Report from Dagstuhl Seminar 17232

Computational Interactivity

Edited by

Xiaojun Bi[1], Otmar Hilliges[2], Takeo Igarashi[3], and Antti Oulasvirta[4]

1 **Stony Brook University , US,** xiaojun@cs.stonybrook.edu
2 **ETH Zürich, CH,** otmar.hilliges@inf.ethz.ch
3 **University of Tokyo, JP,** takeo@acm.org
4 **Aalto University, FI,** antti.oulasvirta@aalto.fi

Abstract

This report documents the program and the outcomes of Dagstuhl Seminar on Computational Interaction organized in June 2017. The seminar focused on the use of computational methods to represent and enhance human-computer interaction. This topic is gaining traction, but efforts have been diluted over multiple research areas ranging from HCI to computer graphics and design. The main objective of the seminar was to get an overview and, moreover, discuss shared fundaments, such as what computational interaction is, formally and in practice. The seminar invitees were 22 researchers from areas such as Human-Computer Interaction, Computer Graphics, Operations Research, and more. The seminar consisted of three days of events, with emphasis on presentations, panels, and group discussions. The following summarizes the main outcomes.

Seminar June 5–8, 2017 – http://www.dagstuhl.de/17232
1998 ACM Subject Classification H5.m
Keywords and phrases crowd-computing, graphics, HCI, Machine learning, optimization, simulation
Digital Object Identifier 10.4230/DagRep.7.6.48
Edited in cooperation with Anna Maria Feit

1 Executive Summary

Xiaojun Bi
Otmar Hilliges
Takeo Igarashi
Antti Oulasvirta

The field of Human Computer Interaction (HCI) as a whole has been tremendously successful in the past both in terms of growth and impact of the premier academic conferences and in reshaping the IT industry. However, as we enter the post-PC era new technologies emerge and bring new challenges along that the traditional user-centered-design approach is not well equipped to meet. For example, artificial intelligence, wearable computing, augmented and virtual reality and custom interactive devices enabled by emerging digital fabrication technologies pose increasingly wicked challenges for interaction design, where designers must consider the entire stack from low-level hardware, through software all the way to the human

Computational Interactivity, *Dagstuhl Reports*, Vol. 7, Issue 06, pp. 48–67
Editors: Xiaojun Bi, Otmar Hilliges, Takeo Igarashi, and Antti Oulasvirta
Dagstuhl Reports
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

factors, implying that it is no longer feasible to abstract away technology and hence design spaces that explode in their complexity.

In June 2016 we assembled a group of 25 researchers to discuss aspects relating to a computational view of interactions. Through a series of talks, breakout discussions and panel discussions we established a broad consensus that HCI can and should be approached via a computational lense. We also identified several areas in which computational models are already being used to answer HCI research questions and to move the field forward. However, it became clear that the area is in its infancy and that much work is necessary to turn computational HCI into one of the mainstream approaches in the larger research community. Primarily, researchers and students need to begin thinking in computational terms (abstraction, modelling, automation) and need to learn how to incorporate such thinking into the typically more design driven thinking prevalent in current research. Furthermore, it was also discussed at length how state-of-the-art methods in numerical optimization and machine learning can advance HCI research and likewise how HCI research can identify and refine research requirements in these adjacent research communities.

In terms of concrete outcomes, many of the present researchers agreed to contribute to a forthcoming book on "computational interaction" and to write a joint overview article further refining the discussions and outcomes of the Dagstuhl seminar. In summary, a very fruitful and productive seminar led to interesting and in-depth discussions and provided starting points for much collaborative and community-driven future work. We also identified the need for further community building work including establishing of recurring workshops, symposia, and similar outlets, outreach via summer-schools, tutorials and other educational efforts as well as establishment of a sub-committee at ACM SIGCHI the premier venue for HCI research.

2 Table of Contents

3 Overview of Talks

3.1 Usor Economicus: Modelling Interaction with Economic Models

Leif Azzopardi (University of Strathclyde – Glasgow, GB)

Joint work of Guido Zuccon, Diane Kelly, Kathy Brennan, Leif Azzopardi
Main reference Leif Azzopardi: "Modelling interaction with economic models of search", in Proc. of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '14, pp. 3–12, ACM, 2014.
URL http://dx.doi.org/10.1145/2600428.2609574

In trying to understand how people interact with search systems I have developed several economic models of search and search behaviour. These models assume an Economic User (i.e. Usor Economicus), one who inevitably does that by which s/he may obtain the greatest amount of information and knowledge, with the smallest quantity of effort. In my talk, I first provided an overview of the typically Interactive Information Retrieval process. Then I introduced an economic model of search, which is derived from production theory. I showed how the model enables us to generate compelling, intuitive and crucially testable hypotheses about the search behaviour of users. They provide insights into how we can manipulate the system and the interface in order to change the behaviour of users. In a series of user experiments, I showed how well the models characterise, predict and explain observed behaviours (and where they fall down). I believe the models not only provide a concise and compact representation of search and search behaviour, but also provide a strong theoretical basis for future research into Interactive Information Retrieval. Furthermore, these economic models can be developed for all sorts of human computer interactions, and so are likely to provide many more insights into how people use systems and how we should design such systems.

