62-706 Generative Systems for Design

Spring Semester 2021 • 12 units • Tuesday + Thursday 3.00–4:20pm (Remote)

Instructors Ramesh Krishnamurti • ramesh@cmu.edu
Pedro Veloso • pveloso@andrew.cmu.edu

Office Hours TBD

SYLLABUS

COURSE DESCRIPTION

This course is for visual artists, enthusiasts, and designers from different areas (game, product, architecture, building performance, etc.). Taking a stance, say of, instead of working directly with analogue media, such as drawings, designers can develop algorithms and computational models that can generate design alternatives, based on custom input. We refer to such algorithms and models as **generative systems**.

Generative systems have been an important topic of research in the recent decades, in books, courses, articles and research conferences in computational design and in other disciplines. Earlier approaches were based on classical artificial intelligence and optimization methods; recently, a variety of computational techniques from different fields, such as parametric modeling, agent-based modeling or neural networks, have been incorporated in the development of new generative systems. With recent developments in machine learning we can even develop models that learn automatically from data or experience. In this course we focus on the basic or classical techniques, although with an acknowledgement to recent developments, to briefly introduce learning techniques.

The main goal of this course is to foster the student's capacity to formulate design problems computationally, with emphasis on the synthesis of design alternatives. This course provides an overview of the main topics in Generative Systems, with historical notes and technical specifications. Along the semester, the students will address different design problems with different generative techniques. The course will address topics such as variational modeling, rule-based modeling, directed and dynamic simulation, optimization and learning. The appropriate data structures, algorithms and models will be discussed, and some implemented in the exercises and projects.

LEARNING OUTCOMES

In this course students will:

- Apply elementary algorithmic thinking to design problems
- Students will be able to formulate a design problem computationally and develop an appropriate generative system to synthesize solutions
- For designers, it is an opportunity to understand the potential of generative systems not only for the automation of repetitive design tasks but also (and mainly) for the exploration of innovative design solutions

- For non-designers, it is an opportunity to understand how computational logic can lead to creative applications outside of the regular demands of the industry or the consolidated problems in science

The course consists of lectures, computer instruction and assignments.

REFERENCES

There is no textbook. Specific references **will be provided** for each assignment; however, the following list is a general overview:

- Adamatzky, Andrew, and Genaro J. Martinez, eds. 2016. Designing Beauty: The Art of Cellular Automata. Emergence, Complexity and Computation. Springer.
- Akin, Ömer, and Ran Sen. 1996. "Navigation within a Structured Search Space." Environment and Planning B: Planning and Design 23: 421–42.
- Akos, Gil, Ronnie Parsons, Sharon Jamison, Andrew Reitz, Armon Jahanshahi, Luis Quinones, Erick Katzenstein, Kimberly Parsons, and Roberto Godinez. 2014. The Grasshopper Primer. 3rd ed. Mode Lab. http://grasshopperprimer.com/en/index.html?index.html.
- Bishop, Christopher M. 2006. Pattern Recognition and Machine Learning. New York: Springer-Verlag.
- Bohnacker, Hartmut, Benedikt Gross, Julia Laub, and Claudius Lazzeroni. 2012. Generative Design: Visualize, Program, and Create with Processing. Princeton Architectural Press.
- Boyd, Stephen, and Lieven Vandenberghe. 2004. Convex Optimization. Cambridge university press.
- Brownlee, Jason. 2012. Clever Algorithms: Nature-Inspired Programming REcipes. lulu.com. http://www.cleveralgorithms.com/nature-inspired/index.html.
- Coates, Paul. 2010. Programming Architecture. London: Routledge.
- Coates, Paul, and Robert Thum. 1995. Generative Modelling. University of East London. http://roar.uel.ac.uk/948/.
- Davis, Daniel. 2013. "Modelled on Software Engineering: Flexible Parametric Models in the Practice of Architecture."
- Downey, Allen B. 2012. Think Complexity. Sebastopol: O'Reilly Media. http://thinkcomplex.com.
- Fischer, Thomas, and Christiane M. Herr. 2001. "Teaching Generative Design." In Proceedings of the 4th Conference on Generative Art. Politechnico di Milano University.
- Floreano, Dario, and Claudio Mattiussi. 2008. Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies. The MIT Press.

