An Introduction to Aesthetic Computing

Paul Fishwick

In this brief introduction to a new area of study, aesthetic computing, we first define the terminology, then position the area in the context of related fields that combine art, mathematics, and computing. Aesthetic computing is concerned with the impact and effects of aesthetics on the field of computing. This text is divided into two primary sections. The first section we discuss aesthetics, art, and the motivation for defining another hybrid phrase. The attempt here is to capture the field by historical context, definition, and a graphical illustration. The close relationship between aesthetics and art (i.e., aesthetics being the philosophy of art) is justified with citations to recent literature, to the point we can use the two words interchangeably. In the second section, we describe research on novel representations created locally at the University of Florida, in the aesthetic computing class and the simulation and modeling research laboratory.

To help spur a discussion in aesthetic computing, an attempt to bring several key researchers and practitioners to the same table prompted a meeting in Dagstuhl, Germany (Dagstuhl), in mid-July 2002. We held a week-long workshop, organized by Roger Malina, Christa Sommerer, and myself. More than thirty representatives of art, design, computer science, and mathematics attended the workshop, which was cosponsored by Dagstuhl and Leonardo (Leonardo). The purpose of the workshop was to carve out an area, or at least to see whether this was possible, based on the notion that aesthetics and art could play a role in computing disciplines. A manifesto was created on the last day of the workshop as a preliminary definition for the area, and was recently published in Leonardo (Fishwick 2003).
Aesthetics and Art

Aesthetic computing is the application of aesthetics to computing. The goal of aesthetic computing is to affect areas within computing, which for our purposes, will be defined broadly as the area of computer science. With respect to aesthetics, this goal also includes the idea that the application of aesthetics to computing and mathematics, the formal foundations for computing, can extend beyond classic concepts such as symmetry and invariance to encompass the wide range of aesthetic definitions and categories normally associated with making art. One might, for example, represent structures in computing using the style of Gaudi or the Bauhaus school. The words aesthetics and computing need further discussion before we proceed. “Aesthetics” stems from the Greek αἰσθητική aisthetiḱ́, derived from aisthesis (i.e., perceived by the senses). Plato’s aesthetics revolved around his forms, and Greek society stressed mimesis (i.e., imitation, mimicry) as central to art’s purpose. Within the continuing history of aesthetics, prior to Kant’s Critique (1790) and including Baumgarten’s (1750) introduction of aesthetics as the science of the beautiful, art and aesthetics have not been well connected. Art was generally not associated with aesthetics, and aesthetics as an area within philosophy was not focused on art. Since Kant’s treatise, aesthetics has been expanded to encompass both the logical and the perceptual. The Oxford English Dictionary (2003) contains the following two definitions for aesthetics: (1) the science that treats the conditions of sensuous perception; and (2) the philosophy or theory of taste, or of the perception of the beautiful in nature and art. In the Encyclopedia of Aesthetics, one of the most comprehensive references on the subject, spanning four volumes, Kelly (1998, p. 11) in his preface, states

Ask contemporary aestheticians what they do, however, and they are likely to respond that aesthetics is the philosophical analysis of the beliefs, concepts, and theories implicit in the creation, experience, interpretation, or critique of art.

Kelly proceeds to highlight the goal of the encyclopedia, which is “to trace the genealogy of aesthetics” in such a way as to integrate both its philosophical and its cultural roles. The word “art,” in the sense in which Kelly discusses aesthetics, is defined broadly enough to combine logical as well as material aspects, or computing and art. Thus, an elegant computer program and a sculpture are both forms of art. Furthermore, one may speak generally of aesthetics in terms of symmetry and harmony or, more singularly, in terms of the aesthetics of the artist Dali, for example, or the surrealist movement as a whole. Other definitions of aesthetics, as found in Bredin and Santoro-Brienza (2000) and Osborne (1970), also emphasize the close relationships between aesthetics and art. In summary, aesthetics provides a philosophical foundation for art in theory and practice.
While the previous discussion provides close connections between aesthetics and art, the term art has yet to be defined. There is a huge literature base for those wishing to define what art is; however, we will refer to Dorn’s overview (1999) in which he characterizes art in two dimensions. First, philosophically, art can be defined as an idea, form, or language. Second, psychologically, one can define art with top-down and bottom-up conceptions. Art may also be characterized in terms of alternative perspectives, which tend to be highly correlated with specific historical and cultural contexts. Adams (1996) and Free-land (2001) take a more categorical approach to art theory. For example, Adams emphasizes the following contemporary interpretations: formalism, iconography, Marxism, feminism, biography, semiotics, and psychoanalysis. In terms of art practice, Wilson (2002) presents a large number of areas, examples, and contemporary issues that affect the artist. Edwards (1986) and Edmonds and Candy (2002) advocate a pragmatic role for art, seated in creativity.