3.2 Enabling Human-Data Supported Interfaces Through Computational Models of Human Behavior

Nikola Banovic (Carnegie Mellon University – Pittsburgh, US)

Joint work of Jennifer Mankoff, Anind K. Dey, Nikola Banovic
Main reference Nikola Banovic, Tofi Buzali, Fanny Chevalier, Jennifer Mankoff, Anind K. Dey: "Modeling and Understanding Human Routine Behavior", in Proc. of the 2016 CHI Conference on Human Factors in Computing Systems, pp. 248–260, ACM, 2016.
URL http://dx.doi.org/10.1145/2858036.2858557

User interfaces that learn about people's behaviors by observing them and interacting with them enable a future in which technology helps people to be productive, comfortable, healthy, and safe. However, despite current advances in Artificial Intelligence that make such interfaces possible, it remains challenging to ensure that models that power those interfaces are free of biases that may negatively impact people. Thus, to make user interfaces that have a positive impact on people requires technology that can accurately model people's behaviors. In this body of work, we focus on behaviors in the domain of human routines that people enact as sequences of actions they perform in specific situations, which we call behavior instances. We propose a probabilistic, generative model of human routine behaviors, that can describe, reason about, and act in response to people's behaviors. We ground our model in a holistic definition of human routines to constrain the patterns it extracts from the data to those that match routine behaviors. We train the model by estimating the likelihood that people will perform certain actions in different situations in a way that matches their demonstrated preference for those actions and situations in behavior logs. We leverage this computational model of routines to create visual analytics tools to aid stakeholders, such as domain experts and end users, in exploring, making sense of, and generating new insights about human behavior stored in large behavior logs in a principled way.

3.3 Rig Animation with a Tangible and Modular Input Device

Daniele Panozzo

Joint work of Alec Jacobson, Alex (Wan-Chun) Ma, Olga Sorkine-Hornung, Oliver Glauser, Otmar Hilliges, Daniele Panozzo
Main reference Oliver Glauser, Wan-Chun Ma, Alec Jacobson, Daniele Panozzo, Otmar Hilliges, Olga Sorkine-Hornung: "Rig animation with a tangible and modular input device", in ACM Trans. Graph., Vol. 35(4), pp. 144:1–144:11, 2016.
URL http://dx.doi.org/10.1145/2897824.2925909

We propose a novel approach to digital character animation, combining the benefits of tangible input devices and sophisticated rig animation algorithms. A symbiotic software and hardware approach facilitates the animation process for novice and expert users alike. We overcome limitations inherent to all previous tangible devices by allowing users to directly control complex rigs using only a small set (5-10) of physical controls. This avoids oversimplification of the pose space and excessively bulky device configurations. Our algorithm derives a small device configuration from complex character rigs, often containing hundreds of degrees of freedom, and a set of sparse sample poses. Importantly, only the most influential degrees of freedom are controlled directly, yet detailed motion is preserved based on a pose interpolation technique. We designed a modular collection of joints and splitters, which can be assembled to represent a wide variety of skeletons. Each joint piece combines a universal joint and two twisting elements, allowing to accurately sense its configuration. The mechanical design provides a smooth inverse kinematics-like user experience and is not prone to gimbal locking. We integrate our method with the professional 3D software Autodesk Maya® and discuss a variety of results created with characters available online. Comparative user experiments show significant improvements over the closest state-of-the-art in terms of accuracy and time in a keyframe posing task.

3.4 Optimization of Text Input

Anna Maria Feit (Aalto University, FI)

Joint work of Srinath Sridhar, Antti Oulasvirta, Andreas Karrenbauer, Daryl Weir, Mathieu Nancel, Christian Theobalt, Anna Maria Feit
Main reference Anna Maria Feit: "Computational Design of Input Methods", in Proc. of the 2017 CHI Conference on Human Factors in Computing Systems, Extended Abstracts., pp. 274–279, ACM, 2017.
URL http://dx.doi.org/10.1145/3027063.3027134

Designing a user interface or input method requires evaluating and trading-off many criteria. The corresponding design spaces are huge, making it impossible to manually build and test every potential design. For example, if we want to design a method to enter letters in mid-air via finger gestures, there are $10^3 3$ possibilities to assign 27 characters to 32 hand gestures. Therefore, my work focuses on using optimization methods for the design of (text input) systems. The use of optimization methods allows us to efficiently and rigorously search very large design spaces, it gives quantitative guarantees on the goodness of a design and helps us to explicitly formulate and trade-off different criteria and constraints. Using optimization methods to design an input method or user interface requires 3 steps.

The first step is how to formulate the design problem. This requires to explicitly state the decisions and constraints in a mathematical way and helps to understand the characteristics of the problem. Second, we need to formulate an objective function that can be used to evaluate and compare different designs. The input data and models we use for evaluation determines the outcome of the optimization. Third, in order to solve the optimization problem, we need computational search methods. This can be mathematical solvers which guarantee to find the optimum, or it can be approximation algorithms.

