Finishing Touches: Adaptive graphic design that leverages human creativity

William Kelly Norton Jr.

Bachelor of Electrical Engineering,
Georgia Institute of Technology
June 1997

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning,
in partial fulfillment of the requirements for the degree of
Master of Science in Media Arts and Sciences at the
Massachusetts Institute of Technology
June 2006

[illegible]

Author: William Kelly Norton Jr.
Program in Media Arts and Sciences
May 30, 2006

[illegible]

Certified by: John Maeda
Associate Professor of Design and Computation
E. Rudge and Nancy Allen Professor of Media Arts and Sciences

[illegible]

Accepted by: Dr. Andrew B. Lippman
Chair, Departmental Committee on Graduate Studies
Program in Media Arts and Sciences

Finishing Touches: Adaptive graphic design that leverages human creativity

William Kelly Norton Jr.

[illegible]

Thesis Reader: Walter Bender
Senior Research Scientist, MIT Media Laboratory
President, Software and content, One Laptop per Child

Finishing Touches: Adaptive graphic design that leverages human creativity

William Kelly Norton Jr.

[illegible]

Thesis Reader: Henry Holtzman
Chief Knowledge Officer
MIT Media Laboratory

Finishing Touches: Adaptive graphic design that leverages human creativity

William Kelly Norton Jr.

Submitted to the Program in Media Arts and Sciences, School of Architecture and Planning, on May 30, 2006, in partial fulfillment of the requirements for the degree of Master of Science in Media Arts and Sciences

Abstract

This thesis proposes the investigation of a distributed architecture to facilitate automatic adaptation of graphical representation in digital documents to constraints imposed by the presentation device. The goal of the system is to select the most effective graphical solution from among a set of such solutions given the characteristics of the viewing environment (i.e. resolution, screen area, aspect ratio, color depth) or to determine that all available solutions lie below a specified effectiveness threshold in order to prompt manual, human redesign for that particular case. Such a system demonstrates a new hybrid architecture for problem-solving systems that is able to leverage the systematic protocols of pure computing systems with the creative intelligence of human invention.

Thesis Supervisor: John Maeda
Associate Professor of Design and Computation
E. Rudge and Nancy Allen Professor of Media Arts and Sciences

Acknowledgments

While my name is plastered on the front page of this thesis, it would not be possible without the combined efforts of a community of good, smart friends. Before I set out in search of my next adventure and source of free coffee, I would like to thank the following people for their lasting contributions, both to this thesis and to my life.

To my advisor, John Maeda; thank you for being a mentor, an inspiration, a creative spirit, a caring human, an informed Tokyo travel guide, and a superb friend. In all of these roles you are much more than "good enough." I came to you in search of intellect, but you taught me so much more about life.

To my thesis readers Walter Bender and Henry Holtzman; thank you for sharing your wisdom and guidance. You sacrificed both weekend and holiday to assist me and, for that, I am truly grateful.

To my teammates and life-long friends in the PLW; Burak, Noah, Carlos, Marc, Amber, Annie, Brent, Isha, Danny, and Kate. If ever there was a Dream Team of creative and intellectual talent, it was right here. We have accomplished many things together and you laughed at all my crazy stories. Thank you.

To Linda Peterson and Pat Solakoff, thank you for always showing me kindness when a lecture on timeliness was probably more appropriate. Linda, I am convinced you are some sort of high priestess of patience and calm. Without your help and understanding, I would not have reached the end.

To my wife, Stephanie; thank you for not killing me. It would have been understandable—living in a tiny apartment, with half our known possessions in storage, trying to pay the rent on a fraction our previous salary, all while teaching Trey the importance of brushing thoroughly. You have sacrificed as I pursued my dreams. Though I have a lifetime to do it, repaying you will be impossible. With every letter I typed in this thesis, I meant—I love you.

To my kids, Trey and Zoe; thank you for giving me an excuse to dress as the Red Power Ranger and run around the house. You both inspire me in the most surprising ways. You teach me more in an afternoon than I could learn elsewhere in an entire year. Trey, you are driven by an energy source that confounds scientists. Watching you dance around the living room with infinite optimism kept me going throughout the thesis process. I love you both.

To Stephanie, Trey and Zoe. You are my life.

Contents

Chapter 1

Introduction

In print formats the design of information graphics is a relatively static exercise. Once the design's veracity has been confirmed on the final paper stock, the designer can be reasonably sure about what each viewer will experience when he/she encounters the design. In the world of digital design, there is more uncertainty. Designers must set their designs afloat on an sea of uncontrollable variables, accounting for variance in unknowns like screen size, color depth, and graphics capabilities of the device of each reader. This uncertainly has been most pronounced on the web, where designers go to great lengths to establish a quality presentation targeted at an ideal environment for viewing but then have to ensure graceful degradation in other environments. Use of flexible style rules, like those provided with Cascading Style Sheets [33], helps considerably, but there is a limit to the range of adaptation that can be supported by a single document. At the extremes, for instance when viewers are using very large or very small screens, even these solutions are insufficient as a completely different data representation is required.

For media authors and distributors, the inability to adapt to different viewing environments imposes limitations on their ability to adapt to changes in future demand. One of the key questions for that particular industry is which of the many devices for experiencing content will users eventually select. Each of the contenders

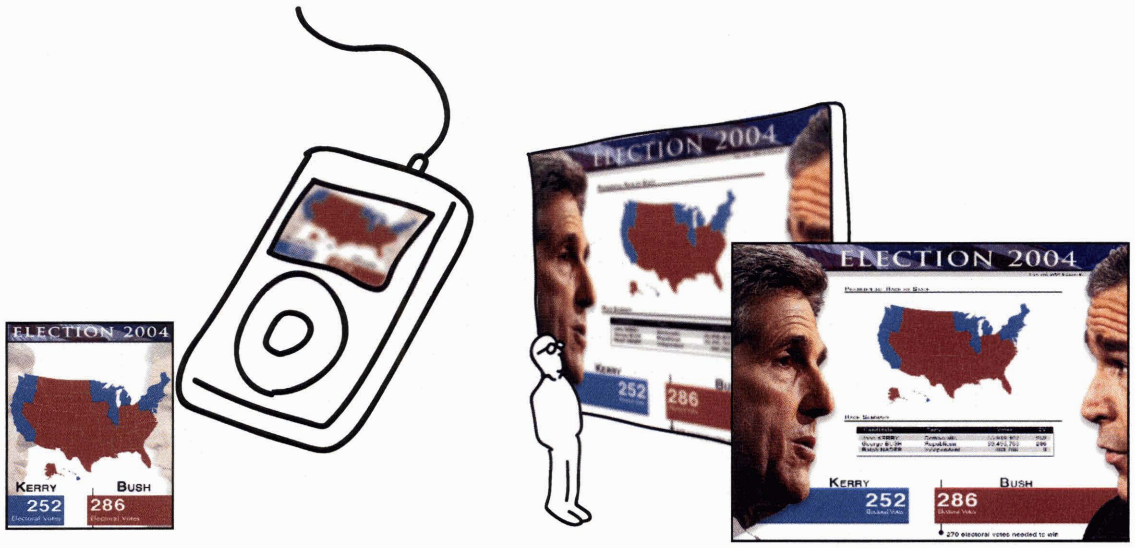


Figure 1-1: The ability to adapt to enormous changes in viewing environment is key to the future of media distribution

introduce their own set of limitations, from the enormous resolution of high definition displays to the slowly changing capsules in e-ink. Providing content for each of the devices is an independent design challenge that limits the ability to adapt quickly to newly introduced technologies.

Additionally, as projects like those being carried out by the One Laptop per Child foundation [10] are able to bring laptop computers to children of developing nations, there will be a continued need to ensure that those machines are able to display content produced for many years to come. The $100 laptop target also introduces a set of constraints that are not likely to be considered by content authors from more developed countries. The presence of an adaptive layer is therefore crucial to ensure that these machines are able to effectively present content from many sources. What this thesis proposes is a system that is able to mediate between design produced for an ideal and the reality of an reader's viewing environment.