**Computing**

While attempting to define aesthetics and art can provoke numerous debates, defining computing may be a little easier. Within the academy, computing is referred to by an assortment of names such as computer science, computer and information science, and computer engineering. Each of these subareas may have a slightly different strategy, but we will associate computing with computer science without sacrificing clarity or scope. Computer science incorporates a large number of areas, some of which are evolving fairly rapidly. In general, the Association for Computing Machinery (ACM) and the IEEE Computer Society (IEEE-CS) have numerous special interest groups and technical committees that give us a handle on the breadth of the discipline. Subareas include discrete mathematics, theory of computing, programming languages, data structures, artificial intelligence, computer–human interaction (also known as human–computer interaction or HCI), operating systems, computer graphics, computer simulation, and computer vision. When we speak of aesthetic computing, we therefore apply aesthetics to one or more of these subareas.

Recently, Denning (2003) suggests a new high-level taxonomy based on application domains, core technologies, design principles, and computing mechanics. While on the subject of computing, it is important to stress the relationship between mathematics and computing. Computer science is founded on core elements of discrete mathematics; thus, we can view aesthetic computing as encompassing a number of mathematical concepts, especially areas involving formal grammar, language notation, geometry, and topology. Discrete mathematics forms the early core of most computer science curricula, along with
the algebraic extension to automata theory, which is generally studied in one's senior year at university. The importance of mathematics to computing cannot be overemphasized; it establishes the formal infrastructure in which mathematical concepts and abstractions can be related to basic computing concepts. Thus, much of aesthetic computing corresponds naturally with mathematical formalism.

**Aesthetic Computing: An Overview**

We are now in a position to combine two words *aesthetic* and *computing*. We define aesthetic computing as *the application of the theory and practice of art to the field of computing*. While this definition lacks the nuances and scope of Kelly’s earlier definition of aesthetics, it defines it more concisely for our purposes. Aesthetic computing relates to the following sorts of sample activities: (1) representing programs and data structures with customized, culturally specific notations; (2) incorporating artistic methods in typically computing-intensive activities, such as scientific visualization; (3) improving the emotional and cultural level of interaction with the computer.

Generally, aesthetic computing involves one of two types of aesthetics applications: *analysis* and *synthesis*. Analytic applications tend to evaluate artifacts of computing and mathematics from the perspective of classical aesthetic qualities such as mimesis, symmetry, parsimony, and beauty. Synthetic applications tend to employ aesthetics as a means of representation of the artifacts. The word “representation” is broadly defined to encompass the concepts of interaction and interface, rather than simply static presentation.

Aesthetics and computing are therefore rich in both practical and theoretical taxonomy, categories, and encyclopedic knowledge. One aspect of aesthetics that may at first seem tangential is considered central to aesthetic computing: *plurality* (Goodman 1978). Most references to art cover aesthetics from a multitude of cultural aspects, genres, and historical episodes. Plurality therefore appears to be a critical component of aesthetics as it applies to computing, lest we imagine that only traditional aesthetics associated with mathematics (Plotnitsky 1998)—parsimony, symmetry, and so forth—are relevant. In fact, one of the goals of aesthetic computing is to facilitate the expanding role of aesthetics in mathematics, and by extension, computing. This plurality must encompass both body and mind, the material as well as the mental. This suggests aesthetic *diversity* (Fishwick 2002a), and perhaps that more traditional aesthetics of mathematics and computing are subsets of those found in art (e.g., minimalism, symmetry, the harmony of the golden ratio in architecture).

Reviewing the numerous historical approaches to art, and the contemporary categories for facilitating critiques, one generalizes about aesthetics’ concern with cultural perspective—that is, the idea that an object can be viewed and considered from many
angles, through numerous lenses. This multiperspectivism is an important concept in aesthetic computing, serving as a bridge between the two areas. Certainly, art has the potential to create new ways of looking, listening, and touching things that are relevant to computing: interfaces, programs, data, and models. The pluralistic heart of aesthetics encourages multiple representations. Computing supports plurality in many of its sub-disciplines, but in aesthetics and art practice, it is virtually a defining criterion.