I present several projects in which the focus is to develop more plausible, empirically valid formulations and objectives for more realistic optimization approaches. Among others, we modeled the performance and anatomical constraints of the hand to computationally optimize multi-finger gestures for mid-air input [1] and studied how people type on physical keyboards, in order to understand the performance of two-hand typing [2]. Most recently, we used Integer Programming to optimize the special character layout of the French keyboard to facilitate typing of correct French. Therefore, we reformulated the commonly used letter assignment problem to make it applicable to large real-world cases with over 120 to-be-mapped characters and quantified the performance, ergonomics, and ease-of-use of a keyboard layout. While my work focuses on text entry, the same problem formulations and optimization methods can be applied to the design of many other input methods and user interfaces.

References

1 Sridhar, S., Feit, A. M., Theobalt, C. and Oulasvirta, A. *Investigating the dexterity of multi-finger input for mid-air text entry*. In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, ACM, 2015.

2 Feit, A. M., Weir, D. and Oulasvirta, A. *How we type: Movement strategies and performance in everyday typing*. In Proceedings of the 34th Annual ACM Conference on Human Factors in Computing Systems, ACM, 2016.

3.5 From n=15 to n=15,000 – Recruiting Unpaid Volunteers For Computational Design Research

Krzysztof Gajos (Harvard University, US)

Joint work of Katharina Reinecke, Krzysztof Z. Gajos
Main reference Katharina Reinecke, Krzysztof Z. Gajos: "LabintheWild: Conducting Large-Scale Online Experiments With Uncompensated Samples", in Proc. of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing, CSCW 2015, pp. 1364–1378, ACM, 2015.
URL http://dx.doi.org/10.1145/2675133.2675246

Many Computational Design projects rely on large quantities of user data. Currently, most researchers obtain such data from public data sets, collaborations with external organizations, recruitment of paid online participants or simulation. We offer an alternative: directly recruiting unpaid, intrinsically-motivated online volunteers. For over three years now, we have run LabintheWild.org, an online platform for conducting behavioral studies with unpaid online volunteers. Our participants are motivated by the promise of finding out something about themselves and the ability to compare their performance or preferences with others. The results of our validation studies demonstrate that results of studies performed with such curiosity-motivated participants match those obtained in conventional laboratory studies as long as a few best practices are followed [1]. Depending on topic and design, individual LabintheWild studies are completed by between one hundred and several thousand participants weekly. LabintheWild was developed without any external funding and without leveraging the brand of our university (Harvard). In other words, we believe this methodology is accessible to all researchers regardless of budget or affiliation. For researchers interested in launching their own studies on LabintheWild, we offer guidelines, a tutorial and code templates at http://labinthewild.org/researchers.php.

References

1 Reinecke, K., and Gajos, K. Z. LabintheWild: conducting large-scale online experiments with uncompensated samples. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, CSCW '15, ACM (New York, NY, USA, 2015), 1364–1378.

3.6 Human-Math Interaction

Alec Jacobson (University of Toronto, CA)

Human-computer interaction (HCI) both studies how people interact with computers and designs new ways for people and computers to interact. In this talk, I propose human-*math* interaction (HMI), a new field of research that would study how people interact with math and design new ways to interact with math. I purposefully treat computers and math as distinct — if only as a thought experiment. HCI has been so successful for computing, I ask whether we can return the favor to applied math. As a motivating example, I first look back on how different notations for calculus provide better or worse affordances for theorem proofs or mathematical derivations. Then I turn to a ripe opportunity for HMI: the interaction with partial differential equations (PDEs) common to computer graphics and engineering. Traditional interactions are via raw boundary conditions. I show an example in the context of computer animation of how analysis of the PDE can lead to homogenization of boundary conditions relieving the burden of their explicit specification from the human "user". In another prototypical scenario of smoothing data over a bounded domain, I show how a small change can lead to dramatically simple natural boundary conditions, allowing the user to focus on the primary smoothing parameters without also managing boundary behavior. I conclude with thoughts on future directions for HMI research.

3.7 Functionality Selection in Interaction Design via Discrete Optimization

Andreas Karrenbauer (MPI für Informatik – Saarbrücken, DE)

Joint work of Antti Oulasvirta, Anna Feit, Perttu Lähteenlahti, Andreas Karrenbauer
Main reference Antti Oulasvirta, Anna Maria Feit, Perttu Lähteenlahti, Andreas Karrenbauer: "Computational Support for Functionality Selection in Interaction Design", in ACM Trans. Comput.-Hum. Interact., Vol. 24(5), pp. 34:1–34:30, 2017.
URL http://dx.doi.org/10.1145/3131608

Designing interactive technology entails several objectives, one of which is identifying and selecting appropriate functionality. Given candidate functionalities such as "Print", "Bookmark", and "Share", a designer has to choose which functionalities to include and which to leave out. Such choices critically affect the acceptability, productivity, usability, and experience of the design. However, designers may overlook reasonable designs, because there is an exponential number of functionality sets and multiple factors to consider. To tackle this problem, we use discrete optimization techniques and propose algorithmic methods to support designers to explore alternative functionality sets in early stage design. Based on interviews with professional designers, we mathematically define the task of identifying functionality sets that strike the best balance among four objectives: usefulness, satisfaction, ease of use, and profitability. We develop an integer linear programming solution that can quickly solve very large instances (set size over 1,300) on a regular computer. Further, we build on techniques of robust optimization to search for diverse and surprising functionality designs. Empirical results from a controlled study and field deployment are encouraging. Most designers rated computationally created sets to be of the comparable or superior quality than their own. Designers reported gaining a better understanding of available functionalities and the design space.

