Arch 7326: Developing Computational Solutions for Design Problems

Instructors:

Mostafa Akbari, Ph.D. Candidate, akbariae@design.upenn.edu

Yao Lu, Ph.D. Candidate, yaolu61@design.upenn.edu

Location: *MEYH B7(8/29, 12/11)*

Meeting time: *Thursday 5:15 – 8:14 PM*

Course Description

Developing Computational Solutions for Design Problems is a seminar aimed at M.Arch students who seek to identify, investigate, formulate, and resolve design problems using advanced computational techniques. This course goes through the critical knowledge and technical foundations that current computational design and modeling tools are built upon, then enables and encourages the students to rethink, reinvestigate, and resolve their design problems from a computational and algorithmic perspective.

Computational design in Architecture is an interdisciplinary field that is deeply rooted in science, technology, and mathematics. Understanding the foundation in these fields is necessary for designers to freely exploit the power of the emerging technologies and unleash the creativity in dealing with design problems. Although there are massive bodies of knowledge in the related fields, there is only a small subset of concepts and knowledge that is pertinent to the work of creative designers and researchers. This subset of essential knowledge is carefully curated and introduced through the seminar. More importantly, this seminar also initiates the students to rethinking and resolving problems encountered in design process from the perspective of computation. Students are encouraged to develop and share their own computational tools, which could have broader benefits for the entire design community.

Course Objectives

- Introduce critical knowledge of computational geometry and computer graphics.
- Introduce programming and tool developing techniques within Rhino environment.
- Participate in hands-on workshops to master the foundational knowledge and techniques.

- Encourage students to identify and rethink design problems related to their own experience and produce a computational tool that aims to resolve this problem.

Seminar Structure

The course will be conducted as a mix of lectures, hands-on workshops, and discussions. Assignments will be a blend of technical exercises and projects, culminating in a final project of the students' choice at the end of the semester. Students will be working individually on the assignments, and in groups of three for the final project.

Deliverables

- A report of the identified design problem and the explanation of the proposed computational solution.
- A working script/tool that can be shared.
- A visual demonstration (a video) of how this script/tool help solve the identified problem.

Detailed Weekly Calendar

Week 1 (August 31st)

Lecture I: Introduction to computational design (Mostafa Akbari)

Lecture II: Introduction to design computation (Yao Lu)

Week 2 (September 7th)

Lecture III: Mathematics behind the computational geometries (Yao Lu)

- point, vector, line, plane
- vector operations: addition, subtraction, cross product, dot product
- NURBS geometries: curve, surface, brep
- Mesh
- Transformation

Grasshopper workshop I (Mostafa Akbari)

- Parametric model

Week 3 (September 14th)

Lecture IV: A deeper look into Grasshopper (Yao Lu)

- Understand how components run, data mapping, wire notation

Grasshopper workshop II (Mostafa Akbari)

- Panelization

Week 4 (September 21st)

Grasshopper workshop III (Mostafa Akbari)

- Folding

Grasshopper workshop IV (Yao Lu)

- Fologram

Week 5 (September 28th)

Grasshopper workshop V (Mostafa Akbari)

- Topology Optimization

Grasshopper workshop VI (Yao Lu)

- Gcode Generation – 3D printing

Week 6 (October 5th)

Python lecture I (Yao Lu):

- Introduction to Python
- Scripting environment and Python syntax
- Values, keywords, types, operations, variables, assignment
- Basic function calls,
- Rhinoscriptsyntax and documentation
- Errors and debugging

Python workshop I (Mostafa Akbari)

Project module I: Final project introduction and mid-term task release.

Week 7 (October 12th)

No class

Week 8 (October 19th)

Python lecture II (Yao Lu):

- For loop

- Custom functions
- Modules and import

Python workshop II (Mostafa Akbari)

Week 9 (October 26th)

Python lecture III (Yao Lu):

- Conditionals
- While loop
- Built-in data structure: list, dictionary, set

Python workshop III (Mostafa Akbari)

Project module II: mid-term presentation.

Week 10 (November 2nd)

Python lecture IV (Yao Lu):

- Python in Grasshopper
- Use grasshopper components as functions in Python

Python workshop IV (Mostafa Akbari)

Week 11 (November 9th)

Python lecture V (Yao Lu):

- Create plug-ins for Rhino and Grasshopper

Python workshop V (Mostafa Akbari)

- Advanced coding examples

Week 12 (November 16th)

Project module III

- Desk crits and troubleshooting

Week 13 (November 21st Tuesday follows the Thursday schedule)

Project module IV

- Desk crits and troubleshooting

Week 14 (November 30th)

<u>Project module V</u>

- Desk crits and troubleshooting

Week 15 (December 7th)

<u>Project module VI</u>

- Desk crits and troubleshooting

Student Evaluation

Attendance and Participation 10%

Seminar Presentations (midterm) 20%

Assignments and exercises 30%

Final Project 40%

Attendance Policy

Attendance at all seminars is compulsory. Students are expected to attend all classes for the entire scheduled meeting time and are responsible for completing assignments and for knowing the material covered in class. Students are allowed one absence without a final course grade reduction for all seminar courses. After the allowed absence, a student's final course grade will be reduced one-half level for each additional absence (e.g. after the second absence from a seminar the final course grade will be lowered from a B+ to a B, after the third absence from a B+ to B-, etc.).

Code of Academic Integrity

All students are responsible for upholding the Code of Academic Integrity as published in the University of Pennsylvania Penn BOOK - https://catalog.upenn.edu/pennbook/. Students are expected to undertake independent and individual research and adopt all expected rules of citation. Please ensure you consult the Code of Student conduct, https://catalog.upenn.edu/pennbook/code-of-student-conduct/, and the Code of Academic Integrity https://catalog.upenn.edu/pennbook/code-of-academic-integrity/, for all pertinent details.

Reading Materials

- Downey, Allen. Think python. " O'Reilly Media, Inc.", 2012.
- Petzold, Charles. 2022. Code: The Hidden Language of Computer Hardware and Software. 2nd edition. Hoboken: Microsoft Press.

- Marschner, Steve, and Peter Shirley. 2021. Fundamentals of Computer Graphics. 5th edition. Boca Raton: A K Peters/CRC Press.
- Skyler Tybbits, Arthur Van der Harten, Steve Baer. 2011. Python for RhinoCeros. CreateSpace Independent Publishing Platform.
- Parent, Rick. Computer animation: algorithms and techniques. Newnes, 2012.
- Lengyel, Eric. "Mathematics for 3D Game Programming & Computer Graphics (Game Development Series)." (2002).
- CELANI, G. (2002) Beyond Analysis and Representation in CAD: a New Computational Approach to Design Education, unpublished Ph.D. thesis submitted to Department of Architecture, MIT.