As an illustration of this point, consider the two graphs below (Figure 1-2) that depict the difference in average temperature between Atlanta and Boston for 2004. When there is enough screen real estate to accommodate the more detailed view, a

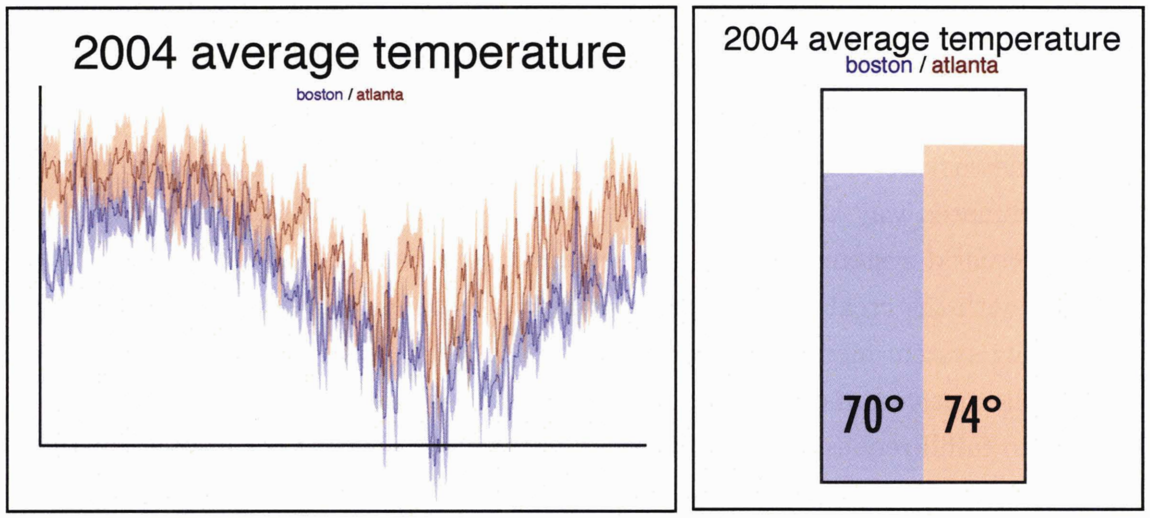


Figure 1-2: Information graphic depicting the average temperature in Boston and Atlanta in 2004 transformed to accomodate very different scales

designer may prefer this view as it gives an accounting of the variance of temperatures throughout the entire year. Yet moving this detailed graphic to devices with lower resolution may render the graphic far less effective as the feature size makes fine-grained comparison difficult. In that case, the designer may prefer to do more visual summary and instead present a graphic that is based on the aggregate of the data rather than each discrete value, as shown in the simplified bar graph on the left. This adaptation is not merely a restyling of a single document's content, but a full transformation from one document representation to another.

Automatic layout and arrangement of data presentation has actually been a rich research field [16, 17, 18, 26, 34, 37, 51], yet it has tended towards solutions that are computationally complex and effective only under a very narrow range of conditions. While it is an ideal that a designer would create a representative design example from which all other possibilities could be inferred, the reality is that human designers are often needed to readdress design decisions when the environment changes drastically. The medium is, after all, a substantial part of the message [39]. So while it is important that automated design presentation systems be able to synthesize design solutions over a range of changing variables, it is just as important for them to be able to determine when those solutions are inadequate.

The infrastructure proposed here abandons the quest to solve such problems in a fully automated way and introduces the concept of a hybrid human/computer system. Hybrid systems leverage the systematic formalism of automated computing systems with the creative intelligence of human problem solvers. In this case, the computing system mediates between the devices requesting content, a collection of design alternatives, and a community of designers-for-hire to adapt designs as needed to fulfill requests. The infrastructure proposed here is not only a practical system in that it provides for a distributed architecture for adapting designs effectively, but it also demonstrates the realization that automated solutions never provide a panacea. This is in line with one of the intentions of the Physical Language Workshop, to create computing infrastructures that are powerful from a traditional computational sense, but also able to leverage the creativity of human invention. Ideally, the system proposed for this thesis project will be intimately linked to a community of creative labor - humans willing to solve fine-grained design solutions as part of an economy of small scale projects [11].

Chapter 2

Background

2.1 Automated Design by Computers

Before there was a computer on the desk of almost every graphic designer, the knowledge of the field lived completely apart from the tools. The knowledge was contained in the living memories of its practitioners and within the pages of the books that adorned their shelves. Their tools were the french curves and t-squares that littered their workspaces. With the introduction of the computer to the field, the knowledge and tools of design began to merge [23]. Since that time, there have been many attempts to formalize the knowledge of the field in a way that permits the computer to become more than a passive implement in the creative process.

In design fields where aesthetic judgement is not one of the primary factors of effectiveness, automated solutions have become standard practice. For instance, in VLSI design, it is quite rare for humans to still play an active part in routing and laying out large sections in the design of integrated circuits [23]. The role of the computer as a graphic designer, on the other hand, has been quite limited. The fact that the graphical artifacts are designed to communicate an idea to humans means that the appropriate decisions require the kind of sophisticated model of human perception and understanding that is still only found in other humans. There are, however, certain areas of graphic design where it is possible to codify at least part of the design criteria. In these areas, including graphic representation of relational data and document layout, there has been significant work in automating much of the design process.

2.1.1 Automatic Layout of Design Documents

The areas of graphic design where the computer has made the most progress in automating the design process has been those where the graphical document exhibits an identifiable and regular structure that is, in some predictable way, related to the stylistic presentation. The process of spatial arrangement, or layout, is an essential part of the graphic design of information documents and generally introduces a predictable, recurring pattern throughout. This document is a prime example. Each of the various sections and sub sections are identified by the interplay of their corresponding headings. Those headings each use identical, more heavily weighted, type and are arranged at the same height above the body of text they represent. The process of creating this document, however, did not require each and every header to be hand placed at an exact pixel offset from the left of the page. That task was left to a typesetting application, which took all the text characterized as a header and applied the layout and style choices from a template. Below is a brief summary of some of the techniques for layout that have evolved in computer-aided design, desktop publishing, and user interface design.

Simple Techniques

There are two paradigms of automated layout that are used widely in software today. And while they do automated layout to an extent, both techniques require extensive human guidance. The first of these is the parameterized *geometry managers* (or *layout managers*) that are found in GUI tool kits such as Sun's Java Swing libraries [8], Microsoft's .NET Windows Forms [9], and Apple's Cocoa [5]. Using this paradigm, user interface programmers specify parameters to control the arrangement of a hierarchy of visual controls and their associated containers.

Clearly, the specialized knowledge required by the author in this case is very high. In fact, the programmer is essentially providing guidelines to the runtime indicating how to accommodate size changes in the user interface. Also, to ensure that the complexity does not affect response time of the interface, these guidelines are typically constructed only in terms of local relationships. Schemes like this one that rely only on local spatial alignment, typically are ill suited for graphic design documents as it makes global grids difficult to maintain [1]. Only in rare case are the guidelines able to adapt to drastic changes in the runtime environment, such as running the application on a portable device.

Another simple layout technique that is often found in word-processing, typesetting, and Web-based documents is based on the conventions of basic text flow. This technique takes advantage of the way textual documents are generally arranged to simplify the task of authoring. These tend to be highly sequential as the order in which items appear in the document determines their order in the presentation. Authors are usually given some degree of freedom in determining how certain objects interact in the text flow. For instance, images can float with either the text flowing in front, behind or around. Layouts of this type tend to leave a lot of details to the discretion of the software.

Constraint-Based Techniques

The vast majority of the research in the area of automated design has focused on constraint-based methods of representing the spatial relationships in a design [34]. Not only is this method sufficient for describing the complex spatial constraints that exist within a document, but it can also be extended to cover more abstract higher-level relations [34]. For instance, a spatial constraint may declare that a caption corresponding to an image should be placed below that image. More abstract constraint will declare higher-level meaning, like the fact that the caption references the image.

Expressing designs as sets of constraints is far less limiting than the methods listed

[1]Design along a global document grid is an essential principle of modern design [40]

above, but two significant problems arise. The first, which has received much attention and is able to leverage work in other areas of computer science, is the task of solving the system of constraints to produce a rendering of a design solution. The second is that designs codified as constraint systems do not fit naturally into the process of authoring. Constraint solvers, at this point, are considered a general tool in computer science. There has been a great deal of work some of which has been applied to the domain of automated design. A detailed description of the techniques that have evolved, however, will not be considered here.

The process of creating a design solution that is based on system of a constraints has also been the focus of several research projects. Approaches have ranged from requiring designers to express the constraints explicitly to automatically extracting implicit constraints in pre-existing designs. A brief summary of some of the most relevant work is included below.

The work by Wietzman and Wittenburg used the relational grammars [52] formalism to represent the hierarchical structures within documents [50, 51]. To aid the author in assembling the hierarchies of primitive objects (i.e. Text, Number, Image), the special authoring environment used an improver-based system to make suggestions to the author as it inferred the creation of potential composite objects. For instance, the close placement of a number, some text and an image might prompt the system to suggest to the user that these be combined to form a figure with an internal caption. The object inferences are formed according to a rule system where relationships among primitives indicate the potential for combination into a composite object. Once the user's input was structured into a more semantically meaningful set of objects, the process of translating those abstract constraints into spatial constraints could be done with consideration of the eventual output medium. Figure 2-3 demonstrates the system's ability to apply different sets of spatial constraints to vary the document's style. The mapping of primitives into composites and the subsequent translation into spatial constraints were controlled by domain specific rule systems that indicate both the make-up of composites and the spatial constraints to be used during the output process. An example of a rule used to generate the output in Figure 2-3 is shown below (Figure ??).

```
(Defrule (Make-Article The-Grammar)
  (0 Article)
  (1 Text)
  (2 Text   (Author-Of        2 1))
  (3 Text   (Description-Of  3 1))
  (4 Number (Page-Of         4 1))
  (5 Image  (Image-Of        5 1))
  :OUT
  ((right-of      1 5)
   (right-of      2 5)
   (right-of      3 5)
   (right-of      5 4)
   (top-aligned  1 5)
   (top-aligned  5 4)
   (spaced-below 2 1)
   (spaced-below 3 2)
   (set-font 1 10pt :bold)
   (set-font 2  8pt :italic)
   (set-font 3  8pt :plain)
   (set-font 4 10pt :plain)))
```