3.8 The Future of Programming and Data Sciences

Jun Kato (AIST – Ibaraki, JP)

Joint work of Jun Kato, Takeo Igarashi, Masataka Goto
Main reference Jun Kato, Takeo Igarashi, Masataka Goto: "Programming with Examples to Develop Data-Intensive User Interfaces", in IEEE Computer, Vol. 49(7), pp. 34–42, 2016.
URL http://dx.doi.org/10.1109/MC.2016.217

In this talk, I discuss three gaps between development and runtime environments of the programs that prevent a fluid programming experience. The first gap is the lack of intuitive representation of complex big data in the integrated development environment (IDE) and can be addressed by adding IDE support for the "programming with examples" workflow [1], such as integrated graphical representations in IDEs. The second gap is the so-called gulf of execution, which forces the programmer to imagine the runtime behavior of the program from the abstract source code. This gulf can be addressed by "live programming" that makes programs editable while they virtually keep running. The third gap is the high threshold for the program users to modify it to match their needs. To fill the gap, the live programming technique can be expanded to provide the end-users partial but intuitive user interfaces to tune the program behavior during its runtime [2].

These research on programming experience has deepened the understanding of, and to some extent, deconstructed the definition of "programming". I foresee that the current programmers' role will be shared among more people with diverse technical backgrounds. Those who make most of such programming deconstruction include creators since the process of creating media content becomes more and more "computational" in the future, as the seminar title implies. In particular, I consider that the process of scientific research – the process of creating scientific knowledge – can be greatly enhanced. I believe that research in computational science, when combined with the programming experience research, will make science more reproducible, enable sharing the process rather than the results, and support reuse and mashups of scientific knowledge.

References

1 Jun Kato, Takeo Igarashi, Masataka Goto. *Programming with Examples to Develop Data-Intensive User Interfaces*. IEEE Computer 49(7), pp.34–42. 2016. http://junkato.jp/programming-with-examples

2 Jun Kato, Masataka Goto. *Live Tuning: Expanding Live Programming Benefits to Non-Programmers*. In Proceedings of the Second Workshop on Live Programming Systems. 2016. http://junkato.jp/live-tuning

3.9 Crowdsourcing Visual Design

Yuki Koyama (AIST – Ibaraki, JP)

Joint work of Issei Sato, Daisuke Sakamoto, Takeo Igarashi, Yuki Koyama
Main reference Yuki Koyama, Issei Sato, Daisuke Sakamoto, Takeo Igarashi: "Sequential line search for efficient visual design optimization by crowds", in ACM Trans. Graph., Vol. 36(4), pp. 48:1–48:11, 2017.
URL http://dx.doi.org/10.1145/3072959.3073598

Parameter tweaking is one of the most fundamental tasks in many design domains. The goal is to maximize the quality of designed objects based on some criteria (i.e., objective functions). In visual design domains, aesthetic preference (i.e., how aesthetically preferable the designed object looks) often plays the role of the objective function. However, as aesthetic preference is closely tied with human perception, it is difficult to mathematically quantify this criterion using only simple rules.

To support this preference-driven parametric design tasks, we have investigated the use of crowdsourced human computation. By asking crowd workers to perform perceptual microtasks in the form of function calls, it is possible to handle human preference in a computational manner. In this talk, I introduced the following two methods. The first method is to construct a computational preference model by analyzing data from crowds and then support interactive design exploration by a designer based on the constructed model [2]. The second method is to solve a numerical optimization problem in a crowds-in-the-loop manner so that the system can efficiently find the design that maximizes the aesthetic quality [1].

References

1 Yuki Koyama, Issei Sato, Daisuke Sakamoto, Takeo Igarashi. Sequential line search for efficient visual design optimization by crowds. *ACM Trans. Graph.*, Vol. 36(4), pp. 48:1–48:11, 2017.

2 Yuki Koyama, Daisuke Sakamoto, and Takeo Igarashi. Crowd-powered parameter analysis for visual design exploration. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, pages 65–74, 2014.

3.10 Probabilistic Models for Text Input and Beyond

Per Ola Kristensson (University of Cambridge, GB)

Main reference Per Ola Kristensson: "Next-Generation Text Entry", in IEEE Computer, Vol. 48(7), pp. 84–87, 2015.
URL http://dx.doi.org/10.1109/MC.2015.185

Next-generation text entry systems will need to satisfy multiple objectives, such as a high entry rate, low error rate, minimal training requirements, etc. However, via an analysis of the design space of mobile text entry methods we reveal that successful mainstream text entry methods must carry two traits in particular: 1) they must exhibit high immediate efficacy; and 2) they must have an effective entry rate which is at least as high as existing commercial alternatives. We demonstrate that insisting that a new design of a text entry method must carry these two traits leads to a very narrow design space with few viable alternative designs. However, we further demonstrate that there exist at least seven solution principles that can increase the design space: 1) from closed to open-loop; 2) continuous novice-to-expert transition; 3) path dependency; 4) flexibility; 5) probabilistic error correction; 6) fluid regulation of uncertainty; 7) efficiency.

