana (SL) and Vienna (A) as well as our industry partners adm (Atelier für digitale Medien) (A) and the FH Joanneum (A). We would also like to acknowledge the support we have received by the European Commission.

References

Blender: http://www.blender.org/.

Grasl, T.; Falkner C.; Kühn C.: 2006, Easy access classes for three-dimensional generative design, "Communicating Space(s)", eCAADe 2006.

Lombardi, M.: 2005, Croquet Learning Environments, http://www.opencroquet.org.

Second life: http://www.secondlife.com/.