Introduction to Computational Design

Yuki Koyama
koyama.y@aist.go.jp
National Institute of Advanced Industrial Science and Technology (AIST)
Tsukuba, Ibaraki, Japan

ABSTRACT

Computational design is one of the hot topics in HCI and related research fields, where various design problems are formulated using mathematical languages and solved by computational techniques. By this paradigm, researchers aim at establishing highly sophisticated or efficient design processes that otherwise cannot be achieved. Target domains include graphics, personal fabrication, user interface, etc. This course introduces fundamental concepts in computational design and provides an overview of the recent trend. It then goes into a more specific case where human assessment is necessary to evaluate the quality of design outcomes, which is often true in HCI scenarios. This course is recommended to HCI students and researchers who are new to this topic.

CCS CONCEPTS

• **Human-centered computing** → **Human computer interaction (HCI)**.

KEYWORDS

Computational design, optimization, human-in-the-loop, crowdsourcing, machine learning

ACM Reference Format:
Yuki Koyama. 2021. Introduction to Computational Design. In *CHI Conference on Human Factors in Computing Systems Extended Abstracts (CHI '21 Extended Abstracts), May 8–13, 2021, Yokohama, Japan.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3411763.3445007

1 INTRODUCTION

1.1 What is Computational Design?

1.1.1 Definition. Computational techniques, such as optimization and machine learning, have recently played an increasingly important role in enhancing human-computer interaction (HCI). *Computational design* is one of the emerging hot topics in this context. Since there has seemed no established definition yet [2], we define it as follows and will use it in this course: *Computational design is a paradigm in which design problems are formulated mathematically and solved by computational techniques.* The formulated design problem takes the form of *optimization* in most cases [2], and the optimization is performed as the process of searching for the best design among some options either in an automatic way by running optimization algorithms, in a manual way with computationally enabled support, or in a hybrid way using computational systems with having a human in the loop [4]. The primary goal of computational design research is to establish sophisticated or efficient design processes that otherwise cannot be achieved [4]. This topic has also been actively studied in related fields, such as computer graphics. Application domains include visual design [7, 8], personal fabrication [12, 15], user interface [11], interactive device [1, 3], robotics [9], *etc.*

1.1.2 Design as Optimization. Typical optimization problems can be described using the mathematical language in the form of

$$\mathbf{x}^* = \arg\max_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x}). \tag{1}$$

The message of this equation is simple: *the variable set* $\mathbf{x}^*$ *is the maximizer of the function* f *within the candidate space* $\mathcal{X}$. See Figure 1 for illustration. This equation involves the following concepts, each of which can be interpreted as a concept in design:

- **Search variable set x:** This concept is about *design parameter set* in the design context and determines which parameters will be computationally manipulated in the optimization process. It can consist of either a single parameter or multiple parameters. For example, it can include font size, font color, background color, *etc.*, in web design scenarios. These parameters can be either discrete (*e.g.*, font) or continuous (*e.g.*, font size).
- **Search space $\mathcal{X}$:** This concept is about *design space* in the design context and represents the set of all the possible design choices. If the search variables are discrete, then the search space is a list of candidates (*e.g.*, $\mathcal{X}$ = {"Times", "Helvetica"} if $\mathbf{x}$ consists of a font parameter). If the search variables are continuous, then the search space can also be continuous.
- **Objective function f:** This concept is about *design goal* and determines the criterion based on which the design artifact is evaluated. In other others, this function quantifies how good a given choice is.
- **Optimal solution $\mathbf{x}^*$:** This concept represents the variable set that provides the best possible design. Here, "best" means that the value of the objective function is maximized (or minimized, depending on the formulation). This is found as the outcome of the optimization process.

We can describe many design problems using these concepts. This course will review representative works from various design domains and explain how researchers have formulated actual design problems using these concepts.