3.11 A BIG (Bayesian Information Gain) Approach for Human-Computer Interaction

Wanyu Liu (ENST – Paris, FR)

Joint work of Rafael Lucas D'Oliveira, Michel Beaudouin-Lafon, Olivier Rioul, Wanyu Liu
Main reference Wanyu Liu, Rafael Lucas D'Oliveira, Michel Beaudouin-Lafon, Olivier Rioul: "BIGnav: Bayesian Information Gain for Guiding Multiscale Navigation", in Proc. of the 2017 CHI Conference on Human Factors in Computing Systems, pp. 5869–5880, ACM, 2017.
URL http://dx.doi.org/10.1145/3025453.3025524

We propose a novel approach BIG (Bayesian Information Gain), which is based on Bayesian Experimental Design using the criterion of mutual information from Information Theory. It captures the uncertainty the computer has about the user's goal as well as the information carried in the user input expressing what the user has in mind in the interaction. By maximizing the notion of the expected information gain, the computer can play a more active role, instead of simply executing user commands. We illustrated this idea with an instance in multiscale navigation – BIGnav. The controlled experiment demonstrated that BIGnav is significantly faster than conventional pan and zoom and requires fewer commands for distant targets, especially in non-uniform information spaces. Though being more efficient, BIGnav incurs higher cognitive load for the users, which leads us to consider more balanced interaction and shared control by leveraging the expected information gain.

3.12 Computational Interaction from a control theory perspective

Roderick Murray-Smith (University of Glasgow, GB)

As part of the Panel: "What is Computational Interaction?" I reviewed the role of control theory and dynamic systems theory in understanding common interaction techniques. Most existing techniques can potentially be described in a canonical differential equation form. I discussed how control can be seen to be at the foundations of Human–Computer Interaction and might be essential for making progress in novel forms of interface.

Traditional views of the human–machine control loop have focussed on the controlled application of external power sources. Here, a framework was proposed for considering how the human–computer interaction loop could be augmented with computational power. I considered how this could reduce mental effort, or empower disabled users to achieve more in their everyday lives.

However, the application of computational power has ethical consequences, depending on where the information supplied by the computational unit comes from.

I also discussed the limitations of control theory for the design of human–computer control loops compared to traditional engineering approaches. In contrast to traditional control theory, you cannot just swap out blocks in your control loop. Given the human's ability to predict the behaviour of other elements of the loop, and apply feed-forward compensation, we often find that the overall closed-loop behaviour does not change in as predictable a manner as in an automatic engineering system.

3.13 Personalized Interactive Devices

Jürgen Steimle (Universität des Saarlandes, DE)

Joint work of Jürgen Steimle, Daniel Groeger, Martin Weigel, Martin Schmitz, Aditya Shekhar Nittala, Alex Olwal, Tong Lu, Gilles Bailly, Antti Oulasvirta, Carmel Majidi, Jochen Huber, Niloofar Dezfuli, Max Mühlhäuser, Elena Chong Loo
Main reference Jürgen Steimle: "Skin-The Next User Interface", in IEEE Computer, Vol. 49(4), pp. 83–87, 2016.
URL http://dx.doi.org/10.1109/MC.2016.93

Advanced interactive devices for physical interaction are deployed in the user's body or embedded in the physical environment. This demands for personalized form factors and functionalities. Printable electronics, paired with computational design and fabrication, is an enabling technology for such personalized interfaces.

I gave an overview of our recent work on printed sensor and display surfaces. I presented techniques for realizing interactive temporary tattoos, which sense touch and deformation and offer visual output. These tattoos conform to fine wrinkles and highly curved body locations and hence turn the human skin into an input and output surface. They can be personalized to individual users and individual body locations through digital design.

Moreover, I presente novel ways of integrating sensing and output in 3D printed objects. These make use of material composites that comprise rigid, flexible, temperature-sensitive, and conductive 3D-printable materials. By 3D printing specific geometric patterns, the desired functional behavior is integrated. Localized touch sensors, deformation sensors, and shape-change capabilities can be realized at the desired locations within a 3D printed object.

Together, these approaches demonstrate the potential of digital fabrication and printed electronics for realizing sensors and output in new form factors, which are compatible with highly individual geometries and custom user preferences.

3.14 A Formal Methods Perspective on Computational Interaction

Harold Thimbleby (Swansea University, GB)

Best practice in HCI is to use iterative design (see figure 1), because getting complex human systems right the first time is hard, and the best solution is to implement a prototype, evaluate it, and improve it. Then repeat until the design meets the usability criteria (and other criteria such as performance and dependability).

In practice this approach does not work, since by the time a system is ready to be evaluated it is also probably ready to be sold and used — and users will anyway start getting used to and even start to rely on the suboptimal design features; further delay in release represents an opportunity cost. It is, moreover, hard work improving a complex interactive system program, particularly if evaluation suggests 'orthogonal' improvements the developers never originally anticipated. Why refactor when the first version is already in use? (This creates technical debt.)