- Foster, David. 2019. Generative Deep Learning: Teaching Machines to Paint, Write, Compose and Play. Sebastopol: O'Reilly Media.
- Frazer, John. 1995. An Evolutionary Architecture. Architectural Association. London.
- Goodfellow, Ian, Bengio Yoshua, and Courville Aaron. 2016. Introduction to Deep Learning. MIT Press. http://www.deeplearningbook.org.
- Goodrich, Michael T., Roberto Tamassia, and Michael H. Goldwasser. 2013. Data Structures and Algorithms in Python. John Wiley & Sons Ltd.
- Herr, C. 2002. "Generative Architectural Design and Complexity Theory." In International Conference on Generative Art.
- Holland, John H. 1998. Emergence: From Chaos to Order. New York: Basic Books.
- Kennedy, James, Russel C. Eberhart, and Yuhui Shi. 2001. Swarm Intelligence. San Francisco: Morgan Kaufmann Publishers.
- Khabazi, Zubin. 2012. Generative Algorithms Using Grasshopper. 3rd ed. Morphogenesism. https://issuu.com/pabloherrera/docs/generative_algorithms_f07be9052a219d.
- Knight, Terry W. 1991. "Designing with Grammars." In Computer Aided Architectural Design Futures: Education, Research, Applications [CAAD Futures '91 Conference Proceedings / ISBN 3-528-08821-4] Zürich (Switzerland), July 1991, Pp. 33-48. CUMINCAD. http://papers.cumincad.org/cgi-bin/works/Show?2559.
- Mandelbrot, Benoit B. 1982. The Fractal Geometry of Nature. Vol. 1. WH freeman New York.
- Mitchell, Tom M. 1997. Machine Learning. Boston: McGraw-Hill.
- Mitchell, William J. 1977. Computer-Aided Architectural Design. New York: Mason Charter Pub.
- ———. 1990. The Logic of Architecture: Design, Computation, and Cognition. Cambridge: The MIT Press.
- Pearson, Matt. 2011. Generative Art: A Practical Guide Using Processing. Manning Publications.
- Prusinkiewicz, Przemyslaw, and Aristid Lindenmayer. 2004. The Algorithmic Beauty of Plants. New York: Springer-Verlag.
- Reas, Casey, and Chandler McWilliams. 2010. Form+Code in Design, Art, and Architecture. Princeton Architectural Press.
- Reynolds, Craig W. 1999. "Steering Behaviors for Autonomous Characters." In Game Developers Conference, 1999:763–782.
- Russel, Stuart J., and Peter Norvig. 2010. Artificial Intelligence: A Modern Approach. 3rd ed. Upper Saddle River: Prentice Hall.
- Shiffman, Daniel. 2012. The Nature of Code. https://natureofcode.com/book/.
- Simon, Herbert A. 1996. The Sciences of the Artificial. 3rd ed. Cambridge: The MIT Press.

- Stiny, George. 2008. Shape: Talking about Seeing and Doing. Cambridge: The MIT Press.
- Tedeschi, Arturo, and Stefano Andreani. 2014. AAD, Algorithms-Aided Design: Parametric Strategies Using Grasshopper. Brienza: Le Penseur Publisher.
- Terzidis, Kostas. 2006. Algorithmic Architecture. Oxford: The Architectural Press.
- ———. 2014. Permutation Design: Buildings, Texts, and Contexts. New York: Routledge.
- Tibbits, Skylar, Arthur Van Der Harten, and Steve Baer. 2011. Python for Rhinoceros 5.
- Wilensky, Uri, and William Rand. 2015. An Introduction to Agent-Based Modeling: Modeling Natural, Societal and Engineered Complex Systems with Netlogo. Cambridge: The MIT Press.
- Wolfram, Stephen. 2002. A New Kind of Science. Champaign: Wolfram Media.
- Woodbury, Robert. 2010. Elements of Parametric Design. New York: Routledge.
- Wortmann, Thomas. 2018. "Efficient, Visual, and Interactive Architectural Design Optimization with Model-Based Methods." PhD Dissertation, Singapore: Singapore University of Technology and Design.

CANVAS

Course material will be on Canvas. All subsequent references to 'Canvas' refer to the course Canvas website.

Canvas has a section for Discussion which will be used for Q&A sessions. Students can post questions about concepts and assignments. Other students can reply to the post to share their experience or ideas and logic about a problem. However, **uploading any file which includes code is prohibited**.

Class will be **conducted entirely remotely** through Zoom. The meeting link will be announced in Canvas.
Links for each class in this course are provided in the Syllabus section on Canvas.
'Jump to Today' tab in the section will be helpful in finding the class link.
Please refer to the section below on Using Zoom.