Figure 2-1: An example of a rule taken from Weitzman's relational grammar based system

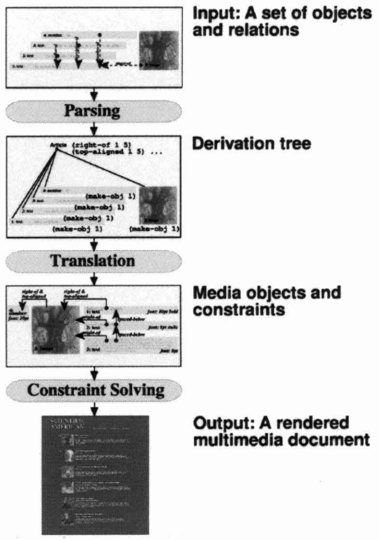

Input: A set of objects and relations
Parsing
Derivation tree
Translation
Media objects and constraints
Constraint Solving
Output: A rendered multimedia document

Figure 2-2: A process-level look at Weitzman's system. Objects and relationships are parsed into a tree structure which is then used to impose constraints on graphical objects.

2.1.2 Automatic Design of Relational Information Graphics

From many of the examples above, it is clear that structured input data is an important criteria in realizing effective automated design. Thus, the areas of graphic design where structured input data is more readily available have been the most heavily effected by automated processes. This is certainly true of graphics representing relational data (such as bar charts, scatter plots, and connected graphs). Work in this area has been able to leverage the familiar graphic language that has evolved in the charting of numerical data. Also, since the focus is primarily on giving the reader an understanding of the nature of the data, it is often acceptable to ignore aesthetic or emotional components associated with the design choices. With freedom from those constraints, a great deal of work has emerged in this area. While the following are not intended to be all-inclusive list of the work, they do provide a guided overview of the work in this area.

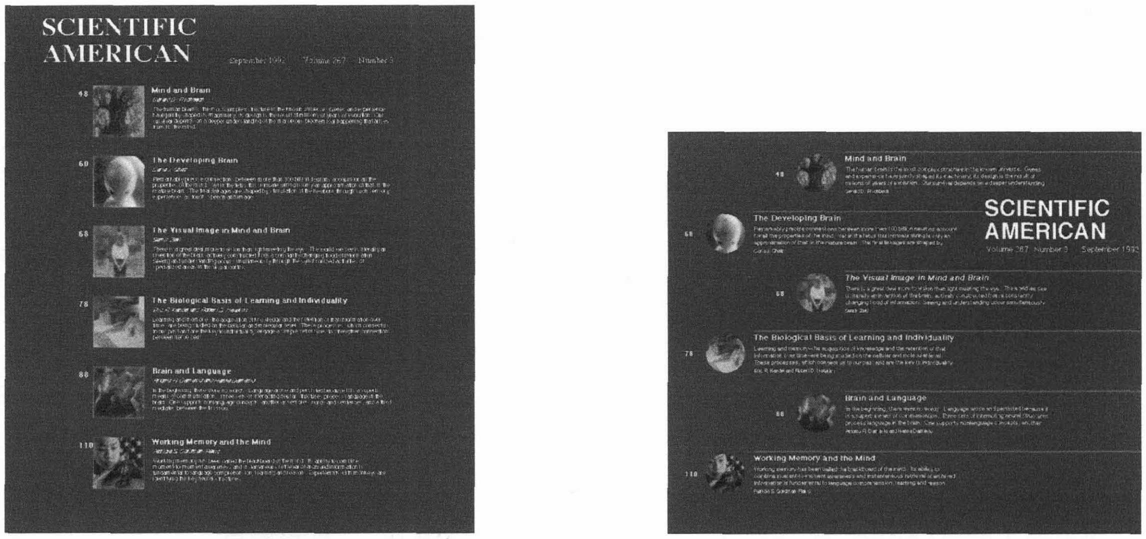

SCIENTIFIC AMERICAN
Mind and Brain
The Developing Brain
The Visual Image in Mind and Brain
The Biological Basis of Learning and Individuality
Brain and Language
Working Memory and the Mind
SCIENTIFIC AMERICAN

Figure 2-3: Automatic layout of table of contents by Weitzman's system. The two layouts are based on the same set of objects, but the one on the left is rendered in the style of *Scientific America*, while the one on the right mimics the look of *Wired.*

Early Systems: Gnanamgari and Beach

While there was earlier work in this field of automated design, they tended to be less focus and less concerned with the graphic design aspects. The first two to begin to tackle these issues seem to be Gnanamgari's BHARAT system [26] and then Beach's work to automate low-level design of 2D tables [17].

Gnanamgari's work focused on transforming tabular data into a set of default representations fitting the characteristics of the data. It did focus heavily on graphic design concepts and has as an express goal of allowing untrained users to perform the task of graphic artists by leveraging a "graphic-expert" system [26]. However, one of the limitations of her work was that much of the criteria governing effectiveness was "hard-wired" and therefore unable to be extended to a wide range of input.

Beach's work focused on giving the user more control of the graphic design properties of the resulting graphic. The work focused on automating the low-level layout and design of tables of data. Users were able to specify the rows and columns and make explicit decisions on the design properties of particular elements. The remainder of the automated process leveraged an earlier research specification called *graphical styles* [16].

Mackinlay's APT

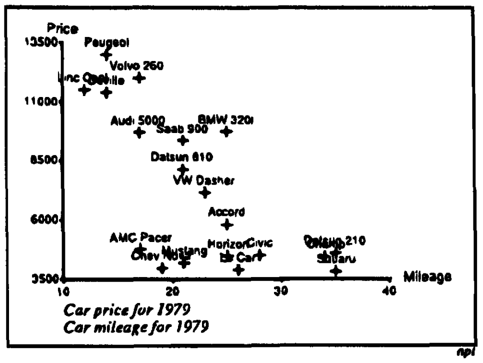
The work done by Mackinlay in implementing the presentation tool APT (A Presentation Tool) [37] represents a significant step forward in the field of automated graphic design. It also forms the theoretical basis for much of the work that followed. The goal of the research was the development of an "application-independent presentation tool that automatically designs effective graphical presentations" [37]. Simply put, the system was able to analyze sets of structured relational data and construct a suitable graphical representation (i.e. bar chart, scatter plot, connected graphs) of that data based on design criteria for both expressiveness and effectiveness. This approach leveraged the view that graphical presentations were part of a larger graphical language and could reduce to *sentences* about the related data. An example of the type of simple sentences found in APT to describe a data set relating car prices to mileage is shown in Figure 2-4 (Graphical representations of these sentences are also given in Figure 2-5.)

The notion of the two criteria, expressiveness and effectiveness, in the work of Mackinlay is important in that it provides a means for evaluating the designs generated by the system both in terms of the amount and accuracy of information that is able to be encoded into the graphical symbols and the degree to which humans are able to perceive the encoded relationships. The former he refers to as *expressiveness* and a design alternative is considered expressive of a set of facts if it is able to encode every fact of the set. The latter becomes *effectiveness*, which is a much harder metric to evaluate algorithmically as it depends heavily on a model of the capabilities of a human perceiver. The effectiveness model used in APT is based on the work on perception by Cleveland and McGill [21]. To better illustrate the difference between a presentation that is expressive for a data set versus one that is effective consider The diagrams in Figure 2-5. Mackinlay points out that the

```
Encodes(VertAxis, [3500, 13000], ScatterPlot)
Encodes(HorzAxis, [10, 40], ScatterPlot)
Encodes(Points, Cars, ScatterPlot)
Encodes(Position(Points, VertAxis), Price(Cars), ScatterPlot)
Encodes(Position(Points, HorzAxis), Mileage(Cars), ScatterPlot)
```

Figure 2-4: An example of APT's *graphical sentences* which describes relationships among the data elements.

presentation on the left is expressive for the data set in that it is able to fully represent its relationships, but it is not necessarily the most effective presentation since the labels are obscured. The diagram on the right, on the other hand, is a much clearer presentation of the same data so it is considered more effective for the data set.



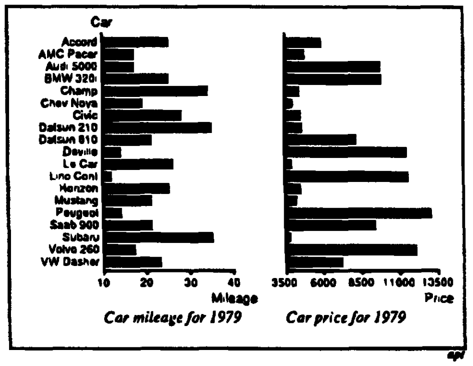


Figure 2-5: The graphic on the left is expressive of the data set, but not necessarily effective as the data labels are obscured for many of the values. The graphic on the right, on the other hand, is considered more effective even though the configuration makes direct comparison more difficult.