Formal methods can help produce a more dependable and flexible initial implementation, for instance, that satisfies important principles like predictability. However, formal methods are often seen as arcane and even at odds with the 'user experience,' and in any case, for most developers, it is easier to get a program (seemingly) working than using best software engineering practice. The idea that formal methods somehow conflicts with user experience is common but is, nevertheless, a profound misunderstanding beyond the scope of this abstract to address.

A solution is to use computational interaction. Here, a separate module of the interactive program performs a restricted form of interaction $\mathcal{C}$, for instance, machine learning, signal processing, information theory, optimisation, inference, control theory or formal methods, as the case may be. This module now embodies mathematical reasoning, and the reasoning is parameterised (for instance, with data from user evaluations). This means that iterating the user interface with respect to $\mathcal{C}$ is now easy and reliable, for $\mathcal{C}$ preserves all the properties of machine learning or whatever.

Now the iteration of the ISO iterative design cycle iterates $\mathcal{C}$; this is fast and easy, if not trivial, and by appropriate choice of one or more $\mathcal{C}_i$ the most important features of the user interface quickly converge onto what the evaluation and data collection suggests. This is a radical improvement over conventional user interface design, provided only that the right data is collected.

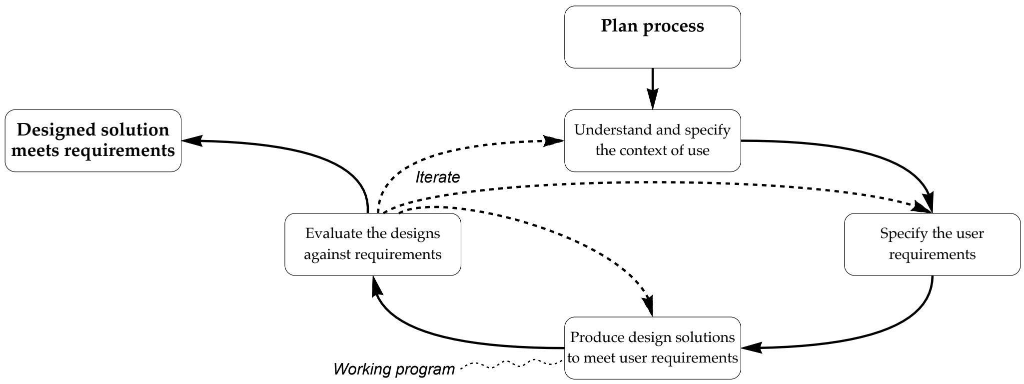

Figure 1 The international standards ISO 9241:210 iterative design cycle. Note that a working program is available before even one iteration is complete, and in conventional program development this may dissuade developers from embarking on the expensive iterative cycles. In contrast, computational interaction identifies a module that is easy to parameterise and easy to update from user evaluation data.

3.15 A Proposed Definition of Computational Interaction

John Williamson (University of Glasgow, GB)

Computational interaction is a distinctive approach to human-computer interaction, which emphasises the role of mathematical modeling of user-system behaviour and computation in augmenting the design and operation of interfaces. We propose some defining characteristics which computational interaction would typically be expected to have at least some of:

- an explicit mathematical model of user-system behavior;
- a way of updating that model with observed data from users;
- an algorithmic element that, using this model, can directly synthesise or adapt the design;
- a way of automating and instrumenting the modeling and design process;
- the ability to simulate or synthesise elements of the expected user-system behavior.

Potential advantages of computational approaches include:

- reduced design time of interfaces, as automation, data and models can supplant hand-tweaking;
- more robust and efficient interfaces, as better models can better predict how interactions evolve;
- better tailored interfaces: to users, contexts, devices, as structure can be learned rather than dictated
- new technologies can be harnessed quickly as fundamental, repeatable processes generalise to new contexts
- the evaluation burden can be reduced, as strong, executable models can predict much of expected user behavior
- our ability to reason rather than rely on experimentation increases as HCI problems can be defined formally
- algorithmic design can support designers in tedious tasks and help designers focus on creative aspects of design

4 Working groups

4.1 Modeling for Computational HCI

Leif Azzopardi (University of Strathclyde – Glasgow, GB), Nikola Banovic (Carnegie Mellon University – Pittsburgh, US), Andreas Karrenbauer (MPI für Informatik – Saarbrücken, DE), and Wanyu Liu (ENST – Paris, FR)

Computational models of human behavior, interactive systems, and the interaction between human and the system are the foundation of computational interactivity. However, despite availability of such models, user experience (UX) designers still mostly depend on intuition, supported by existing knowledge and experience, when designing user interfaces. This likely follows from lack of awareness about the existing computational models, lack of information about the benefits of such models, and the difficulty that designers face when integrating such models into their work. Our attempt to provide a brief taxonomy of such models is a first step in bringing these models closer to the UX design community. We propose the following dimensions that describe the space of computational models: 1) granularity – the level at which humans, systems, and interactions are being modeled (e.g., cognitive, activity, task); 2) realism – how closely to the real world the model attempts to represent information; 3) flexibility – the ability of the model to change in response to new information; 4) dynamicity – whether the model is static (offline) or active (online); 5) explainability – if the model is predictive or explanatory; and 6) origin – data driven or theory driven. These dimensions offer the ability to compare and contrast the existing models and provide an overview of the current state of the field. It is also a first step towards defining metrics to evaluate such computational models to ensure they provide scientific, principled foundation for computational interactivity. These dimensions will lead to the development of new and the improvement of existing computational models and offer a promise of a unified, overarching model of human behavior which will drive the future of computational interactivity.

