Tools of Expression: Notation and Interaction for Design Computation

Robert Aish
Director of Software Development, Autodesk

Design considers function, fabrication, and aesthetics collectively. Computation is beginning to affect the competitive dynamics of design. Using algorithms, designers are exploring forms that are essentially "undrawable," even with advanced modeling and direct manipulation techniques. Determining the appropriate functional characteristics may require the application of increasingly complex structural- and environmental-performance analysis techniques. To realize physically a design may require further geometric analysis and rationalization, and the use of complex computer-controlled fabrication techniques.

Design and Computation are converging, whether this is in the use of computational techniques to realize conventionally drawn or modeled configurations or to generate the original form or configuration. In fact, the fundamental basis for Design Computation is the recognition that algorithms are an expressive medium of design: algorithms are not just the prerogative of computer scientists; designers with computational skills can develop them, and these algorithms can play a critical role in the design of buildings and other artifacts. The implications are:

- Design Computation allows the creation of geometric forms and configurations that would be difficult, if not impossible, to conventionally draw or model by hand with interactive graphical manipulation.

- Changes to the driving parameters or to the underlying algorithm allow for the rapid exploration of alternative forms and configurations (again with greater ease than with conventional design methods).

- Combining generative and evaluative computational techniques enables a process of "form finding," in which the design's function, fabrication, and aesthetic characteristics can be synthesized (in ways previously unachievable).

Design Computation suggests a different approach to design. The designer need no longer focus on designing a specific artifact (or configuration), but rather, he has the opportunity to externalize the underlying "design logic" and use this to explore a whole range of alternative solutions. Design Computation also suggests a different approach to the development of software applications. Instead of presenting a completely comprehensive "view" of the design domain, the design application effectively becomes a platform that presents a more general abstraction of geometry and composition. The designer (with computational skills) uses this platform to create the final, customized layer of application logic, specific to the design problem at hand.

Essentially, with Design Computation, the designer is moving from "doing" (directly designing) to "controlling" (defining the algorithm and the parameters with which the design or range of designs will be generated), and the role of the programmer is being blurred between the professional software developer and the designer as "end-user programmer."

As the title of this paper suggests, our focus is to develop "tools of expression" for designers. This involves creating an appropriate "notation for design logic" (in the sense of an end-user programming language), as well as "instruments" that can present the expressive possibilities of this notation to a creative audience of designers (in the sense of interactive applications). These applications should not only be able to be "played" intuitively, but also support and respond to "compositions" that are more complex. In addition, these applications should be capable of progressively

revealing the potential of the underlying notation and allow for the expression of design that is more consciously considered. The notation and related interactive applications present a unique opportunity to understand and support the diversity of design, which spans, potentially, from the improvised to the contrived.

The essential challenge here is to understand the learning curve that might enable designers without formal computational training to explore progressively the new possibilities that Design Computation has to offer. However, to be successful, the designer must be willing to free himself from established design patterns and rebuild his design knowledge based on a fundamental understanding of geometric and logical abstraction.

There is no imperative that suggests that Design Computation is “inevitable” or “has to be adopted.” It is an option, a possibility, for those creative designers for whom geometry and logic are the essential foundations of design, and who enjoy moving effortlessly between the abstract and the practical and understand the potential of computation as an expressive medium of design.