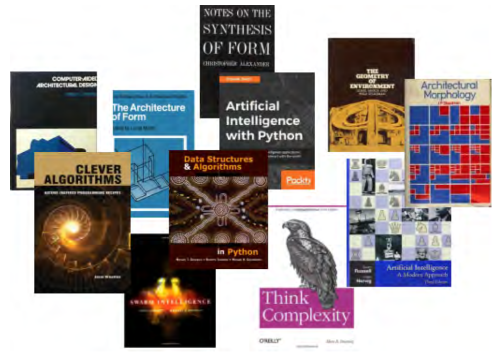
ASSIGNMENTS

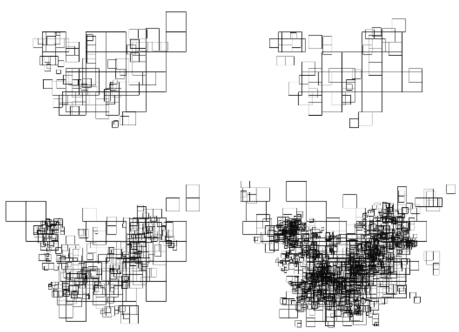
The course grade will be based on **five homework assignments**, which includes **a small project**. Each assignment contains questions about the formulation of design problems in different generative schemas and exercises to implement them as small projects. The students will choose one of the projects and extend it for an exhibition, with prototypes and interactive displays. Some parts of the assignments and project may be done by groups of two. Groups will not be permitted to repeat between assignments or project.

Each assignment has at most four parts tentatively titled:

theory, exercises, challenge, analysis

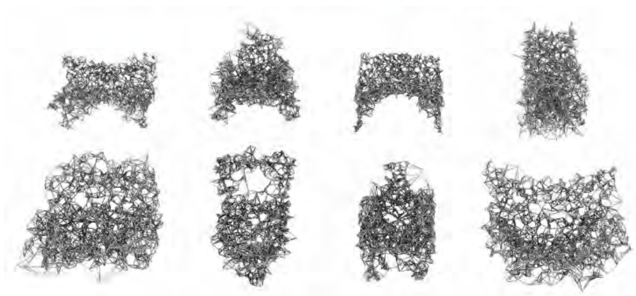
The **theory** part contains questions on readings related to specific topics and may include general references to data structures, algorithms, models and application. These questions may refer to excerpts from books, papers or websites.

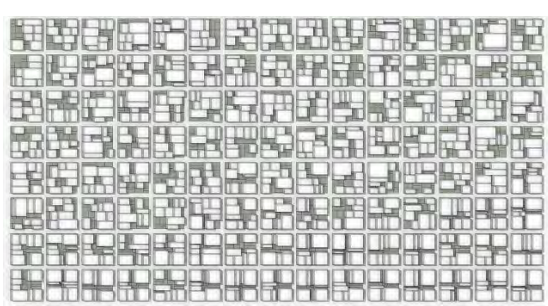




The **exercises** part refers to implementations that are started in class to be completed as part of the assignment. Each exercise may consist of multiple tasks. The student will select a subset of these to implement. The breakdown of the grade for the exercises will be specified in the assignment.

The **challenge** part reflects the student-selected mini-project to which they apply the principles learnt. In the general these are set at level slightly more complex than the class exercises.





The **analysis** essentially visualizes the design or solution space of the challenge.

Students will receive feedback on each assignment.

GRADING

The **final grade** is determined by a weighted **average** of assignment 4 and the other three assignments. Assignment 4 includes the required project component.

Grades are based on the following scale with a finer ± refinement:

A: 90% and over **B**: 80-89% **C**: 70-79% **D**: 60-69% **R**: < 60%

Students are not graded on a curve however we will consider the degree of difficulty experienced in the assignments in determining the final scores.

PREREQUISITE

Familiarity with basic Python programming and Grasshopper. Students are expected to have Rhino3D with Grasshopper installed on their own laptops.

POLICIES

All university academic and student policies as set out in http://www.cmu.edu/graduate/policies/ and https://www.cmu.edu/policies/student-and-student-life/index.html apply to this course.

Specifically:

- You are expected to be on time at all lecture and lab sessions.
- Please backup your work in the cloud. We cannot accept hardware failure as a valid excuse.
- You may not copy code without citation. Copying code without citation is plagiarism.
- Late work may result in a reduced grade.
- Email should only be used for crucial queries and concerns. Please direct software related questions to Pedro during office/lab sessions.

REMOTE INSTRUCTION

This semester involves regular use of technology during class. Research has shown that divided attention is detrimental to learning, so we encourage you to close any windows not directly related to what we are doing while you are in class. Please turn off your phone notifications and limit other likely sources of technology disruption, so you can fully engage with the material, each other, and me. This will create a better learning environment for everyone.