Designing an appropriate objective function is the key to success and often requires an extensive understanding of the target design problem. It can be implemented using predictive models of human perception or behavior, which can be either rule-based or data-driven, physical simulation, direct query-by-query response from human evaluators, *etc.*

CHI '21 Extended Abstracts, May 8–13, 2021, Yokohama, Japan

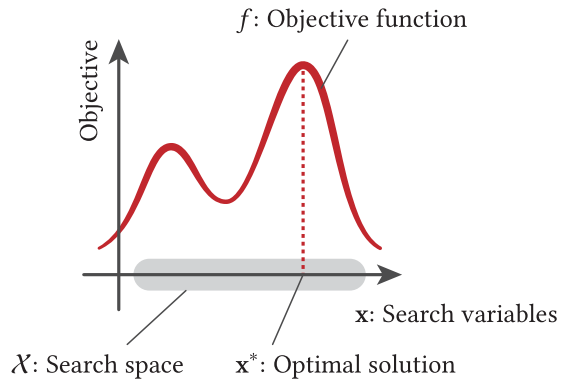

Figure 1: Illustration of basic concepts in optimization. An optimization problem is considered as a problem of finding the maximizer of the objective function within the search space. We apply this framework to various design problems.

Once the target design problem is formulated as an optimization problem, we can solve it by applying existing optimization algorithms (*e.g.*, gradient descent, Newton's method, and simulated annealing) from existing libraries (*e.g.*, SciPy [16]) in most cases. Note that researchers need to choose an appropriate optimization algorithm that is compatible with the target problem. This course will explain basic considerations for the choice, but for those interested in further details, we suggest referring to the book [10].

1.2 Human in the Loop

Objective functions often involve subjective preferential evaluation. For example, the goal of photo color enhancement is typically to find the most "subjectively pleasing" photo enhancement by adjusting parameters such as brightness, contrast, and saturation [6]. In such cases, it is often difficult to implement appropriate objective functions since accurately predicting human preference is a challenging task.

Human-in-the-loop optimization is an effective approach to handling human preference in computational design systems. In this approach, the computational design system iteratively asks human evaluators (*e.g.*, the user [7] or crowd workers [8]) to perform some microtasks while running the optimization process. That is, the system considers the human evaluators as a processing module that plays the role of the objective function, in the spirit of *human computation* [13]. Another effective approach is to use *preference learning* techniques [5, 6] for approximating the preferential objective function and then let users manually find the optimal solution with the help of this approximated objective function.

When gathering subjective preferential feedback from human evaluators, we need to care about how to design queries. In general, researchers consider *relative* assessment (*e.g.*, let evaluators choose the best design from multiple options) to be more suitable than *absolute* assessment (*e.g.*, let evaluators provide a score value for a specific design) [4, 14]. This course will discuss how this consideration has been incorporated into computational design frameworks by reviewing several representative works.

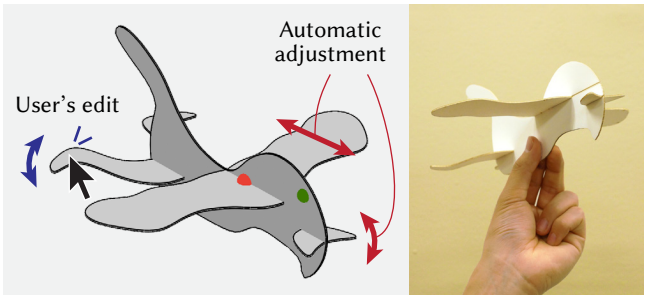

Figure 2: Example of computational design for personal fabrication [15]. The system assists users in designing free-form model airplanes that fly well. It automatically adjusts design parameters to maximize the flyability.

1.3 Examples

Figure 2 shows an example of computational design for personal fabrication [15], in which a system for designing free-form model airplanes is presented. To ensure that the airplane can fly well without letting users perform time-consuming fabricate-and-test iterations, the system automatically adjusts design parameters (*e.g.*, wing positions) by solving an optimization problem, where the "flyability" (*i.e.*, how long and stably the airplane can fly) is considered the objective function to maximize.

Figure 3 shows an example of human-in-the-loop computational design for parametric visual design [7], in which the presented human-in-the-loop optimization method is applied to photo color enhancement. In this design problem, the user needs to adjust design parameters (*e.g.*, brightness, contrast, and saturation) such that the target photograph looks subjectively best. The system finds the optimal parameter set by iteratively asking the user to perform a simple assessment task (clicking the best option among some options in this case).

