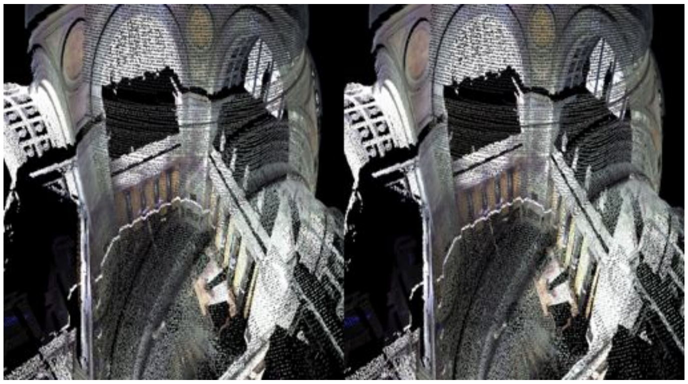
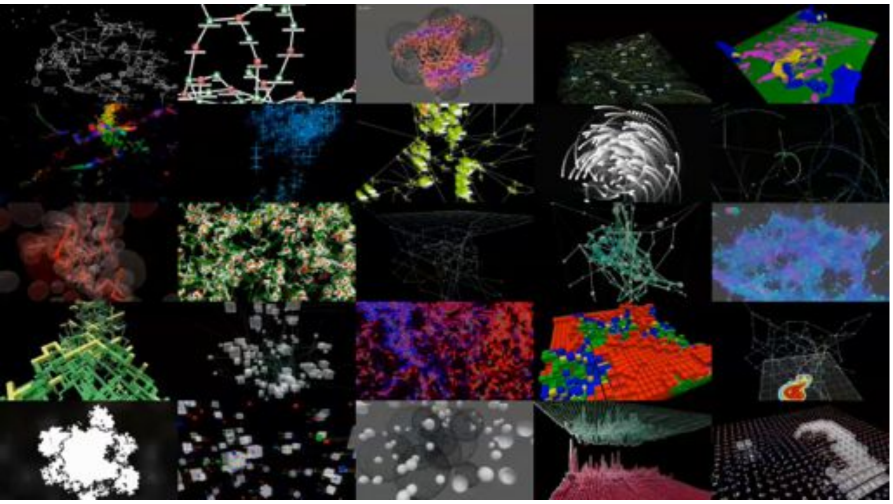
Columbia University GSAPP
Visual Studies Sequence A
Instructors Toru Hasegawa (th2114) & Mark Collins (mlc2115)

Search: Advanced Algorithmic Design



Stereographic (3D) point cloud of Low Library, Search:AAD Spring 2010

Design computation encompasses a broad range of approaches and techniques. This course examines fundamental concepts in computing, including recursion, procedural geometry, design space exploration, agent-based and physics-based simulation through simple visual code examples - designed for visual thinkers. Using object oriented programming and Processing as a starting point, the seminar explores the practice and promise of these new means of design conception through a process of experiment, play and rewiring. These workshops support a focused research led by the student in the exploration of an emerging topic or technology in design computing. The results of the class are visualizations that reveal the hidden algorithmic logic and generative potential at work inside each code.



Mosaic of Search:AAD student work.

Course Goals and Methodology

This workshop will explore generative design methodologies through the application of algorithmic techniques - we will be looking at fundamental coding principles (recursion, feedback, modularity and I/O) while working within an object-oriented framework, opening the door to complex simulation and animate formation. Artificial life, material intelligence, interactivity, and other second-order principles will be approached from the vantage point of “dynamics” and “search” – or the introduction of directed intelligence into a dynamic process of making.

Development - A process in which something passes by degrees to a different stage.

Behavior - The aggregate of responses to internal and external stimuli.