IN PERSON INTERACTIONS: Although this semester, this is a remotely conducted class and there may be occasions which require face-to-face interactions, in which case all university recommendations on face-to-face contact must and will be observed.
See https://www.cmu.edu/coronavirus/students/tartans-responsibility.html.

USE OF ZOOM IN THE CLASS (INCLUDING USE OF VIDEO)

In our class, we will be using Zoom for synchronous (same time) sessions. See the Zoom link on Canvas.

Please make sure that your Internet connection and equipment are set up to use Zoom and able to share audio and video during class meetings. (See https://www.cmu.edu/computing/start/students.html for information on the technology you are likely to need.) Let me know, as early as possible, if there is an issue with your technology set-up to sort it out.

SHARING VIDEO: In this course, being able to see one another helps to facilitate a better learning environment and promote more engaging discussions. Our default will be to expect students **to have their cameras on** during lectures and discussions. However, we also completely understand that there may be reasons students would not want to have their cameras on. If you have any concerns about sharing your video, please email us as soon as possible and we can discuss possible adjustments.
Note: You may use a background image (preferably static image) in your video if you wish; just check in advance that this works with your device(s) and internet bandwidth.

- During our class meetings, **please keep your mic muted** unless you are sharing with the class or your breakout group.
- If you have a question or want to answer a question, please use the chat or the "raise hand" feature (available when the participant list is pulled up). We will monitor these channels in order to call on students to contribute.
- Our synchronous meetings may involve breakout room discussions, and those will work better if everyone in your small group has their camera turned on. During large group debriefs, you may keep your video off.

RECORDING OF CLASS SESSIONS

All synchronous classes will be recorded via Zoom so that students in this course (and only students in this course) can watch or re-watch past class sessions. Please note that breakout rooms will not be recorded. However, chats are recorded. We do not encourage private chats during recorded Zoom sessions, instead we recommend that you send us emails. We will make the recordings available on Canvas as soon as reasonably possible after each class session. Recordings will live on Canvas. **Please note that you are not allowed to share these recordings**. This is to protect your FERPA rights and those of your fellow students.

RESPECT FOR DIVERSITY

Although open to advanced undergraduate students in architecture and other departments, this is primarily a graduate course in Computational Design in the School of Architecture, and as such I have always had students from many diverse backgrounds. It is my intent that all students irrespective of background or perspective continue to be well served by me, that students' learning needs are addressed both in and out of class, and that the diversity that you bring to this class be viewed as a resource, strength and benefit. It is my intent to present materials and activities that are respectful of diversity in all its forms. Your suggestions are encouraged and appreciated. Please let me know ways to improve the effectiveness of the course for you personally or for other students or student groups. In addition, should any of our class meetings conflict with religious events, please let me know so that we can make suitable alternate arrangements for you.

ACCOMMODATION FOR STUDENTS WITH DISABILITY

If you have a disability and are registered with the Office of Disability Resources, I encourage you to use their online system to notify me of your accommodations and discuss your needs with me as early in the semester as possible. I will work with you to ensure that accommodations are provided as appropriate. If you suspect that you may have a disability and would benefit from accommodations but are not yet registered with the Office of Disability Resources, I encourage you to contact them at access@andrew.cmu.edu.

STUDENT WELL-BEING AND SUPPORT

Carnegie Mellon University is deeply committed to creating a healthy and safe campus community including one that is free from all forms of sexual and relationship violence. To that end, University Health Services, the Office of Community Standards & Integrity, and the Office of Title IX Initiatives have partnered to expand their educational efforts for graduate students in this domain. There is an educational opportunity for all graduate students at Carnegie Mellon that reflects its commitment to sexual assault and relationship violence prevention as well as to your overall safety:

It is important to take care of yourself. Do your best to maintain a healthy lifestyle this semester by eating well, exercising, avoiding drugs and alcohol, getting enough sleep and taking some time to relax. This will help you achieve your goals and cope with stress.