2 BENEFITS

This course aims to let students and researchers who are novices to computational design become ready to start research on this topic. Specifically, this course tries to offer the following benefits to the audience:

- Understanding of fundamental concepts in computational design and an overview of this topic.
- Understanding of how researchers have formulated design problems as optimization in various application domains.
- Understanding of common considerations when human evaluation is involved in optimization problems.

3 INTENDED AUDIENCES

Anyone with interest in computational design can attend this course. Students, researchers, and practitioners who are novices to this topic are especially welcomed.

4 PREREQUISITES

This course is self-contained, and no particular prerequisite knowledge is required. This course will involve some basic concepts of

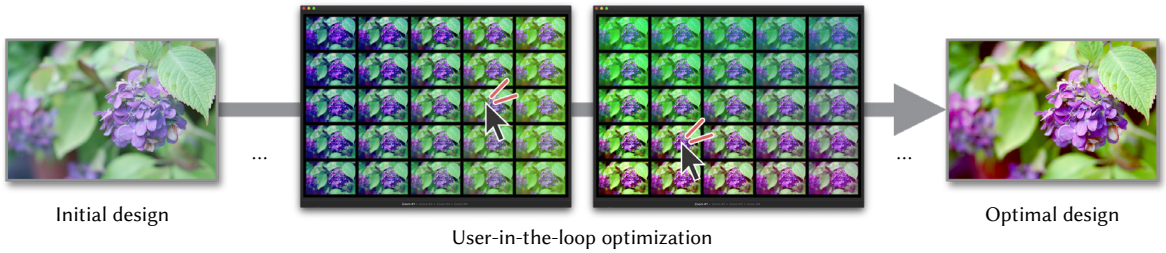

Figure 3: Example of human-in-the-loop computational design for parametric visual design [7]. The system optimizes the target design parameters by iteratively asking the user to perform a simple task to find the best design.

Table 1: Course content.

Content	Duration
Part 1: What is Computational Design?	45 minutes
Part 2: Human in the Loop	25 minutes
Part 3: Summary	5 minutes

mathematics and coding, but we will explain the necessary concepts during the course.

5 CONTENT

Table 1 shows the course content, which consists of three parts. In the first two parts, we will introduce many representative works from various application domains. This is not intended as a comprehensive survey but as material for a better understanding of computational design, and so we will focus on explaining why and how each of the works can be considered computational design. The last part summarizes the course and mentions future challenges.

6 PRACTICAL WORK

This course will be provided in a lecture-style format, but it will also include a coding demo for a toy design optimization problem. We plan to allow participants to access the code on web browsers and run it by themselves later so that they can get a sense of what coding in computational design is like.

7 INSTRUCTOR BACKGROUND

Yuki Koyama is a Researcher at National Institute of Advanced Industrial Science and Technology (AIST), Japan. He has been working on computational design for years in HCI and graphics domains and published papers on this topic at important venues such as CHI, UIST, SIGGRAPH, and SIGGRAPH Asia. His recent interest is to apply computational techniques for formulating and supporting design processes that involve preferential assessment by human; for example, he has worked on supporting parametric visual designs using crowdsourcing, machine learning, and Bayesian methods [5–8]. He is also interested in solving computational design problems in the personal fabrication domain [3, 15]. He received his Ph.D. from the University of Tokyo in 2017.

8 RESOURCES

Details of the course will be published at https://koyama.xyz/courses/chi2021-computational-design/. Learning materials for the course will be made available before the CHI conference. Some of the course topics are also discussed in details in the book chapter [4] written by the instructor and his colleague.