Casner's BOZ

BOZ [18] is an automated graphic design tool that extends the previous work of Mackinlay [37] and does so in two very significant ways. First, Casner challenges Mackinlay's assumption that there exists an effective graphical representation for a particular data set. He claims instead that effectiveness can only be measured with regard to how well it permits a user to carry out a given task. Following on this

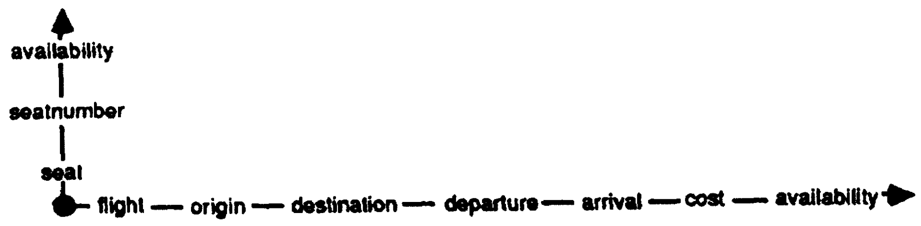

availability
seatnumber
seat
flight — origin — destination — departure — arrival — cost — availability

Figure 2-6: Grouping the operators based on characteristics of their relationships reduces the task to a vector representation. Each orthogonal vector indicates the need for a separate graphic.

correction, BOZ begins with a logical representation of a task to be carried out by a user. An example of such a task is given below, in natural language form.

> Find a pair of connecting flights that travel from Pittsburgh to Mexico City. You are free to choose any intermediate city as long as the layover in that city is no more than 4 hours. Both flights that you choose must be available. The combined cost of the flights cannot exceed $500. Find an available seat on each flight.

This is reduced to a set of domain specific logical statements representing the discrete subtasks that must be carried out in accomplishing the overall task. In comparison to Mackinlay's relational data set input, the set of *logical operators* are relatively complex.

The second significant extension is in BOZ's more formalized codification of the perceptual criteria that determines the resulting graphic. BOZ faces the difficult challenge of mapping the logical operator, derived from the task description, to a set of what are called *perceptual operators.* These operators are built atop a set of primitive graphical languages that is based directly on the graphical sentences specification in APT. In general, the perceptual operators characterize the information-processing activities performed within the context of the graphic. For instance, judging the distance between two objects and locating an object of a particular color are both examples of such operators. Interestingly, this abstraction to operators also allows BOZ to produce tiered graphical representations where the task cannot be fully carried out with a single graphic. Casner shows that grouping on the operators based on the characteristics of their relationships can reduce the task to a vector representation of the operators where each orthogonal vector results in a secondary graphic (Figure 2-6). In the case of the example task, a user has to first use a graphic to identify the particular flights before selecting seat assignments in the second graphic. The process of moving from the vector representation to a rendered graphic reduces to essentially the same generate-and-search solution used by APT. A representative graphic for the task statement given above is shown in Figure 2-7. Notice the two graphics are aligned with the vectors shown above (Figure 2-6).

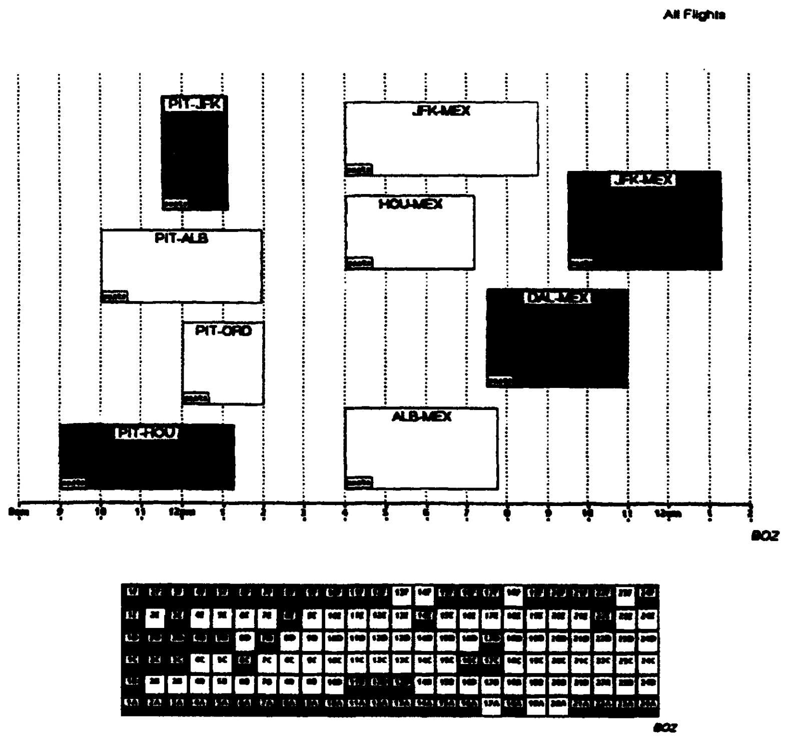


Figure 2-7: The graphic generated by BOZ for the task statement presented above.

2.2 Infrastructures for Adaptive Content Delivery

As the Internet has grown a viable means for distributing multimedia content, the number of different types of devices used to access that content has virtually exploded. Displays for desktop computers and laptops have continued to grow larger and more dense. Popularity of cell phones and other mobile devices have resulted in devices with a wide array of computation and display capabilities. Yet, the majority of the content available on the Internet was authored to be experienced on a single type of device (usually a web browser running on a desktop computer). As this convergence has taken place, there have been many attempts to remedy this situation by outfitting the network and server infrastructure with the capabilities to adapt the content to the device.

2.2.1 Architectural Variations

Adaptive infrastructure research has taken two general forms: server-based and proxy-based [24]. Server-based approaches generally work at the application level to tailor the content to the needs of the requesting client. An example of such systems are web sites that adjust the presentation to accommodate mobile devices (i.e. msnbc.com). In such systems, the computing machinery that carries out the adaptive transformations is tightly integrated with the system that is delivering the content. In the web context, this generally means that the web server from which the content is fetched is responsible for sending the content appropriate for the client. The other general approach is to leverage proxy servers distributed in the network infrastructure to transform the documents on behalf of a particular device or client. The client connects to the proxy server, which in turn fetches the content from the server. The proxy performs any necessary transformations to the content before returning an altered version to the client. A diagram that demonstrates the architectural differences between these two approaches is included below (Figure 2-8).

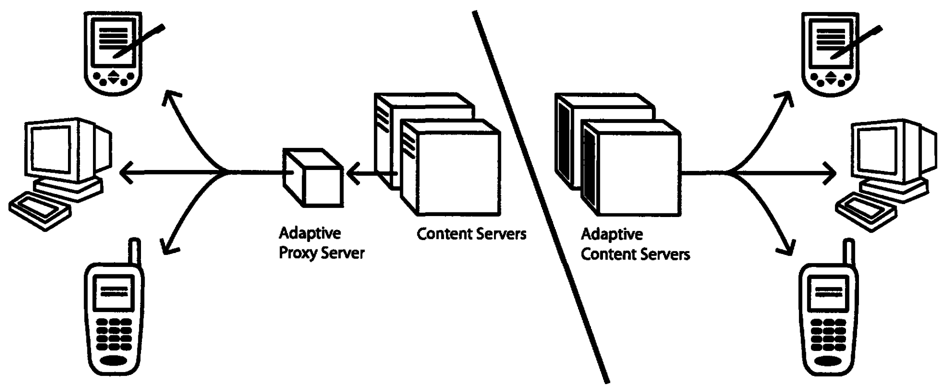


Figure 2-8: Proxy-based vs. Server-based architectures

Server-based approaches

Server-based approaches offer the advantage of being able to leverage the full data set available to the content delivery application. Whereas proxy approaches are limited to transformations on the content that was delivered from the content server, server-based approaches can deliver vastly different content to different clients. This permits much more freedom in adaptability. There is however a price in both engineering complexity and scalability. The content application and the mechanism for adaptation, for instance, are tightly coupled making it difficult to adapt to the specific constraints of individual devices. In fact, developers of such applications must generally be involved in the adaption for every device that needs special consideration. As a result, such systems tend to accommodate only a small number of device classes (generally, normal and small screen). The tight coupling of these two mechanisms also limits the degree to which such a service can scale to handle the loads of larger numbers of viewers. The fact that both the adaptation and the content application itself are effectively embedded within the same application often means they cannot be separated onto separate machinery for load balancing. This also creates constraints that prevent the adaptivity to be moved to aggregate points in the network. For instance, an optimal place to adapt to content to mobile devices is typically at the entry point to a wireless provider's network.

With server-based approaches, it is impossible to separate the adaptive processes and redeploy them elsewhere.