The architectural diagram for aesthetic computing shown in figure 1.1 defines it as a unique area. In the hourglass-shaped figure, imaginary sand particles flow from top to bottom “spheres.” The particles are modulated by a “Subject/Medium” filter. The diagram allows us to discuss a variety of approaches, beginning with a computing discipline identified by one of its objects (i.e., an artifact), its theory, or its practice (i.e., how the discipline proceeds within the computing field).

The “Subject/Medium” filter suggests that we are not using the computing discipline strictly to generate tools, but rather to provide a raw medium or the subject material for art.
Various media theorists (Manovich 2001; Coyne 1999) have discussed the concept of materialism at length; however, we use the word in its dictionary definition of *embodiment*, in contrast to *mind*. Thus, virtual reality, as discussed within the art literature (Grau 2003), is materialistic because it is consistent with embodiment and immersion in an enhanced sensory experience, regardless of whether this experience is *real* or *illusory*. Mental constructs, on the other hand, are nonsensory and so have no material existence. Continuing HCI and visualization research extends such materialistic qualities as presence, engagement, and immersion which facilitate human sensory connection to otherwise *invisible information*, or information that has minimal sensory qualities.

Returning to the filter in figure 1.1, when a program is used for its “tool-worthiness,” there is little or no reflection on the essence of computing (i.e., the internals of the program or data structures, their underlying mathematical structures) or the practices of computing. However, art created using the medium of programming (i.e., as in the emerging area of *software art*) involves greater reflection and emphasis on the computing (i.e., the programming, as a subarea of computing, in this case). Likewise, when artistic approaches or styles are used in representation, the elements of computing are treated as the subject material—the focal point of the artwork. An important part of the figure diagram is the flow terminus in the bottom sphere; the result may be usable or unusable in the strict sense of performance-based interface usability (Nielsen 1993). It could be seen as art to be displayed or an interface to be used, or as some combination of these two. The word “usability,” in a more general sense, can be quite complex; for example, improving a user’s emotional state is also a valid use (Picard 1997; Jordan 2000; Brave and Nass 2002; Norman 2004). The concept of *use* can also extend beyond human performance.

Some examples are in order to understand this flow from top to bottom in figure 1.1. First, let’s refer to a recent paper written by four coauthors (Fishwick et al. 2004), two computer scientists, one interaction designer (HCI), and one artist. Other examples will be delineated as subsequent items.

The two computer scientists (Diehl and Fishwick) apply aesthetics to the *representation* of formal structures in computing, such as computer programs and mathematical models. Thus, artifacts at the top of figure 1.1 flow through to the bottom sphere of aesthetic influence. The resulting artifacts are meant to be usable, and the focus is on representing the computing artifacts as *subject material* for art. Prophet (the artist) works closely in a team that includes, among others, a computer scientist whose expertise is scientific visualization. Her involvement in the *Cell project* stems from affecting the *practice* of computing. While this work was aimed at producing a usable visualization product, she and her collaborators will produce subsequent artifacts relevant to their individual disciplines. Löwgren, the interaction design specialist, enumerated several key qualities HCI designs...
must have to address the aesthetic requirements of future interfaces, including pliability, fluency, and seductivity.

Focusing on subject material for art, the area of “software art” recently highlighted within the 2003 Ars Electronica conference (ARS 2003) used computer code as raw material (i.e., a medium) for art. The Processing language (Processing) developed by Fry and Reas is a good example of this activity within the art world. Based on Java, Processing is a language oriented toward designers and artists. Interestingly, some of the programming examples at the Processing web site stretch the boundaries between surfacing computing artifacts as medium and subject. For example, some Processing Java applets such as distance_2d, directly represent and surface underlying computational structures. In distance_2d, the essence of what it means to be a matrix is surfaced, making the computing artifact (i.e., a matrix) the subject material of the piece.

Since the theory of computing is founded on mathematics, the architecture in figure 1.1 provides room for representing mathematical structures through aesthetic filters. The focus is usually on representing the solution space for mathematical structures (i.e., manifolds, surfaces, tessellations) (Emmer 1993); however, other visualizations based on problem spaces (i.e., representing the notation) are also possible (Fishwick 2002a). Leyton (Leyton 2001) provides a group-theoretic approach for generating art and music, whereas Ferguson is a hybrid artist-mathematician who specializes in building mathematical artifacts in “stone and bronze” (Ferguson and Ferguson 1994). Lakoff and Núñez (2000)—though not explicitly indicating a role for art—provide strong evidence that metaphor lies at the root of mathematics. Extending this argument, if art plays a key role in embodying metaphor, then aesthetics and art should play increasingly significant roles in all aspects of mathematics from its cognitive roots to its material notation.