REFERENCES

[1] Moritz Bächer, Benjamin Hepp, Fabrizio Pece, Paul G. Kry, Bernd Bickel, Bernhard Thomaszewski, and Otmar Hilliges. 2016. DefSense: Computational Design of Customized Deformable Input Devices. In *Proc. CHI '16*. 3806–3816. https://doi.org/10.1145/2858036.2858354

[2] Xiaojun Bi, Otmar Hilliges, Takeo Igarashi, and Antti Oulasvirta. 2017. Computational Interactivity (Dagstuhl Seminar 17232). *Dagstuhl Reports* 7, 6 (2017), 48–67. https://doi.org/10.4230/DagRep.7.6.48

[3] Eisuke Fujinawa, Shigeo Yoshida, Yuki Koyama, Takuji Narumi, Tomohiro Tanikawa, and Michitaka Hirose. 2017. Computational Design of Hand-Held VR Controllers Using Haptic Shape Illusion. In *Proc. VRST '17*. 28:1–28:10. https://doi.org/10.1145/3139131.3139160

[4] Yuki Koyama and Takeo Igarashi. 2018. Computational Design with Crowds. In *Computational Interaction*, Antti Oulasvirta, Per Ola Kristensson, Xiaojun Bi, and Andrew Howes (Eds.). Oxford University Press, Chapter 6, 153–184. https://doi.org/10.1093/oso/9780198799603.003.0007

[5] Yuki Koyama, Daisuke Sakamoto, and Takeo Igarashi. 2014. Crowd-Powered Parameter Analysis for Visual Design Exploration. In *Proc. UIST '14*. 65–74. https://doi.org/10.1145/2642918.2647386

[6] Yuki Koyama, Daisuke Sakamoto, and Takeo Igarashi. 2016. SelPh: Progressive Learning and Support of Manual Photo Color Enhancement. In *Proc. CHI '16*. 2520–2532. https://doi.org/10.1145/2858036.2858111

[7] Yuki Koyama, Issei Sato, and Masataka Goto. 2020. Sequential Gallery for Interactive Visual Design Optimization. *ACM Trans. Graph.* 39, 4 (July 2020), 88:1–88:12. https://doi.org/10.1145/3386569.3392444

[8] Yuki Koyama, Issei Sato, Daisuke Sakamoto, and Takeo Igarashi. 2017. Sequential Line Search for Efficient Visual Design Optimization by Crowds. *ACM Trans. Graph.* 36, 4 (July 2017), 48:1–48:11. https://doi.org/10.1145/3072959.3073598

[9] Vittorio Megaro, Bernhard Thomaszewski, Maurizio Nitti, Otmar Hilliges, Markus Gross, and Stelian Coros. 2015. Interactive Design of 3D-Printable Robotic Creatures. *ACM Trans. Graph.* 34, 6 (Oct. 2015), 216:1–216:9. https://doi.org/10.1145/2816795.2818137

[10] Jorge Nocedal and Stephen J. Wright. 2006. *Numerical Optimization* (2nd ed.). Springer Science+Business Media. https://doi.org/10.1007/978-0-387-40065-5

[11] Antti Oulasvirta, Niraj Ramesh Dayama, Morteza Shiripour, Maximilian John, and Andreas Karrenbauer. 2019. Combinatorial Optimization of Graphical User Interface Designs. *Proc. IEEE* 108, 3 (February 2019), 434–464. https://doi.org/10.1109/JPROC.2020.2969687

[12] Romain Prévost, Emily Whiting, Sylvain Lefebvre, and Olga Sorkine-Hornung. 2013. Make It Stand: Balancing Shapes for 3D Fabrication. *ACM Trans. Graph.* 32, 4 (July 2013), 81:1–81:10. https://doi.org/10.1145/2461912.2461957

[13] Alexander J. Quinn and Benjamin B. Bederson. 2011. Human Computation: A Survey and Taxonomy of a Growing Field. In *Proc. CHI '11*. 1403–1412. https://doi.org/10.1145/1978942.1979148

[14] Kristi Tsukida and Maya R. Gupta. 2011. *How to Analyze Paired Comparison Data*. Technical Report UWEETR-2011-0004. University of Washington, Department of Electrical Engineering. https://vannevar.ece.uw.edu/techsite/papers/refer/UWEETR-2011-0004.html

[15] Nobuyuki Umetani, Yuki Koyama, Ryan Schmidt, and Takeo Igarashi. 2014. Pteromys: Interactive Design and Optimization of Free-Formed Free-Flight Model Airplanes. *ACM Trans. Graph.* 33, 4 (July 2014), 65:1–65:10. https://doi.org/10.1145/2601097.2601129

[16] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods* 17 (2020), 261–272. https://doi.org/10.1038/s41592-019-0686-2