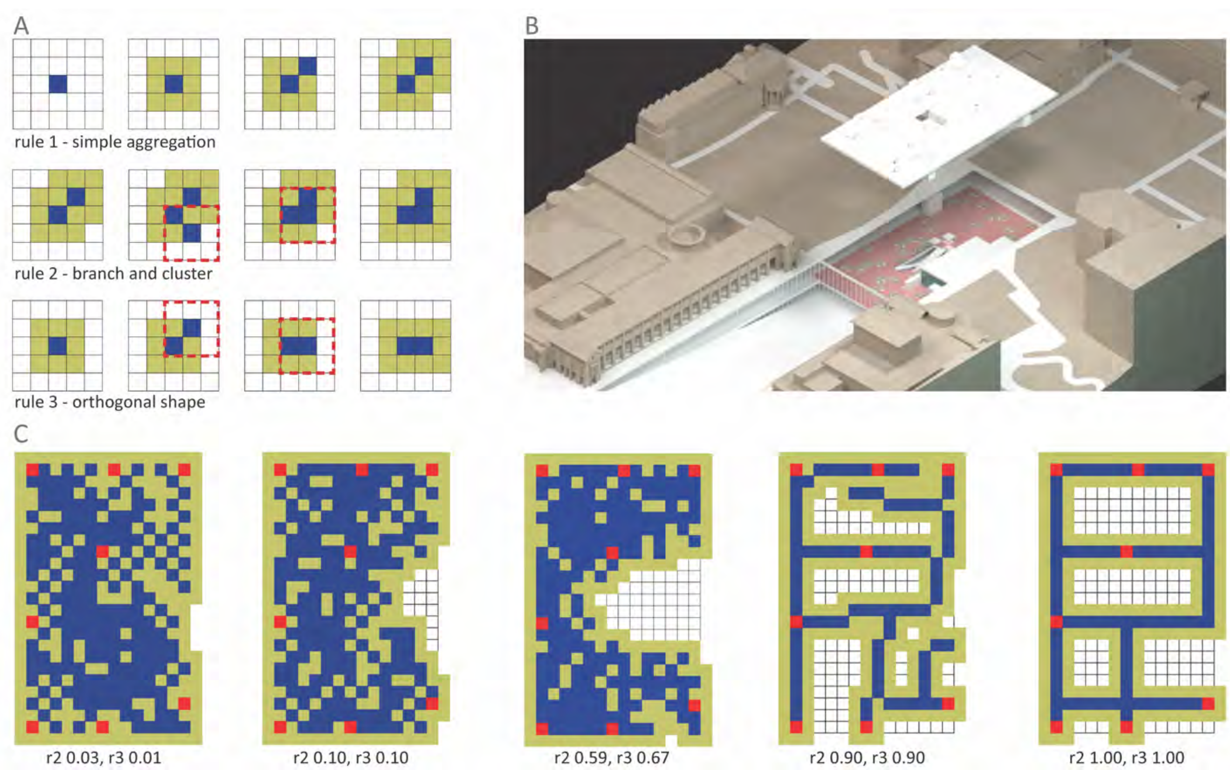
All of us benefit from support during times of struggle. There are many helpful resources available to all students on campus. Asking for support sooner rather than later is more often helpful. If you or anyone you know is experiencing academic stress, difficult life events, or feelings like anxiety or depression, we strongly encourage you to seek support. Counseling and Psychological Service (CaPS) is here to help: please call 412-268-2922 or visit their website at http://www.cmu.edu/counseling/. Consider reaching out to a friend, faculty or family member you trust for help getting connected to the support that can help.

If you or someone you know is feeling suicidal or in danger of self-harm, call immediately, day or night:
CaPS: 412-268-2922 **Re:solve Crisis Network**: 888-796-8226

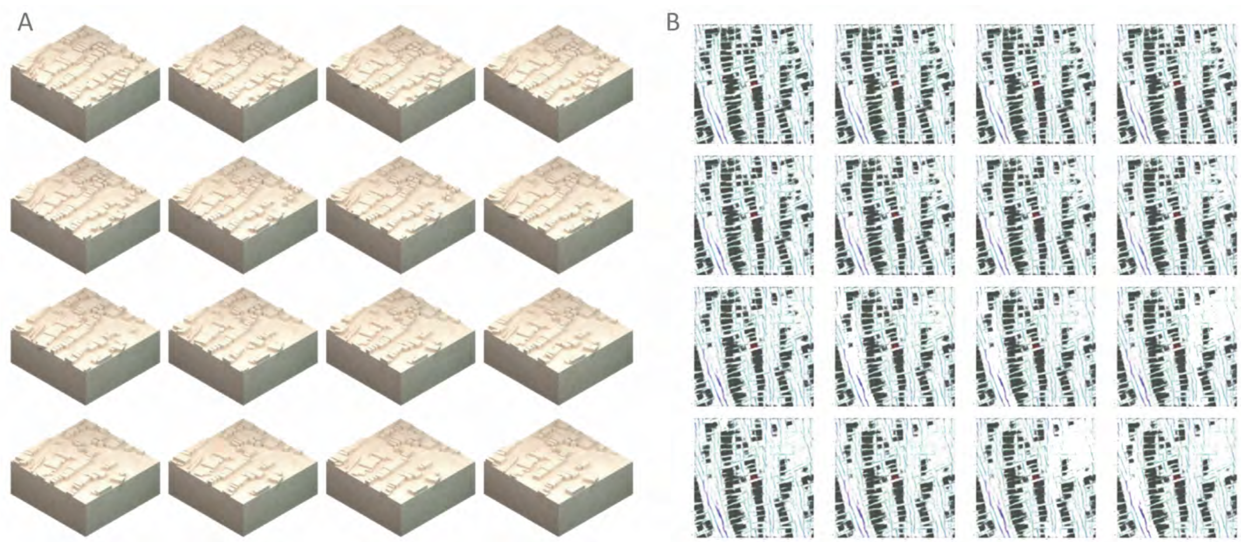
If the situation is life threatening, call the police:
On campus: CMU Police: 412-268-2323 **Off campus**: 911