Behavior and development are understood to be a sum, or aggregate, of a multitude of innocuous decisions. Each is a dynamic, or a process ‘in time’ that necessarily feeds-back and self-regulates to promote higher levels of form and organization. Students will develop a focused inquiry into a specific area of algorithmic dynamics. Here, “dynamics” is meant as a inclusive term for all kinds of activity: formal development, flocking, embryology, automata, FEA, fractals and l-systems are all examples of time-based recursive practices. The class is meant to flesh out a vocabulary and structural understanding of a wide array of algorithms, to look for correspondences among dynamics, mapping and search heuristics. By casting a wide net, we hope to see opportunities for portability and the development of a critical stance towards algorithmic 'tooling.'

Processing

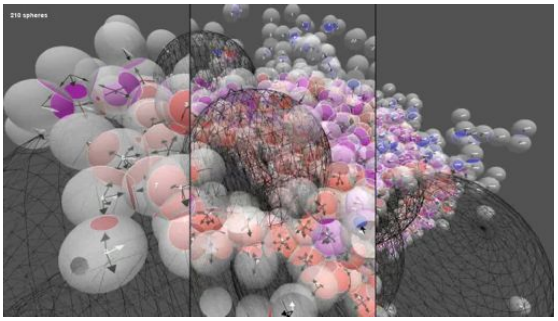
Search goes beyond "scripting" – using limited algorithmic structures in conjunction with other software packages – towards the authorship of completely new, custom software. Why do we need to open up the black box of computation? Using ready-made software, even extended with scripting, we remain embedded in the nested universe of the software. A project, especially a building, is its own universe of variables, materials, means and agency – it must go beyond, it must construct its own universe. Object oriented programming gives us a rich, flexible language to specify these new spaces, to specify a true animate ecology. In constructing our own software we will draw upon a fundamental understanding of computational language, a primary and necessary "literacy" for innovation.

Object Oriented Programming

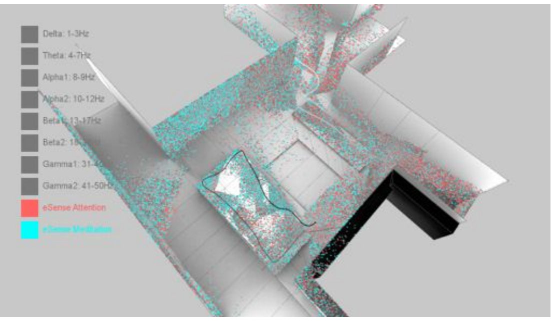
Object-oriented programming (OOP) is a crucial part of the seminar's approach to algorithms. Modularity is key to moving beyond simple scripting, which necessarily focus purely on geometry, towards a behavioral architecture. We wish to provoke architecture into a robust dynamism, to look for correspondences between formal and spatial articulation, environmental factors and other mediums of agency. To achieve this, we must exploit platforms such as Processing that can support spatial research at a speed, intensity and multiplicity beyond that available in the scripting languages of Maya or Rhino.

The workshop will consist of two short projects: a generative procedure and a system of valuation. As a last step, we will be using meshing as a means to translate various types of information into topologically complex form. This workshop is an exploration of algorithmic design methodologies rather than a scripting tutorial and although some prior scripting experience will be useful, NO scripting experience is required. We will be interested in the economic deployment of short, modular pieces of code that become inter-operable and give rise to complex structures and novel behaviors.

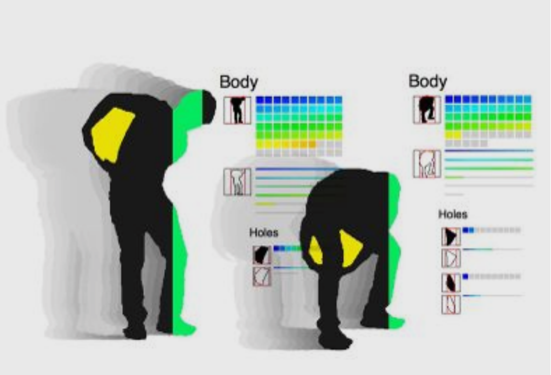
Recrusive geometry at work in nature. Recursion is created through self-reference, either in geometry or transformation.



Physics based modeling in Processing.



Mapping brainwave data onto a BIM model.



Processing visualization of body analytics using the OpenCV library.