4.2 Challenges and Problems in Computational Interaction

Alec Jacobson (University of Toronto, CA), Anna Maria Feit (Aalto University, FI), Krzysztof Gajos (Harvard University, US), and Harold Thimbleby (Swansea University, GB)

Our working group undertook the daunting task of defining "grand challenges for computational interaction". After brainstorming, the group separated proposed challenges into two categories: A) ways computational interaction can make the world a better place and B) technical challenges within computational interaction research. The working definition of "computational interaction" was based on the notion that computational interaction is interaction that improves with improved computational power. We identified two modalities on the spectrum of computational interaction. I) Interactions that are created as the result of intensive computation. For example, using machine learning to determine optimal placement for buttons or knobs. II) Interactions that involve intensive computation at run time. For example, a gesture based controller that learns to better recognize a particular user's motions after continued use. With these possibilities in mind, we enumerate promising and potentially impactful directions for the field.

Ways computational interaction can make the world a better place:

1. Automatically generate personalized curricula for people given their knowledge goals, current knowledge, and cognitive abilities
2. Automatically plan meeting activities given goals, time budget and personalities of participants
3. Interactively model solution spaces for designers or design teams and identify under-explored regions of this space by breaking implicit, human assumptions
4. Invent computational tools for facilitating arguments and mediating conflicts
5. Aid understanding of deep neural networks by designing human-understandable networks or providing tools to determine whether a network is understandable
6. Invent information extraction tools for life-long datasets such as photos, GPS data or files
7. Improve user experiences in mixed initiative interactions by modeling predictability (user's guess about the system) and uncertainty (system's guesses about the user)

Technical challenges for computational interaction

1. Create benchamarks for quantitative comparisons for fundamental computational interaction problems à la computer vision.
   a. Utilize labinthewild.org for open-source, large-scale testing
   b. For example, X minutes to build a model of the user, then quantify improved performance of some UI Y.
   c. Info-vis is especially ripe for such testing. Fix dataset and measure time and how well a visualization conveys information.
   d. Other immediately testable tasks are text input, GUIs, and pointing.
   e. Unlike fully automatic benchmarks in computer vision, extra care will be needed to prevent bias, noise, and foul play.
2. Foster a community of open-source, shared, reusable toolboxes. Upstart time for experiments could be dramatically reduced by collectively creating plugins for common methods and models in the field. Reproducibility should increase as well.

Solving these challenges would mark great strides for the emerging field of computational interaction.

4.3 What is Computational HCI? Definition, Types, and Scope

Per Ola Kristensson (University of Cambridge, GB), Andrew Howes (University of Birmingham, GB), Jun Kato (AIST – Ibaraki, JP), and John Williamson (University of Glasgow, GB)

In this working group, we discussed a definition of Computational HCI. The outcome of the discussion was the following definition:

Computational HCI applies computational thinking (abstraction, automation, analysis) to explain and enhance the interaction between a user (or users) and a system. It is underpinned by modelling which admits formal reasoning and involves at least one of:

- a way of updating a model with observed data from users;
- an algorithmic element that, using a model, can directly synthesise or adapt the design;
- a way of automating and instrumenting the modeling and design process;
- the ability to simulate or synthesise elements of the expected user-system behavior.

Computational HCI is founded on fundamental theoretical considerations but is constructive in its approach rather than descriptive. It is empirical and quantitative but focuses on how to use computationally powered models to do what could not be done before rather than describing what has been done before. It emphasises generating motor themes in HCI, and robust, replicable and durable approaches which go beyond point sampling of the interaction space.

4.4 Machine Learning for and in HCI

Roderick Murray-Smith (University of Glasgow, GB), Emre Aksan (ETH Zürich, CH), Otmar Hilliges (ETH Zürich, CH), Takeo Igarashi (University of Tokyo, JP), and Jerry Talton (Slack Technologies – New York, US)

This working group discussed the relevance of machine learning (ML) with human computer interaction (HCI) and raised questions in order to better address problems of HCI research. Our discussion consisted of four main topics:

1. Why has HCI often not seen the 'ratchet effect' seen in other fields where people productively build on the progress of others?
   - How can we change this (motor themes)?
   - Would having an HCI Kaggle/OpenAI Gym like combo be beneficial to encourage sharing of data, models, and problems?
   - How do you generate, adapt and curate data in HCI for ML tools?

