Jean Brangé
École Spéciale d'Architecture
254 B d Raspail
75014 Paris FRANCE
jean_brangé@esa-paris.fr

An Introduction to Algorithms and Numerical Methods Using Common Software

Jean Brangé describes an approach using Photoshop, VRML and C4D to allow architecture students to manipulate geometric and algebraic formulas, recursion, random functions, statistics, splines, the fourth dimension and other complex mathematical concepts.

Introduction

This paper, which was first presented at the Nexus 2002 conference in Óbidos, Portugal, will explain how to introduce algorithms and some complex mathematical concepts to architecture students without getting into practical math, but by using practical software functions which implement these concepts, allowing the students to manipulate math with the tools they are used to, and without the need to buy an expensive “Mathematica”-level software. In about 80% of the cases the students have no special interest in mathematics or in programming and one of the challenges is to figure out how to get them to work on these topics without boring them. I do not intend to prove or show that using numerical methods through algorithms is THE way to produce architecture, neither that c-space needs to be architectural, this would need another more complex paper; the bibliography contains some pointers to these issues.

I have developed this approach in my class in Paris, at the Ecole Spéciale d'Architecture, using Photoshop, VRML and C4D to get the students manipulate geometric and algebraic formulas, recursion, random functions, statistics, splines, fourth dimension, etc. The purpose of the class is to familiarize the students with mathematical and geometrical concepts outside the acknowledged “Euclidian 3 dimensions” they have been educated in and make them aware of the power of the mathematical, geometrical or analytical models, together with the necessity of defining a good “model” as a basis of a good architecture project. My definition of model is not the 3D model nor the wooden model but the logical or conceptual model. The goal is to get the students to define sets of rules that can generate a shape. The design is in the process and the definitions of the rules. This course is not an architecture course, so even if I try to connect the students' work to their actual studio projects, I do not spend time in the validation of the original constraints or intuition choices; we concentrate rather on the validity of the process.

Context

I started this course in the second semester of 1999. For some time I have been involved in redefining the general pedagogy of the school regarding the role of computer graphics (CG). This has led to a redefinition of the Computer Graphics courses in the school (France is quite behind in this topic compared to the US), with a set of introductory and instrumental Computer Graphics courses in the first and second years so that the students are all computer literate by the end of the second year. This includes “Office”-like, PSD and Internet software in first year, and 2D drafting and 3D modeling in second year. This would be the answer to “how” do we use computers. Starting in the fourth year we have set up a seminar on simulations, which is the “why” we do use computers.

Lab

The computer lab is divided into three sections. The main part is the classroom: divided in two parts face-to-face and equipped with twenty seats – half Macs, half PCs – with two video projectors and two teaching seats. On each side of the classroom we find two rooms: one for students working on their final thesis or a long-term work, and one used as a self-service room. Related to the computer lab we have an Intranet based on the First-Class workgroup solution, which gives us collaborative work places for any course, web publishing facilities, and a general bulletin-board system. The workgroup server is also available through the web and is replicated on an external server, so we can access the bulletin board from anywhere. Dial-up access to the Intranet is also available through Apple remote Access protocol. The software used in this hardware and network solution has been chosen for its pedagogical efficiency. We did not follow the standard market shares but the ease of use, understanding and appropriation by the user. The set of software we base our courses on includes Photoshop, GoLive, Word, Excel, iMovie, VectorWorks, Cinema4D and Archicad. Other softwares are available and installed – AutoCAD, Architectural Desktop, 3Dmax, Artlantis, Radiance, Illustrator, etc. – but these are not taught within a specific course, and are more used as self-training solutions.

Reform

One of the goals of the reform we are setting up is the strict definition of the vertical coherence of the disciplines in the curriculum and their inter-related horizontal connections. Horizontal connections are available through the fourth-year seminars, which allow outsiders to participate. For instance, the “Figuration” seminar has produced conferences on the evolution of drawing techniques up to Computer Graphics systems.