The use of the word “aesthetics” related to computing deserves some discussion. Recognizing that the core specifications for computing theory are mathematical, we note that Hadamard (1945) introduced and documented the psychology inherent to mathematics. The classic Platonic definition of mathematical aesthetics describes mental pleasures associated with specifying theorems and deriving proofs. More generally, the mathematician’s aesthetics involves concepts such as invariance, symmetry, parsimony, proportion, and harmony. Hadamard’s studies of famous mathematicians points out that the vast majority of them perform mental visualizations both in posing a problem and in solving it. Describing the proof involving an infinity of primes, Hadamard (1945, p. 75) refers to “strange and cloudy imagery” and relative physical distances. In one of the study’s better known quotes (Hadamard 1945, Appendix II), Einstein wrote to Hadamard that “The words or the language, as they are written or spoken, do not seem to play any role in my mechanism of thought. The psychical entities which seem to serve as elements in thought are
certain signs and more or less clear images which can be ‘voluntarily’ reproduced and combined.’’

One wonders, after reading Hadamard’s treatise, whether these mental visualizations are a sheer economic necessity based on the relative expense of exercising other types of notations, or whether mathematics is forever constrained to the mind. The mind is the fount for mathematics, but it serves the same function for all human activity. Descartes’ binding of algebra and geometry and contemporary “math-art” activities (Emmer 1993; BRIDGES) provide interesting revelations on aesthetics, and mathematics. A strong foundation has been laid for ongoing research in the field. In the domain of computer programming, Petre and Blackwell (1999) document a significant number of visual and aural effects imagined by programmers as they perform their craft. One programmer describes such effects moving “in my head . . . like dancing symbols. . . . I can see the strings [of symbols] assemble and transform, like luminous characters suspended behind my eyelids. . . .”

Petre and Blackwell caution against leaping to prematurely positive conclusions about the benefits of visual programming for all situations. However, any empirical test of the efficacy of visual paradigms seems to be necessarily bound by technological limitations of graphics, sound, and interaction hardware devices. Until these environments become commonplace, affordable, and efficient, nontext–based representation will always be biased by less than adequate technology. Moreover, all new interface modalities suffer from a cultural bias against adopting new interfaces when existing modalities are familiar and still function. What would empirical studies show if we had an ideal user-friendly environment like the immersive holodeck of “Star Trek” (Paramount)? Unfortunately, further design, engineering, and empirical studies are needed to answer this question.

Early on in the history of computing, Donald Knuth showed himself to be a strong advocate of aesthetics in programming (Knuth 1997; 2003). As Knuth points out in his discussion of Metafont (Knuth 1986), which underlies his TeX typesetting system, “Type design can be hazardous to your other interests. Once you get hooked, you will develop intense feelings about letterforms.” More generally, Knuth directly addresses the issue of aesthetics as more than purely cognitive, beyond the Platonic mental ideals. A textual section of a computer program has both denotative and connotative signifiers, and it is easy to imagine the program aligning with the goals of art, stretching the traditional boundaries of what may be considered a usable computer program representation. Nake (1974) explores the idea of aesthetics as information processing. More recently, Gelernter (Gelernter 1998a; 1998b) has provided significant justification for aesthetics in computing. Defending the vital role of attributes such as emotion, style, and aesthetics in all aspects of computing, Gelernter illustrates with a case study of how the Macintosh inter-
face and style has revolutionized the industry, though it was first viewed as strictly for novices.

Mathematics has historically emphasized \textit{solution} spaces, and not notational spaces (i.e., for framing \textit{problems}), but visualization in computer science is playing an increasing role in visualizing structures and data (Stasko et al. 1998; Card et al. 1999; Diehl 2001).

**The Novelty of Aesthetic Computing**

Every discipline should have to justify its existence or, at the very least, include formal critiques to put its subject matter in the proper context. Is aesthetic computing a new field, or simply rehashed old material? It has not yet survived the test of time or the rigors of comprehensive criticism; to date, only a single workshop has been held on the subject.