Proxy-based approaches

The proxy approach to content adaptation reaps all the benefits of having modular, autonomous systems. The proxy and the content application have no functional dependencies upon one another. This means each is able to benefit from simpler design and able to be relocated as needed for network efficiency. In practice, proxies are often created as device specific agents. Wireless providers, for instance, may deploy a proxy within their own network infrastructure to rewrite web content for mobile phones using lightweight web browsers. Such systems also tend to provide more flexibility in building scalable solutions as proxies can often be used to also cache transformed content for a large network of clients with similar constraints. Additionally, this configuration allows for more general use of the adaptive proxy. Rather than providing an interface to a specific application, as it does in a server-based approach, the adaptive proxy can provide a general conduit for access to a large number of applications (i.e. the entire Web). The generality of such systems also bring about a few limitations. The most serious of which is the fact that the proxy must use as its source the original content that was delivered by the content application. Two major issues arise. The first is that the proxy is only given the opportunity to remove or transform data in the original content. It is not possible, for instance, to add additional content to accommodate very large devices since the data is never presented to the proxy. Also, the proxy approach must be able to parse and make sense of the original content. On the web, this has proven to be a severe limitation as most existing web pages do not conform to standards for markup.

2.2.2 Approaches to Network Impoverishment

Most mobile devices rely upon wireless network connections to communicate over the Internet. While the data rates and reliability of such connections continue to increase, higher end connections have also increased in parallel. Also concurrent with these developments has been advancements in storage, networking and authoring tools that have driven the production of large amounts of rich multimedia content [44]. The limitations in networking, therefore, still provide an additional constraint which must be taken into consideration when designing and building applications for mobile devices. Much research work has explored the possibility of using infrastructural elements to improve network access for portable devices.

There have been several approaches to improving Internet connectivity in portable devices. General data compression has been one reliable way of improving end-to-end performance. The TranSend system proposed by Fox,et al. [24], leveraged data compression in proxy servers embedded in the network infrastructure upstream from pervasive devices. Later, the technique emerged in commercial offerings with Intel's QuickWeb [44] and Spyglass' Prism [44] and the latest major release of the Apache HTTP Server [4] includes a compression filter. Other techniques have included using multi-client proxies to reduce the resolution of images based on frequency of access [41]. Most of these, with the exception of the TranSend system, represent what Ma, et al., refer to as Content Delivery Networks (CDNs) which focus only in overcoming network deficiencies in delivering the original content [36]. Content Services Networks, on the other hand, introduce more elaborate processing services in the infrastructure nodes; including image and video trancoding, content adaptation, and watermarking.

2.2.3 Approaches to Screen Impoverishment

In addition to the network limitations of mobile devices, the small form factor of the devices themselves introduce perhaps a more limiting factor for the display of digital content. The screen sizes are often large enough only to display a mere fragment of content designed for larger devices [22]. Additionally, the small resolution of such screens affects the way images are displayed as they take up a larger portion of the screen unless they are down-sampled in some way. Most of the research work in using adaptive content delivery to overcome the screen impoverishment of mobile devices has been carried out as part of efforts to make mobile devices an effective medium for viewing content on the Web. An example is the Small-Screen Rendering^{TM}(SSR) mode available in the version of the Opera Web Browser that runs on phones and mobile devices [12]. SSR ensures that the content of the pages is linearized to produce a narrow vertical stream of content fitting the width of the device's screen filters out elements of the design which become extraneous in the new configuration. The primary challenge in such systems has been to transform or elide presented data in a way that does not disrupt or destroy the meaning that is being conveyed in the document. Two perspectives remain at odds in evaluating how meaning is conveyed within digital documents [29]. The *document engineering perspective* calls for the separation of content and style so that the content can be easily re-appropriated with different styles and in different contexts. This perspective generally promotes the use of hierarchical data formats where lower tiers of information can be safely elided while preserving meaning at a more general level. The *graphic design perspective* sees the content and presentation elements as being much more reliant on one another. Here the content is part of a complete message to be conveyed to the viewer and cannot be effectively separated from the presentation.

The work presented in this thesis attempts to begin to bridge the gap between the document engineering and graphic design perspectives. The majority of work in the area of Content Service Networks has attempted to further the state of the art within the document engineering perspective. Efforts here have tended to adapt documents with consideration to two different sets of relationships present in digital documents: the semantic structure of the markup language and the visual or interactive function of particular objects.

Adaptation using semantic structure

Web documents are typically authored in markup languages, like HTML, which have a structure that provides meaning within the context of data presentation. Much work to adapt web content has attempt to leverage these meaningful relationships to provide equally meaning fully transformations on the digital document so that the meaning is preserved after the transformation has been applied.

In theory, the use of semantics of markup should provide an ideal way to transform content without disrupting the encoded meaning in a document. It takes the structure, as given by the author or designer, as a guideline. The general thinking is that transformations that preserve the document outline (i.e. preservation of all major sections set off with header elements), will also effectively preserve the meaning, if only summary form, of the overall document. The approach, however, suffers two very practical limitations. First, most “real-world” web pages do not use proper semantic markup. This has been most pronounced in the use of the tabular data element (TABLE) to do general page layout. The second is a shortcoming of HTML itself. While HTML is intended as a language to describe the presentation of content and data, there is a fine line between presentation and structural representation of data [19]. One of the classic examples is the ordered and unordered lists elements (UL and OL). If for instance the navigation of the site consists of a series of links to other areas, does this constitute presentation as a list? The answer is often ambiguous and authoring practices vary greatly. Web standards advocates, for instance, have debated whether the chronological enumeration of reader comments that appear on weblogs should be presented as a list. These ambiguities often make it very difficult to automatically infer an author’s intent as that intent could be based on either presentational or content structure aims. Finally, it should also be stressed that this approach exemplifies the document engineering approach which generally assumes that the presentation of the content within documents does not convey any essential meaning.

Adaptation using visual function

Another approach to the problem of understanding the relationships of objects in a web document has been to attempt to categorize objects by their function. For example, a sidebar with the primary navigation links could be classified as being essential to navigation and could be adjusted in a different way from a less essential element, like a decorative border. The primary body of work in this area was lead by a group of researchers at Microsoft Research Asia [28, 19, 20]. The goal has been to identify logical relationships between web content based on layout information to facilitate automatic content adaptation. Their work aimed to make web pages designed for larger devices accessible to PDAs and Smart Phones. Pattern recognition and clustering algorithms are used to identify particular screen objects and categorize them according to the function they provide in the document including Navigation, Interaction, Decoration, Special and Page objects [19]. Using these object groupings, a full summary view of the pages can be displayed whereas details of each content object is made available in a hierarchical fashion depending on the function of the object. For instance, clicking on the navigation bar can either provide a zoomed view of the navigation options or extract the options into a textual menu for a user to select from. An example of a categorized page and the its equivalent representation on a PDA is shown in Figure 2-9.

Relying on the visual function of graphical objects provides a nice way to achieve meaningful adaptation while ignoring much of the meaning of the content contained in a document. Disambiguating an author's intent, as mentioned above, can be problematic. Additionally, approaches that rely on markup rest on the fallacy that final appearance has no effect on meaning. If visual function can be effectively determined for each graphical element, adaptation can be carried out in preserving what a user sees and not on what information the document contains. Extracting visual function, however, is also problematic. Finding the screen location for an object in the markup is difficult unless the process if tightly coupled with a renderer. Also, determination of function based on markup and characteristics of screen rendering is a highly inaccurate endeavor at this point and is plagued by the same impracticalities of the semantic approach.

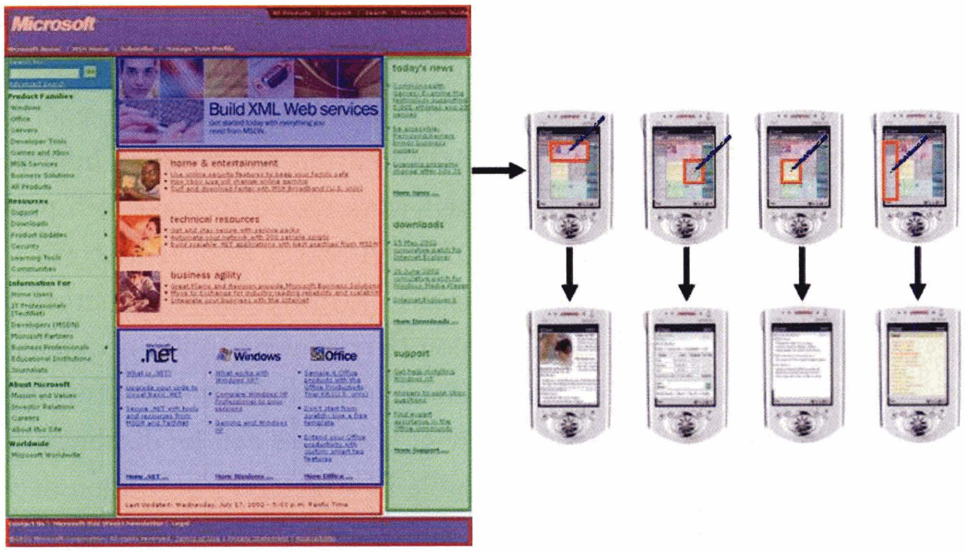


Figure 2-9: The FOM approach attempts to categorize graphical objects by their function. The navigational column in this graphic is recognized as such and can be rendered according to navigational needs. This also allows interaction adjustments. For instance, the navigational elements might produce a menu when the users clicks in that region.