Example 1. Automation as “fast to execute repetitive task” and “batch” is one of the main characteristics of the computers. Nowadays young people do not feel the power of modern computers since they are already accustomed to it, so they are not surprised when a renderer computes a picture in a few minutes. However, they are shocked when they find out that some settings can blow the rendering up to a few hours. By asking computers to produce quickly repetitive tasks they rediscover why programming is really useful. Photoshop is the best tool I have found to explain this concept to architecture students. Photoshop’s macros and batch utilities are a simple and yet powerful development system to work on pictures, which can be used for picture manipulation but also for image analysis.

Photoshop also provides a large set of predefined functions, which can be called up by commands in the “filters” and image “adjust” menus. The Photoshop macro program also makes it possible to include some user interaction by input dialogs. This allows students to understand the sequential programming style and the use of variables.

The main exercise consists in getting a large set of pictures, acquired by any means available to the students – screenshots, digital video, cellular automata software, etc. – and to apply a repetitive process to these pictures that will lead to the definition of a 3D shape, built from the translation in 3D space of the isovalue curves of the pictures analyzed.

The first part of the exercise is to build a Photoshop script, which will transform the raw picture into a set of 2D curves. The script is a simple macro recording of a succession of steps defining image manipulation functions, which can include, brightness and contrast, resolution, selections, focus, color manipulation, noise filtering, black/white levels, thresholds, etc. The list is not limitative.

From the filtered image, we can select a range of values to extract a 2D curve or surface. This curve is then sent to a 3D Software for 3D generation of a surface or a volume. Once the script is defined it can be applied to a large set of original pictures to produce a new set of 2D curves or surfaces. The process is set up. The script can be saved and transmitted, modified, included in more complex constructions. Most of the script steps can include stop points and user interaction for fine tuning or parameter input. The result is the automated production of sections which can define a 3D mesh from a bunch of raw data. Two examples of this exercise are joined in the graphical section.

This exercise implements the automation, the sequence, the function calls, the user interface including user input and data exchange, but we still miss branching, looping and Object Oriented concepts. VRML will be used for that.

Example 2. VRML as derived from the Silicon Graphics OpenInventor Format is a toolkit for developing interactive graphics application and includes some algorithmic features such as looping and conditional branching through specific VRML Nodes, such as the Switch, Level Of Detail, and Sensors. Further, the VRML Script Node enables the student to create specific functions that share parameters with the other VRML nodes.

Thanks to these Algorithmic tools we are now able to deal with some more complex notions as recursion and fractals while using VRML. VRML includes scripting among the other fundamental modeling elements, such as “box”, “cylinder”, or “texture”, etc. Using scripts we have access to nearly unlimited object-oriented programming functionalities inside our models, but this tends to be quite difficult to transmit, since we need to get into real algorithm codes and language syntaxes. We can still approach some basic algorithm concepts, without writing complex codes. Some VRML nodes implement basic algorithm concepts as branching, switching, functions and loops.

Simple loops can be implemented by using a time sensor, that is, loops based on a running clock, whose increment and step can be set at our convenience.

Other sensors like proximity sensor or visibility sensor, as well as the “Level Of Detail” node are based on a “if/then/else” sequence,

```
if I see the object
then do that
else do that

if I am within a certain distance of an object
then do that
else do that
```

The switch node implement is a “switch/case” concept :

```
Case x:
1: show object 1
2: show object 2
3: show object 3
```

By using these specific nodes together, students can achieve complex process definitions, where the model responds dynamically to the user interaction. The most basic interaction is the navigation of the user in the model.

Observations

Computation power. The data analysis forces the student to work quickly on a large amount of files. Since we are able to produce rather large models and since we use them in real-time interactive software, we quickly reach the power limit of our available hardware. Both exercises point out the power (and the weakness) of the computers.

Trans-disciplinarity. Some students use the results of this course in some other classes, not only in the architecture studio. Reciprocally they apply in this course ideas and concepts borrowed from other disciplines.

Curiosity on these new territories for architects. Some students choose to pursue the work while preparing their final thesis a project where design process is really an issue, and usually this involves other types of “built” environment. The unbuilt, ruins, light, networks, media, cyber space, etc., may be used as new architecture materials.