Aesthetic computing combines two key areas: art and computing. One might object that aesthetic computing appears to be about design, and that art and design have different agendas. Their goals merit alternative approaches and philosophies, but aesthetic computing can have either purely artistic or design goals, depending on the practitioners involved. Its goals may also produce usable or nonusable results, if we adopt a strict definition of usability. Traditional design and illustration research involving computing tends to be rather sparse for several reasons. The design of web pages and operating system interfaces (i.e., the desktop metaphor) is a small subset within the field of computing. Significantly greater \textit{diversity and depth} of aesthetics needs to be applied to all areas of computing, from notations to formal structures. We must analyze the subarea lists of the Association for Computing Machinery (ACM) and the Institute for Electrical and Electronics, Engineers Computer Society (IEEE-CS), which define the breadth of computing. A recent National Academy study on computing (CSTB 2003, chap. 4) recommends considering the effects of the arts within the computing field.

The richness of work in digital design and arts, or the \textit{information arts} (Wilson 2002), suggests relations to aesthetic computing, but the goals of each area are quite different. For one thing, aesthetic computing is not meant to be an all-encompassing “bridging term” between aesthetics and computing. It is about surfacing the core components of computer science, its areas of study, and its methodologies. One of its core goals is to modify computer science through the catalysis of aesthetics. This is not the same as using artificial intelligence to create designs or algorithms to effect new forms of artificial life, as exciting as these enterprises may be. The goal of work done to date in applying digital methods to art is the \textit{converse} goal to that of Aesthetic Computing.

Visualization, specifically scientific, information, and software visualization, lacks the sort of personalization or customization aesthetic computing makes potentially viable.
These areas play vital roles in combining aesthetics and computing, but to date, designs have tended to be visually minimalistic and oriented toward a generic concept of user.

The assumed roles of aesthetics as applied to computing are too limiting. First, we are not limited to traditional concepts such as symmetry and harmony when defining computing aesthetics. Instead, we are free to choose, say, the aesthetics of a particular artist or art movement. Second, formal constructs within computing are sometimes bypassed in considering aesthetics. One can interpret “aesthetic algorithm” (Nadin 1991), for example, in several ways—assuming the algorithm has aesthetics traditionally associated with mathematics (i.e., as in the first example), or referring to the artistic phenomena resulting from executing the algorithm code. But these interpretations differ from one in which the algorithm itself has an artistic manifestation. The structure and representation of algorithms are part of computing, whereas the aesthetics of the algorithm’s execution is more closely aligned with the visual arts.

Exploring design, art, and computing, we hope to carve a niche for aesthetic computing enriching these other disciplines in the process.

**Applying Aesthetics: The Artistic Influence**

For as long as mathematics and technology have existed, artists have used them for their own purposes, applying these tools to create new works of art. Examples include the use of Euclidean geometry in perspective drawing and painting (Kemp 1992), Vermeer’s postulated use of the camera obscura for his paintings (Steadman 2002), the influence of multi-dimensional space and non-Euclidean geometry on the art of Duchamp (Henderson 1983; Robbin 1992; Schlain 1993) and Escher (Schattschneider 2004), mechanization and mass production trends in modern art, and more recently, computing trends (i.e., artificial life, genetic algorithms, chaos theory) on new media (Wilson 2002). The literature is rife with examples of artists applying mathematics, technology, and most recently, computing to the creation of art.

In studying the converse situation, however, we must ask why no corresponding history of artistic practice affecting computing exists. Also, what aspects of aesthetics can be applied? On the whole, it is clear that both mathematicians and computer scientists have been deeply affected by aesthetic qualities such as beauty, symmetry, and abstraction. However, one does not normally see the same level of artistic theory and practice applied mathematics and computing. Why is this? One hypothesis is that, with advanced computing technologies, we are only now beginning to see the effect of art on computing. Let’s begin with what it means to apply aesthetics to computing, dividing aesthetics into three broad groups: modality, quality, and culture.
Modality refers to the ways in which we interface and interact with objects. Art practices encourage things like pluralism in representation (see Deem 1993 for an unusually precise example), interaction, dynamism, and materiality (i.e., embodiment). One might ascribe these concepts to fields such as human–computer interaction, when in actuality, these are part and parcel of the arts. Exploring one or more modalities in the interface is what artists do, therefore, any aspect of computing that stresses this approach owes a significant debt to the arts. However, fields such as HCI, ubiquitous computing, augmented reality, virtual reality, and tangible computing are made possible only by rapid advances in computer-related technology. We have had to wait for the technology to become available to leverage the arts. This same requirement for advanced technology to apply art to computing is present for the next group.