2.2.4 Adaptation of Video

While the techniques described above have been used to adapt digital documents to particular constraints of the requesting device, a different set of techniques have emerged for dealing with and adapting the contents of video. The work of Teodosio and Bender, for instance, demonstrate the use of optical flow, a concept taken from computer vision, to stitch together video volumes that span a segment of video in order to deduce a *salient still* that best represents the contents and context of the video segment [45]. A *salient still* of a video clip is not necessarily an extracted frame from the original video sequence, but a high quality single image representing the aggregate of temporal changes that occur over the entire time frame. Also Holtzman's work on Media Banks [30] contained an early demonstration of videos available in a modified version of the Mosaic web browser that were rendered as single frames in browsers that did not have the appropriate software to link to the video stream coming from the larger Media Bank. The Media Bank system delivered to the client a single representative image accompanied by additional media stream locators. Un-enhanced browsers displayed the image, while those with the Media Bank Plugin initiated a video stream. The adaptive decisions within the Media Bank were, therefore, primarily client based.

2.3 Working life in the age of computing

In the past fifty years, computers and telecommunications have become increasingly present in daily workplace activity. Throughout this rise in popularity, the speculations of how these systems and machines would transform the way we carried out our work responsibilities has been the source of much fascination and anxiety. Lately, there has been a new wave of anxiety as organizations explore the feasibility of outsourcing work to countries where labor wages are significantly lower. At the same time, however, trends are emerging that enable more freedom in the way that workers are coordinated to carry out work for organizations. In this section, we look at two models of work organization with different relationships to computing. The first is the speculated trend in eLancing, where workers are enabled by the telecommunications infrastructure to organize themselves in a more autonomous manner. eLancers also reap the benefits of emerging software systems to connect employers with specialized workers that constitutes an economy where labor is engaged with finer granularity, i.e. at the project level. Secondly, we consider efforts to combine humans and computers in different fashions so that elements of machine and human computation become part of a single system, with each solving the parts of problems that are most fitting their abilities. This too constitutes a potential labor economy where human workers are paid to solve problems that are not possible for or not elegantly solved by automated computing solutions.

2.3.1 eLancing

In 1998, two MIT researchers projected that freelance work where the Internet served as the primary means of finding, conducting and delivering work would become increasingly prominent [38]. Others have argued further that this model will eventually be the dominate type of work organization. [35]. The main components of the is model, commonly called eLancing, are [32, 31]:

1. Workers are self-employed freelancers who organize their working life at the project level, where projects might last a few hours, several months or more than a year.

2. Trading takes place in an open market of services, framed by skills and personal preferences.

3. Virtual organizations are formed to accomplish complex tasks or projects. Such organizations consist of eLancers bringing together their unique skills to act temporarily much like a traditional firm. When the task is fulfilled the organization disbands.

4. The networks of eLancers are mostly self-managed with no need for centralized managerial control.

5. Marketing and collaboration is carried out on electronic marketplaces established on the Internet.

There is some indication that their prediction by Malone and Laubacher may be gaining ground. While the number of freelancers who can be described as eLancers remains a difficult measure to isolate, one study has shown that in a few technology sectors the percentage of tele-workers who are self-employed is significantly different from percentage of all workers who tele-cooperate [25]. While these findings are far from conclusive, organizations in the United States have shown a willingness to fragment the traditional corporate hierarchy in favor of the reduced operating costs promised through outsourcing and offshoring. The extent of this trend is still the object of much speculation and debate.

Recent years have also seen the success of online communities that serve to bring together freelancers with employers for project-level employment [1, 2, 3]. In such sites, there is typically a process of advertising a request for bids. The details of the project are then collaboratively tuned and agreed upon by the trading partners. The sites usually also provide an escrow service to mitigate risks of non-payment or non-delivery.

2.3.2 Human Computation

Beyond computing and networking systems being used to bring people together for work, there are efforts emerging to bring people and computers together as a single problem solving system. Efforts in artificial intelligence have proved successful in very limited domains. In most areas, however, such systems have limitations that continue to make them inappropriate for production-level systems. At the same time many of these tasks are considered trivial to humans. The only restriction to leveraging human intelligence has been providing the appropriate access and the appropriate incentive.

The work of von Ahn, et. al, uses the Internet as the means of access and game play as an incentive to solve problems such as automated generation of text description of images and automated object recognition in images. Both of these domains are areas where algorithms for computer vision have shown much promise but still lag well behind the average human in performance (both in terms of accuracy and expressiveness). Two projects, Peekaboom and Phetch, are of particular interest since they demonstrate that humans can work as part of a computational system to solve hard artificial intelligence problems if given the proper incentive.

Peekaboom [49] is a web-based game where as the participants play the game they are building a bank of metadata that describe the objects featured in the collection of images used in the game. Participants are randomly paired with partners and take turns Peeking and Booming. The player who is booming is given an image and a word related to it and must reveal a part of the image to help the peeking player guess the word correctly. The peeking player is able to enter multiple guesses in trying to determine the given word. Booming players can also Ping, or highlight a particular point in an image to disambiguate a specific part of the image. As part of a bonus round, players are asked to click on a particular object in a given image. The data collected from each of these game play levels is collected in a database to be used to help researchers produce more advanced computer vision algorithms.

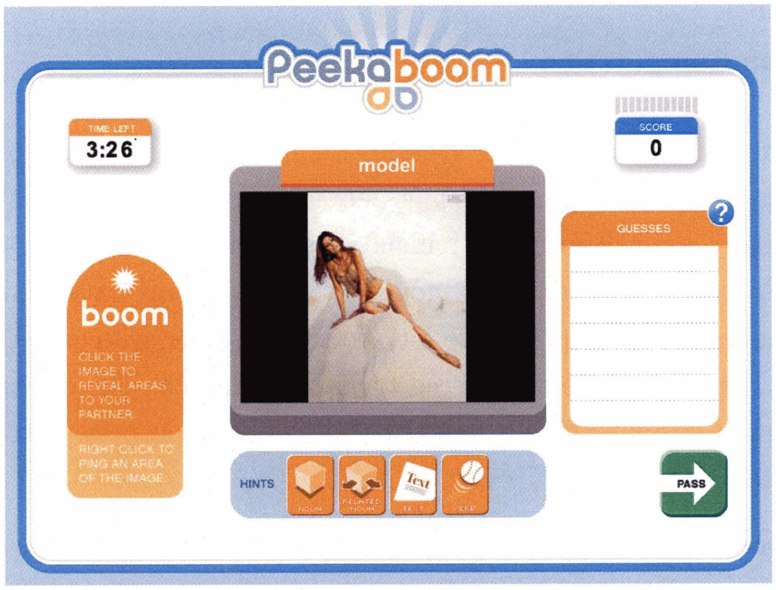
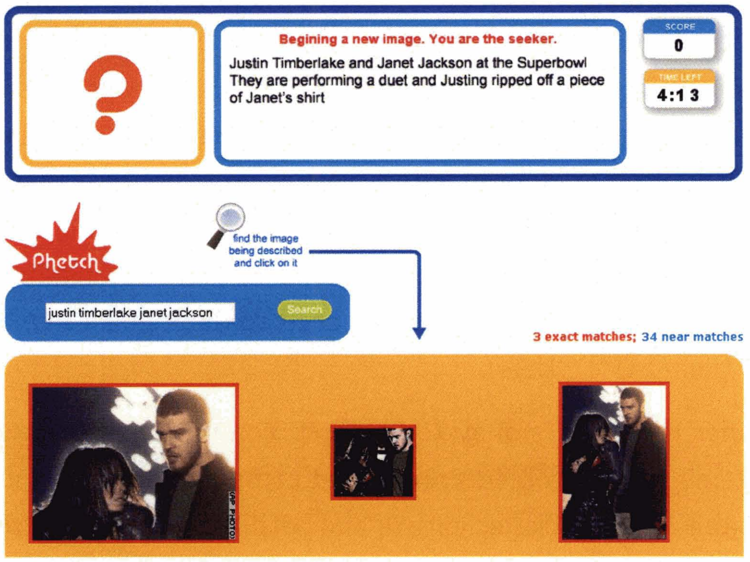

Peekaboom
TIME LEFT
3:26
model
SCORE
0
GUESSES
boom
CLICK THE IMAGE TO REVEAL AREAS TO YOUR PARTNER
RIGHT CLICK TO PING AN AREA OF THE IMAGE
HINTS
PASS

Figure 2-10: A screen capture of Peekaboom during gameplay

Phetch [48] is a system similar to Peekaboom in that it brings humans together in the context of game play where their leisure activity is actually contributing to the solving of a large-scale knowledge acquisition problem. In Phetch, there is a *Describer* and several *Seekers*. The Describer is presented with an image and must help the Seekers to find it using an online search engine by providing a textual description of it. Communication is one-sided as the Describer is only allowed to broadcast a description to the Seekers, who not allowed to ask for further clarification. This permits the Seekers to be played by bots in situations where there are not enough humans for full game play. Points are obtained for both the Describer and the Seeker when the proper image is located. The system collects the successful descriptions for the images as part of a database of natural language descriptions which can be used to augment images for the visually impaired. The system is presented as a solution to poor accessibility to the web for the visually impaired since web authors rarely included effective textual descriptions for images.