Limits. The basic limits that we encounter are algorithm, math, statistics and time.

We do not have enough time to have a real algorithmic course of even only 12 hrs. The data analysis we simulate are only useful as a simulation of the process. The real process is not done on real data, as we do not have access to the real data. On the math level, we are limited to simple arithmetic functions when we have to code them ourselves, and we cannot go into the detail of the more complex ready-made functions available in the software we use.

Another limit is the amount of prior architectural work needed to start the process. The course is only an elective course. Last semester, the topics of the exercises were individually connected to the students’ projects in their architecture studios and the results were less interesting in the CG course than in the Architecture Studio. The students explored a lot but were not able to produce a specific part of their studio work using the CG course as requested. The best results were obtained by students who had quickly jumped into the “processes” to set-up a starting point for their architectural project. This has led to interesting development in the studio, but we were not able to establish a bidirectional link between the studio and the lab. The studio’s “time” caused a huge slowdown in the middle of the semester (time was needed to refine the projects, and at that point, the dexterity of the students with their new “dynamic” C4D software was not highly enough developed). This should change in the next couple of years since all the students will be provided with a good knowledge of 2D, 3D and animation, acquired in the first two years. For us, the best results were achieved when the CG course had its own “architectural” exercise disconnected from the current project studio.

Conclusions

Are we in presence of a new tool used with the same methods and concepts, or is it the same tool while the methods are new? Can we break the ancient rules? Are we setting up new rules? Can we set up “rule breaking” as a rule?

Efficiency. By using our chosen tools we can bring the students to a faster understanding of the computer graphics concepts. They will therefore spend more time on “how” and “why” to use these tools, rather than on “where is the function I need?”, or “is there a function to do what I

want?”. We have observed that some students are switching from 3DSmax to Cinema4D, although they have worked with 3DS for a few years. They discover that a simpler but better designed tool is more efficient. All the students admit that the C4D interface is much less time consuming and gives more direct access to the tools and functions available. The result is that in a few sessions students with no previous 3D modeling knowledge become able to model their Architecture Studio project and to build a final full polygonal, nurbs, Boolean, texturized, and lighted model.

Process as an element of design. The basic idea of these exercises is to formulate in non-graphical language how a situation can be approached and how we can interact with it. This defines the process, and this can set the roots of the design intuition.

Measure. We might use this process as a system to try to find in the object the definition of the object itself. If a human being can be considered as a sample of a living species defined by its DNA sequence, the DNA sequence can be imagined as a computational object with the data, the program and the processor all included in one entity. The fact that the DNA sequence is made out of the simplest alphabet we could imagine does not give us the clues to the infinite combinations that are available within these sequences. But still the DNA defines itself. In the same way, by being able to define a proportion system, an organization scheme, a procedural pattern in terms of process, and letting this process model a shape, the shape will contain in some case the basic structure of its processing development in a human readable form.

Bibliography

Publications:

Architects in Cyberspace. 1995. *Architectural Design* 65, 11/12.

BENEDIKT, M. 1993. *Cityspace, Cyberspace, and the Spatiology of information*. Austin: University of Texas Press.

MITCHELL, W. 1994. *City of Bits*. Cambridge MA: MIT Press.

MOUSSA, Arnaud, et al. 1997. Meta-architecture en France. *kubos*.
<http://www.kubos.org/vrml/metarchi.html>

Websites:

Centrifuge. <http://www.construct.org/motet>

Marcos Novak. <http://www.centrifuge.org/marcos>

Objectile. <http://www.objectile.com/>

Assymptote. <http://www.asymptote.net>

About the author

Jean Brangé teaches computer-aided architectural design at the Ecole Spéciale d'Architecture of Paris. His own research focus on the applications of computer science in architectural drawing and design. He collaborated in the development of some specific softwares (Radiance, Octree and Arris, among others). He is also involved in the development of internet technologies for professional purposes and is a specialized consultant in PLM for the French association of the world-wide-web users ("Afnét").