Culture in the arts is manifested in many ways—specific artists, art movements, and genres. Genres range from impressionism to romanticism and modernism to feminism and postmodernism. Aspects of these movements, sensory styles, or their philosophies can be applied within computing. Modalities for such representation is evolving slowly due to economic constraints. Subjectivism is expensive; a single standardized objective interface is cheaper. Multiple representations are more costly; however, technologies such as XML (e.g., with its pronounced content-presentation capability) and mass customization are making it possible to apply multiple styles and representations to computing. As the subjectivist hallmark of the arts becomes less expensive, representations in computing will change.

Quality refers to aesthetics before Kant’s blending of mind and body, that is, general aesthetic qualities. These are not so much applied from as made consistent with some of the arts. Qualities such as mimesis, symmetry, complexity, parsimony, minimalism, and beauty, for example, could be said to be present in the arts.

To apply aesthetics to computing, we draw on a long history of the arts in which modality, culture, and quality have played significant roles. Only fairly recently have we begun to think about concrete ways in which the arts play an increasingly critical role in computing.

Mathematical Modeling: Research at the University of Florida

At the University of Florida, our primary emphasis has been on artifacts in computing best termed mathematical models, to represent the dynamics of systems (Fishwick 1995). Dynamical models represent how system attributes change over time. The “aesthetic filter” is applied, noting that our software framework emphasizes the ability to easily change model representations, thus enabling customization and culturally diverse notations. Thus, we take the synthetic approach, described earlier, to applying aesthetics to computing.
Our work can be seen as being a type of three-dimensional (3D) design for these notations. Customization appeals directly to the concept of plurality discussed earlier in the context of aesthetics. A simple example is illustrated and described in (Fishwick et al. 2004).

In applying aesthetics to computing, we need to confine ourselves to some aspect of computing, or one of its subfields such as automata theory, HCI, visualization, or discrete structures, to name a few. Potentially, any of the subfields can be enhanced with a more thorough investigation of aesthetic application. For the RUBE Project (Kim et al. 2002; Hopkins and Fishwick 2001; 2003), we have focused primarily on representations, informed through an artistic sensibility, in mathematics and computing notation, from the notation of algebraic and differential equations to that of program and data structures. Our basic approach is to build a system that allows construction of a multiplicity of notations to reveal the same underlying formalism in numerous ways. Not only do different people and cultural entities enjoy working with different metaphors, but the same person or group can benefit from exposure to diverse presentations.

The RUBE software system we have constructed allows us to apply different representations to a select number of formal dynamic model specifications. Using RUBE, it is possible to change the way formal models look and sound. By formal models, we refer to a large class of models used to specify systems incorporating time for analysis and simulation: finite state machines, Petri networks, Markov models, queuing models, System Dynamics graphs, as well as ordinary and partial differential equations. RUBE uses XML (eXtensible Markup Language), which separates content from presentation while allowing arbitrary style-defined bindings to unite them. In XML parlance, content refers to an abstract specification defined as a document tree, and presentation refers to how the tree is presented to the user, its look and sound. Thus, using RUBE and guided by the XML philosophy, one may specify an equation, but then present it as linear text, a network, or a 3D structure. Choosing which presentation to employ can be guided by XML style sheets and their associated transformations. These transformations bind the presentation to the content.

Based on open source software, RUBE’s architecture begins with authoring toolkits: SodiPodi for 2D vector drawing, and Blender for 3D modeling. Let’s consider the 3D pipeline beginning with Blender. Creating a 3D model in Blender, the artist then uses a Python scripting interface that allows attributions to be made regarding semantics. For example, one might identify an object as a state or a function. After the semantic assignment, the artist creates an X3D (eXtensible 3D) file for the presentation, and a special XML file for specifying the formal model. After some XML transformations, this XML file is translated into Javascript or Java, to reincorporate it into the X3D/VRML file, resulting
in an interactive Virtual Reality Modeling Language (VRML) world. The 2D transformations are similar, except that scalable vector graphics (SVG) are used for presentation.