Begining a new image. You are the seeker.

Justin Timberlake and Janet Jackson at the Superbowl They are performing a duet and Justing ripped off a piece of Janet's shirt

0

4:1 3

justin timberlake janet jackson

3 exact matches; 34 near matches

Figure 2-11: A screen capture of seeker inferface while playing Phetch

These two systems rely on play as an incentive to encourage human participation in problem solving systems. Yet other incentives exist, including money. Google Answers [7] relies on money as an incentive to research and answer arbitrary questions posed by users through an interface that very much resembles Google's standard search interface. Such systems suggest the opportunity for humans to earn money by serving as the intelligence behind computer interfaces. This thesis, for instance, proposes such a system for adapting graphic design solutions to the myriad contexts in which digital documents are displayed using a similar technique. Such systems demonstrate a new kind of intelligent system and a new kind of work organization.

2.3.3 Computing Systems, Human Systems and Hybrid Systems

As the work cited above demonstrates, the realm of computing has become more than just an attempt to automate the repetitive and solve the logical. Just as important is the ability of modern computing systems to connect people and we have seen enormous growth in social-based software in the past few years. At the same time, we have watched artificial intelligence systems make significant progress and be applied to a wide range of different problems. And the promise of fully automated, systematized solutions for problems that make our lives easier or richer continues to drive this pursuit. However, we are seeing a trend in the limitations of AI solutions. They often work only within very narrow bounds, theoretical ideals, which can lead to poor performance in practical systems. More often than not, the cases where such automated systems break down are trivial for humans. In fact, we often depend on this phenomenon when we need to distinguish humans from computers [47] (i.e. CAPTCHAs).

Below we consider three types of systems. First, computing systems represent fully automated systems where problems are solved computationally. Such systems are very much the aim of computer scientists and artificial intelligence researchers. Second, human systems are problem-solving systems where large numbers of people are united through a communication infrastructure and are available to cooperate and solve problems. Finally, Hybrid systems are problem-solving systems which combine the power of both humans and computers and try to leverage the advantages of each to create a more powerful and adaptive system.

Computing Systems

Pure computing systems have represented an ideal in the emergence of the information age. We envisioned a computer working tirelessly on our behalf and responding to our queries with immediate and accurate feedback. Their primary advantage is their on-demand nature; ask and you shall receive. The computing system provides a very formal interface to pose a problem and returns in a deterministic time frame. The calculator, for instance, is a computing system that has proved its worth. Posing the problem requires the user to punch an appropriate sequence of keys and the response is almost immediate. Unlike the calculator which can accurately solve a relatively easy problem, computer systems begin to break down under more complex situations. For example, despite the efforts of many brilliant computer scientists, most spam filters are still unable to identify 100% of the spam arriving in users' inboxes. Such is the primary disadvantage of pure computer systems. They are limited by the automated problem solving algorithms which tend to work only for narrowly focused and ideal situations. Practical deployments are often plagued by edge cases (the small percentage of SPAM that slips through the filters, for instance). Computers do not solve problems creatively, which will continue to limit their ability to return results that are satisfactory to humans.

Human Systems

Human systems are essentially the opposite of the pure computing system. Problems are solved by humans and therefore they are able to cover a vast spectrum of problem domains, including those that require creativity. Such systems tend to emphasize collaboration among members and are highly social in nature. However, this also leads to very organic interfaces since problems are typically posed in natural languages and the format and protocol of the response is a typically negotiated through the collaboration. The time frame to reach a solution is also high non-deterministic. Many postings to online forums, for instance, go completely unanswered. In fact, the availability of a human to solve a problem in such system is not guaranteed at any time. In practical terms, this concern is often alleviated by the involvement of large numbers of people. However, human systems tend not to be highly available and reliable. A final disadvantage is that unlike computers, humans must have an incentive to take part in such systems. Incentive can take on many forms: from play (as in the work of von Ahn), monetary compensation, social belonging, fame, etc. Such incentives have to be sufficient at all times to foster adequate human involvement.

Hybrid Systems

Hybrid systems represent a new kind of system that attempts to bring together humans and computers in a network to solve problems. The hope is that each will contribute in a way that best leverages their strong points. They combine the availability and on-demand nature of computing systems with the enhanced problem solving and creativity of human systems. Such systems strive to provide formal interfaces to facilitate automated communication, yet there is a human element alive in the background that is available to contribute knowledge or solve problems directly. The combination of the two types of problem solvers now changes the game for each in a lot of cases. Computer systems, for instance, now have added incentive to know their own limitations. It is feasible, for instance, for computers to solve the 90% of the problems that fit within the bounds of their automated solving rules and allow humans to field the remaining 10% of edge cases.

2.4 Summary

This thesis project sits at the intersection of the background work presented here spanning automated computer-based design, adaptive content delivery systems, and systems for leveraging human creativity. Important elements of this previous work are present in the principles and design of the adaptive rendering infrastructure that is evaluated as part of this thesis.

In the domain of automated computer-based design, the large systems presented here (specifically BOZ and APT) represent significant engineering accomplishments. However, these systems suffer the trappings of automated pure computing systems described above. They tend to work well under very ideal conditions but breakdown in a more practical environment. Additionally, they fail to take into account the contribution of human creativity to the process of graphic design. Consider the solution designed by BOZ in Figure 2-7. The bottom diagram depicts the vacancy of seats on a particular flight. The automated system has been able to generate a diagram that leverages abstract shapes suited for displaying a particular type of value. However, a human designer has the added ability to leverage shared experience and metaphors. The designer might instead design the diagram to more closely resemble the interior of an airplane. Perhaps framing the diagram with the airplane's silhouette and depicting the seats themselves (as opposed to squares meant to represent seats). The use of metaphors and shared experience is essential when designing for constrained environments as it favors information we know over information that must be conveyed.

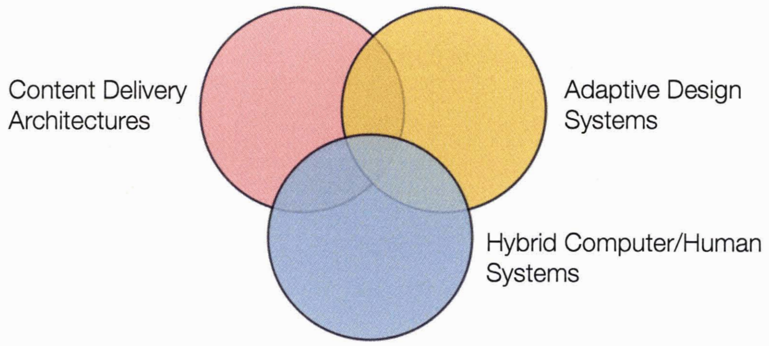
Adaptive content delivery systems provide a means of supplying content over the network in a way that most suitably fits the device on which the content is displayed. While the simplicity and scalability of proxy-based approaches is desirable, the adaptive power that is inherent in server-based models is generally required to do anything but the most basic adaptation. In the project carried out as part of this thesis, a server-based model was essential as no adaptation is carried out with a document engineering perspective. In other words, we do not try to redeploy content and presentation separately. The document engineering perspective has offered a number of opportunities on the web and elsewhere in terms of accessibility for people with disabilities, automated summarization of content and others. It still, however, does not take into account important elements of the graphic design perspective. Most importantly the separation of content from presentation commits to a decoupling that has proved inappropriate to effectively apply the principles of good graphic design. Finally, both approaches to content transformation, semantic markup and visual function, both represent automated processes that due to edge cases and ambiguities in the document formats remain impractical for general use. We feel this is an area where human intelligence and creativity can offer drastic advantages over automated solutions.

The process of enabling the human intelligence at a system level requires a hybrid system which is aimed to exploit the advantages of both computer and human systems of problem solving. The roles played by computers and humans in hybrid systems leave a lot of room for experimentation. However, ensuring that the hybrid system is able to provide the on-demand characteristics of automated systems yet provide the level of creative intelligence available to human system is the most desirable situation. Peekaboom and Phetch both represent hybrid systems that rely on play as an incentive for humans to take part in the system. While play is certainly a compelling incentive for some tasks, pay is a more universal and extends greater flexibility in the task requested. It is also in keeping with observed trends in eLancing and other work models that emphasize self management and project level assignments.

Chapter 3

Design and Implementation

The execution of this thesis project represents a two year assemblage of experiments all aimed at exploring the concept space around three interrelated ideas and fields: adaptive design systems, content delivery infrastructure, and hybrid human/computer systems for problem solving. The driving motivation is to produce a centralized architecture for delivering content to widely varying classes of multimedia devices in a way that guarantees the desired user experience by opening up the digital design space to a more graphic design oriented perspective. Additionally, we feel this goal is best achieved by a new system that leverages human intelligence and creativity over automated AI systems and assert that such systems stand to change nature of the problems which can be systematized. At the same time, deployment of these hybrid systems represent new ways of working as the funnel fueling a demand for creative labor is effectively automated. The following chapter describes the design and implementation of this architecture and highlights the experiments upon which these ideas have evolved.