EXAMPLES OF STUDENT WORK FROM PREVIOUS YEARS

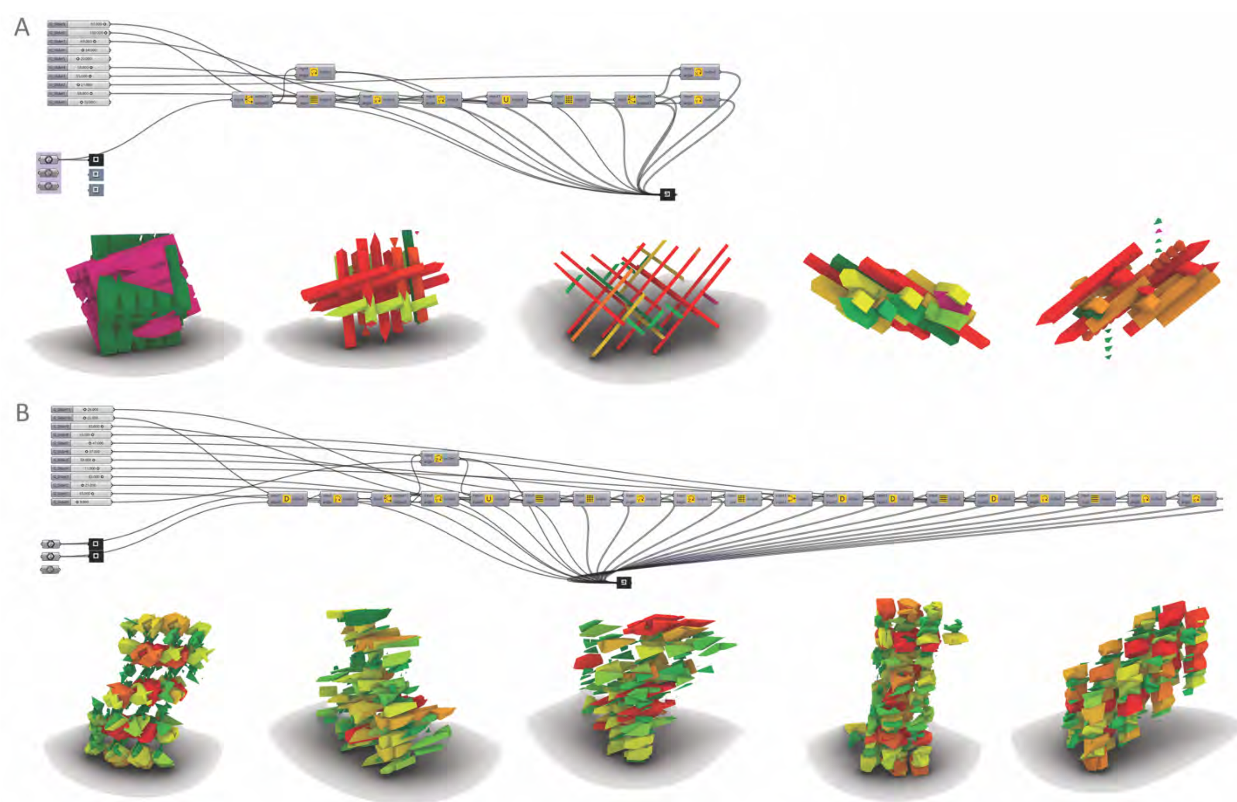
Autoilet (Jichen Wang and Joon Young Choi)
A: parameter space of the input (boundary, corner and door placement) for rectangular and L-shaped footprints. B: examples of solutions using Depth First Search (DFS) for placement of toilet, sink and ADA circle.