Let’s begin with a formal definition of a finite state machine (FSM) $M$ (Fishwick 1995). These machines have states interconnected through transitions that are activated by an input to the machine of a particular value. $M$ is formally defined in traditional notation as

$$M = \langle Q, I, O, \delta, \lambda \rangle$$

$$Q = \{S_1, S_2, S_3\}, \quad \delta : Q \times I \rightarrow Q$$

$$\delta(S_1, 0) = S_1; \quad \delta(S_2, 0) = S_2; \quad \delta(S_3, 0) = S_3;$$

$$\delta(S_1, 1) = S_2; \quad \delta(S_2, 1) = S_3; \quad \delta(S_3, 1) = S_2$$

$$I = \{0, 1\}, \quad \lambda : Q \rightarrow O$$

This text might seem to be the formal specification for $M$, but it is actually one of many ways to look at the underlying formalism encoded in XML. It is one type of presentation among many. In general, all presentations require additional natural language semantics if we are to make sense of them. $Q$ is the state set for $M$; $I$ the input set, $O$ the output set, $\delta$ the transition function from one state to another, and $\lambda$ the output function. Figure 1.2 illustrates our second presentation for the FSM. The iconic presentation of a circle for the $S_2$ state encodes the concept of a boundary and that which it encompasses. This iconicity

![Figure 1.2](image-url)
is similar to that noted by Shin (2002) in her discussion of Peirce’s logic diagrams. That is, the graphical depiction of S2 is consistent with the underlying metaphors of set theory, whereas the purely textual presentation does not capture these metaphors. Moreover, as represented on a noninteractive medium such as paper, figure 1.2 is incomplete since the additional information (such as the input values needed to effect a change of state) encoded in the text representation is equally present during interaction with the figure. The arrows convey the notion of transition from one state to another. The figure’s metaphors dramatically improve our understanding of the machine semantics, leading to the possibility that using presentations with alternative aesthetics might strengthen these metaphors’ impact. The underlying assumption is that material aspects of levels of representation are based largely on what is available, affordable, and materially efficient.

Consider figure 1.3 as a representation that has only recently become possible through computer graphics and the ability to employ 3D components. The metaphor, encouraged by the iconicity of diagrams (Shin 2002), of the circle as a boundary has been replaced by a set of tanks (on the left), and small gazebolike structures (on the right). The arrows in figure 1.2 are replaced by either a pipe filled with water (left) or a red-clothed woman walking from one state to another along a lamp-lit walkway (right). These examples provide different metaphors for understanding the formal structure. Even with something as basic as a circle in iconic mapping, one can imagine beyond what’s “inside the circle” and conceptualize moving from one circle to another. The 3D metaphors strengthen this feeling of immersion, more clearly envisioning being inside the gazebo, like the woman, or watching fluid move from one position (i.e., state) to another.

A host of philosophical issues come into play here. Isn’t there a need to enforce visual minimalism within this sort of structure? What cultural barriers might prevent the adoption of models like figure 1.3 for science and engineering? Regarding minimalism, we should note that is quite possible to preserve abstraction without requiring visual minimalism. Within the context of the art community, this can be seen when we compare and contrast the genres of abstract expressionism and surrealism. Both genres contain a wide variety of works that employ symbolism, iconography, and the richness of semiotics even though the visual presentations are strikingly different. Consequently, abstraction as a one-to-many mapping has nothing to do with how we visually or aurally represent notations; the circle in figure 1.2 and the gazebo in figure 1.3 are both at the same level of abstraction regarding notating a state. Both require the same number of bits from an information theoretic perspective of recording that the entity is a state, although the bits to record the alternative presentations are different. Deriving the idea of an abstract state in an FSM, for example, need not imply that the state be presented visually in a minimalist
Figure 1.3  Two 3D models representing the diagram in figure 1.2: (a) a set of three transparent, cylindrical tanks that transfer water (representing a change in state), and (b) three gazebos with adjoining walkways, and an agent walking from state to state. Both FSMs were constructed with the Virtual Reality Modeling Language (VRML).
Figure 1.4  Four views of a network of nodes, with feedback, for modeling a banded waveguide physical model for sound (Joella Walz). Modeled with Maya.
fashion. The key objective is to strengthen the metaphor underlying what it means to be a state, and the corresponding metaphorical elements of boundary. The abstraction afforded by states suggests a one-to-many mapping in which one FSM may map to a large number of different applications.