   We anticipate triple benefits for the community:
   - Defining problems and providing datasets generates citations.
   - Beating established baselines generates progress (and papers).
   - Solving hard problems requires the development of new methods (and creation of insights) that may become useful outside of academic research.
2. What is different about the nature of HCI compared to typical problems in ML?
   - Data is more expensive (sometimes).
   - Users (and hence data) adapt over time which requires 'living' datasets (see information retrieval approaches).
   - Conventional metrics in ML are not representative enough. For example, optimizing (e.g.) recognition accuracy is a poor proxy for usability.
   - Users learn and adapt, which means that cost functions and data changes.
   - Variability across users (behavior, preferences) is high (high variance problems typically require a lot of data).
3. Should we approach ML as a tool or invest resources in ML research for HCI problems?
   - Should HCI researchers be more involved with ML and seek solutions for HCI related issues?
   - How can you calculate usability gradients through the user? Can we optimise usability as a cost function?
   - How do you optimise usability when the human is adapting continuously to the new system?
   - How do we enable and ensure a smooth co-evolution of the user and system to a state of improved interaction?

  - How do you design for the individual (en masse, via adapting to each end user)?
  - Many recent ML approaches learn from large datasets and then become rigid. Adaptive algorithms such as one-shot learning should be favored.

4. What can HCI contribute to ML?
   - Can we enable nonML-trained end users to productively use machine learning?
   - Can we 'un-black-box' ML models for naive users?
   - ML is making certain interaction scenarios worth investigating again from an HCI perspective. (e.g. sudden use in speech-based interaction in home settings with Alexa)
   - Has HCI then lost 'control'? (e.g. much modern commercial VR/AR activity ignored the older AR/VR research.)

4.5 Computational Design

Antti Oulasvirta (Aalto University, FI), Yuki Koyama (AIST – Ibaraki, JP), Ranjitha Kumar (University of Illinois – Urbana-Champaign, US), Daniele Panozzo (New York University, US), and Jürgen Steimle (Universität des Saarlandes, DE)

This workgroup looked at the intersection of computational design and computational interaction. This intersection is highly relevant for HCI research because although interactive computational design methods are used across several fields, there seems to be no principled understanding of how to design efficient user interfaces for these tools. Computational design is researched, among others, in graphics, visualization, computer-aided design, engineering design, and architecture, and there are multiple fields that come close, such as recommendation systems and computational creativity. To begin with, the group defined computational design as (1) influencing the design of an artefact, (2) supporting any part of the design process from user research to generation and evaluation (although it noted that most work focuses on generation), (3) represents the problem mathematically as the problem of finding the best design among some options, and (4) solves it using algorithmic solution methods that broadly fall within the scope of mathematical optimization.

There are several practical and scientific benefits to this approach, which were observed to stem from two sources: (1) new ways of representing design problems (a new way to think and imagine design); (2) increased problem-solving capacity (e.g., solving problems in design that were previously unsolvable). The group made a somewhat startling conclusion: at the core of it computational design is applied math, and in particular mathematical optimization. Interactive design tools in computational design are essentially about interfacing mathematics. The group concluded that this poses a grand challenge to computational HCI: what is the best way to interface mathematics in a given design project? Can something foundational be said about the success or performance of a designer using a particular interaction style in a particular project, such as about success probability, time cost, or mental load? The challenge for HCI research is to identify ways to interface objective functions with their constraints and parameters. This invites research on how to obtain, edit, analyze, visualize, validate, and interpret objective functions as part of a design process.

The group suggested that while an essential part of this multi-disciplinary field has looked at the generation or refinement of designs (construction), there is a significant opportunity space in other parts of the design process, such as user research, sketching, prototyping, deployment, and evaluation. Another significant contribution HCI research can make in this space is systematic reporting of the needs, requirements, and practices of professional and non-professional designers. The group concluded with five objectives for future work: (1) Formulate the research problems of (interactive) computational design; (2) Define success metrics of interactive computational design; (3) Review methods explored and progress achieved this far; (4) Articulate the changing roles, requirements, and practices of design; and (5) To position computational design in user-centered design, its position to current understanding of design thinking must be articulated.

Participants

- Emre Aksan
ETH Zürich, CH
- Leif Azzopardi
University of Strathclyde – Glasgow, GB
- Nikola Banovic
Carnegie Mellon University – Pittsburgh, US
- Anna Maria Feit
Aalto University, FI
- Krzysztof Gajos
Harvard University, US
- Otmar Hilliges
ETH Zürich, CH
- Andrew Howes
University of Birmingham, GB
- Takeo Igarashi
University of Tokyo, JP
- Alec Jacobson
University of Toronto, CA
- Andreas Karrenbauer
MPI für Informatik – Saarbrücken, DE
- Jun Kato
AIST – Ibaraki, JP
- Yuki Koyama
AIST – Ibaraki, JP
- Per Ola Kristensson
University of Cambridge, GB
- Ranjitha Kumar
University of Illinois – Urbana-Champaign, US
- Wanyu Liu
ENST – Paris, FR
- Roderick Murray-Smith
University of Glasgow, GB
- Antti Oulasvirta
Aalto University, FI
- Daniele Panozzo
New York University, US
- Jürgen Steimle
Universität des Saarlandes, DE
- Jerry Talton
Slack Technologies – New York, US
- Harold Thimbleby
Swansea University, GB
- John Williamson
University of Glasgow, GB