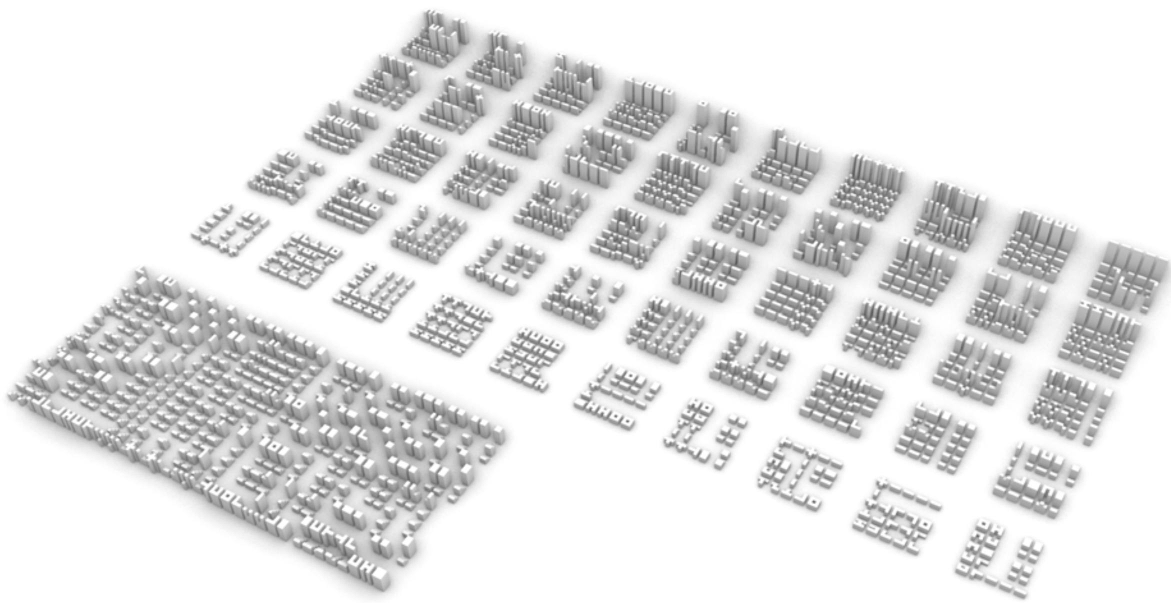
Agents generating an underground gallery on campus (Jinmo Rhee).
A: stochastic program with three rules. B: design of campus gallery based on agents with r2 = 0.59 and r3 = 0.67 at time step 200. C: the resulting spatial configuration for the same initial environment with different parameter: r2 and r3 increase the probability of selecting rule 2 and rule 3 respectively.



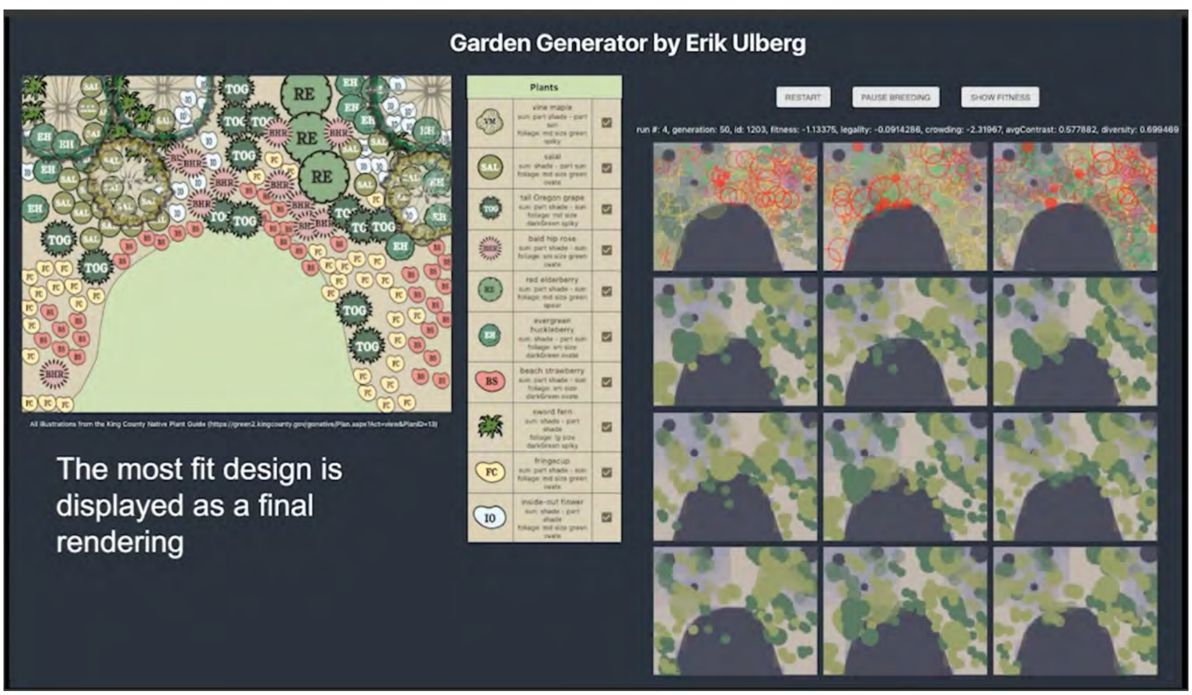
Urban Structure Synthesizer (Jinmo Rhee)
3D models (A) and 2D diagrams (B) with incremental variations of an urban fabric. The diagrams were generated with a model (WGAN-GP) trained on a dataset of images from Pittsburgh, USA.