The second question about cultural barriers may be at the heart of the aesthetic computing challenge. Educated with minimalist figures and text, computer scientists may be shocked to realize our representations for formal objects are not as constrained as originally thought. Until the era of computer graphics and fast computers, we had little need to
inquire about what initially appeared to be exotic ways to encode formal knowledge. This is a challenge not only for computer scientists, however, but also for artists, who should be encouraged to consider the computer, and computing practices, as *subject material* as well as raw material. This suggestion may strike some artists as a modernist era agenda; however, as a tool or a subject, the computer with its mathematical foundation creates significantly higher complexity than paint, palette knife, or chisel ever could.

The following two examples were created in the author’s aesthetic computing class (AC) at the University of Florida. The class includes both artists and computer scientists who work individually and on group projects to represent mathematical models found in mathematics, program and data structures, and computer simulation.

Figure 1.4, based on a virtual model created by an artist (Joella Walz), represents a functional feedback data flow network, whose purpose is to physically model sound. One type of modeled instrument, with the appropriate parameter settings, is of a Tibetan Prayer Bowl, which gives off a resonant bell sound.

Figure 1.5 shows the equivalent 2D diagram (Essl 2002) for the structure shown in figure 1.4. The first thing one notices is that the diagram has more complete information, since it is specifically made for print media, whereas the structure in figure 1.4 requires a highly interactive environment to determine which nodes are the delays, the band pass filters, and the primary interaction node. For figure 1.4 to be as useful as figure 1.5, barring issues of aesthetics and customization, the necessary *interaction environment* must be in place to easily determine node roles and connections.

Figure 1.7, created by a computer scientist (John Campbell), shows a physical model of the Taylor series (figure 1.6), found in most introductory calculus books. The Taylor series is an infinite sum resulting in a polynomial approximation to function $F(x)$.
Summary

As in any new field, many issues are bound to stand out, and they often cluster around a discipline. Each discipline involved with aesthetic computing is assumed to have its own interests at stake. For art, the issues will likely center on the need to do new, contemporary work rather than retread what has already been tried. The same can be said for design and computing. For computing, art is seen in all of its history, not just its current leading edge; thus, the application of aesthetics to mathematics and computing can take on a wide range of art genres. Artists and computer scientists are similar, then, in that they select from the whole history of the other area, while focusing mainly on the newer research in their own field. While we may promote their interconnection, each field must explore its own potential, according to its own concept of advancement.

To progress, we need to take up the key challenges in each field. For art, the reflection on computing naturally has a utilitarian component—computers are used to achieve specific results. This is not to suggest that artists working in aesthetic computing must always yield to utility, but the goal should be to fully explore the range of utility from “useless” art to art with a specific technological purpose. The very word “use” is fraught with complexity, of course, since one could argue that a nonuseful work of art showing the attributes of a matrix actually has an aesthetic or educational use. Usability should not be limited to the very strict definition of performing a task in a robotic fashion. If the challenge for art is learning to live with utility as almost a revisiting of the Greek concept of techné, the challenge for computing is to recognize that the interface should be as much about quality as it is about quantitative performance. This quality includes attributes such as emotion and aesthetics, and it reflects the fairly new wave of human-centered activities saturating the computing discipline. Computing is not just about mental formalisms and algorithmic complexity; it is also about how to more effectively interface along the lines of tangible, augmented, and ubiquitous computing. Computing professionals need to pay more attention to these areas and observe the critical role art plays in promoting novel representational techniques.

Usability is further complicated by the fact that the newer generations of computer interfaces are expensive and more difficult to operate. All figures in this text serve as static
Figure 1.7  Two views of a metal, Plexiglas, and wood representation of the Taylor series expansion (John Campbell).
presentations of potentially interactive, immersive, and engaging artifacts. Unfortunately, because media such as paper do not capture these qualities, information-dense diagrams appear to be the only valid representations. Time, effort, and progress in both art and computing are needed to engender the sort of environment that aesthetic computing promises. Along the way, artists and computer scientists should design new environments, evaluate them, and interact with each other.

Acknowledgments
The author would like to thank his graduate students in the RUBE Project (Minho Park, Jinho Lee, and Hyunju Shim), as well as all students of the aesthetic computing class, and with special thanks to Joella Walz and John Campbell for their final virtual and physical projects. Also, special thanks to Mihai Nadin for his constructive comments on an earlier draft of this chapter. This work would not have been possible without grants from the National Science Foundation (EIA-0119532) and the Air Force Research Laboratory (F30602-01-1-05920119532).

References