Use of genetic programming (plug-in **Embryo**) (Michael Stesney)
Generating parametric definitions for sculptures. A-B: two parametric graphs with custom operations and variations of the resulting sculpture below.



Alphabet City (Jichen Wang and Aprameya Pandit)
In his book *The Alphabetical City*, Steven Holl suggests that buildings in a continuous urban space are like letters in words or sentences, such as T, I, U, O, H, E, B, L, X.



Garden Generator (Erik Ulberg)
The generator combines a genetic algorithm with physics simulator. The genotype is used to generate the initial design (placement of plants), the physics is used to solve the collisions (plants, boundaries and sun levels) and make the design viable (phenotype). A Weighted Delaunay triangulation is used to calculate contrast between adjacent plants.

COURSE SCHEDULE*

* Schedule subject to changes

Topic	Lecture & Lab topic(s) + problem(s)	Exercises
0 Introduction to course 02/02/21	• course structure • canvas • assignment structure • what are generative systems?	
1 Variational modeling 02/04/21 02/09/21 02/11/21	• formulating a design problem as variational model • parametric modeling and scripting • iteration and recursion **Lab –** GHPython – loops, conditionals, and geometry	**Hw1** **release** 02/04/21 **due** 03/02/21
2 Rule-based modeling 02/16/21 02/18/21 02/23/21 02/25/21	• formulating a design problem as a rule-based model • fractals • L-systems • Shape Grammars **Lab –** slice tree, fractals, L-systems, and simple shape grammars	
3 Searching for solutions 03/02/21 03/04/21 03/09/21 03/11/21	• automating decision making in design • trees and graphs as design spaces and navigational structures • uninformed and informed search strategies • constrained search • applications **Lab –** flood fill, tiling, and automatic shape grammar	**Hw2** **release** 03/02/21 **due** 03/30/21

4 Cellular models and agents 03/16/21 03/18/21 03/23/21 03/25/21	• using cellular models in design • grid-based simulation (cellular automata, reaction-diffusion and diffusion-limited aggregation) • cellular models for design • pheromone-based models **Lab –** cellular automata, reaction-diffusion, and beady rings	
5 Agent-based models 03/30/21 04/01/21 04/06/21	• the idea of agents in design • particles or agents? • growing agents • swarm behavior **Lab –** 2d-growth and Boids	**Hw3** **release** 03/30/21 **due** 04/22/21
6 Optimization 04/08/21 04/13/21 04/15/21 04/20/21	• formulating design as an optimization problem • basic optimization strategies (generate and test and hill-climbing) • metaheuristics: genetic algorithm and genetic programming • sources of evaluation (structural, environmental, spatial, etc.) • applications **Lab –** hill-climbing, genetic algorithm & genetic programming	
7 Final Project 04/22/21	**Checkpoint:** select, discuss, and propose final projects	**Hw4** **release** 04/22/21 **due** TBD
7 Neural networks 04/27/21 04/29/21	• generative systems based on learning • machine learning tasks • neural network and deep learning • autoencoders • generative adversarial networks • applications **Lab –** machine learning resources for designers	
8 Final Project 05/04/21 05/06/21	**Checkpoint:** prototyping and curation for the exhibition (work session – no lectures till end of semester)	**Exhibition** (TBD)