3.1 Early Experiments

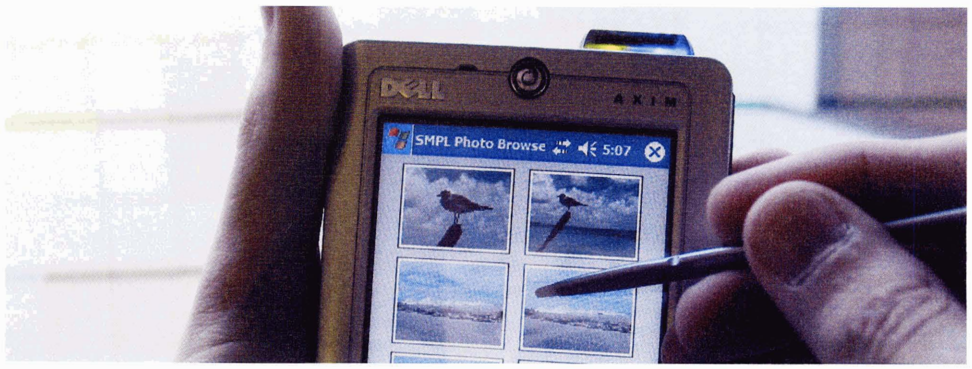


Figure 3-1: Mobile Photo Browser

3.1.1 Mobile Photo Browser - October 2004

Inspired by the popularity of such online services as Flickr [6] which allow users to share photographs easily by uploading them to a central server, we created a networked based rendering service to support more elaborate forms of sharing and interaction with photo databases. The project conceived of a media server that not only made smaller copies of photos for viewing on the web, but served as the primary storage for the original high resolution photos and adapted to the particular needs of the current viewer. In the demonstration, we were able to interactively browse a large collection of photos using a small PDA with wireless networking capabilities. The demonstration allowed the user to browse the full collection of photos as a set of small thumbnails, all of which were being generated by the backend rendering service. Selecting a single file gave the user the ability to adjust the scale of the photo on the screen. As with the thumbnails, the image being shown on the PDA is custom rendered by the backend server, which scales to the appropriate resolution and then clips the image so that only the contents currently showing on the PDAs screen were sent over the network. This prototype demonstrates the power of such services to provide visual access to a large dataset even on a device that could not typically accommodate even a single element of that data, in this case one of the full resolution images.

The Mobile Photo Browser was the first experiment of many that leveraged a Content Delivery Infrastructure with special consideration for smaller and mobile devices. Rather than adaptive, however, the considerations were mostly optimized for interaction on the mobile devices. When an image was displayed at full resolution on the PDA, for instance, the content server would scale the image to it's native size but crop it to the dimensions of the screen. This permitted the PDA access to content that would have previously overwhelmed the memory limitations.

3.1.2 eReader Content Rendering - March 2005

Similar in concept to the photo browsing demonstration, a later demonstration adapted the set of rendering services to provide a means to deliver electronic book content. In this scenario the rendering service is again creating thumbnails to assist in the navigation through the pages of content and then delivering raster views of the appropriate viewing window into each page the reader views. This demonstration was also the first to consider the ability to adapt to changes in context, as it provided a low quality mode for skimming or "flipping" through the pages of text quickly.

This experiment is the second to consider interactive limitations on mobile devices when dealing with heavyweight formats. In terms of content delivery, the experiment is similar in functionality to the photo browser with the new ability to render vector graphics (PDF). Similar to the photo browser, this experiment the adaptive functionality is all geared at supporting interactive browsing of content. The same cropping and scaling procedure that was used with the photo browsers is also user here. There is, however, an additional consideration in adapting content to the device. The fact that the user could switch the application into lower quality to further increase the responsiveness represents continued exploration of how the infrastructure can adjust to accommodate limitations on the client side.

3.1.3 OPENSTUDIO Artwork Rendering - August 2005

OPENSTUDIO was launched in October 2005 by the Physical Language Workshop as a new experiment in creativity, collaboration and capitalism. OPENSTUDIO is an online creative community that offers the means to create artwork using a set of network enabled creative tools and then share and sell them in the same environment. Documents created by each of the different tools are stored in the database in vastly different formats. To ensure that the content produced in the community is able to be used in environments other than the java-based design tools on the site, we introduced a rendering service similar to the two described above which is able to produce a fully rendered raster image for any document in the database, regardless of which tool was used to create it. Additionally, this version of the rendering service included the new ability to composite together multiple documents into a single image. This artwork renderer currently provides the rendering of all the web content for the OPENSTUDIO web site and has allowed us to incorporate the varied content contained there into a number of different application environments.

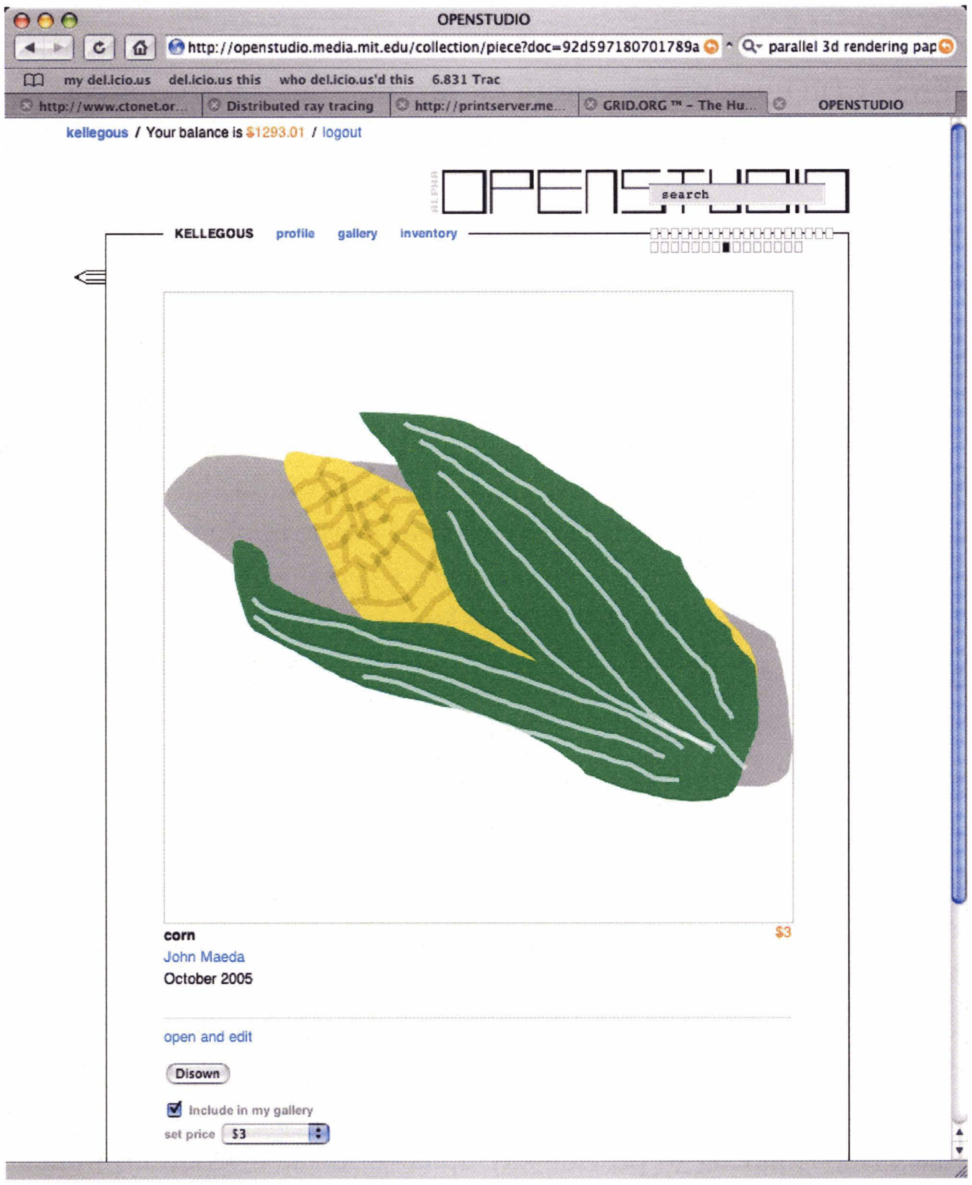
The experiment carried out through the artwork renderer focuses primarily on the system issues that emerge in building server-based content delivery infrastructures. The server was required to provide quality raster renderings of the artwork within the OPENSTUDIO creative community at any requested size. Yet there were also strict requirements on the amount of latency that could be tolerated since the server bears the burden of rendering all the artwork for the community's web interface. This particular experiment also considered the content delivery infrastructure as a mechanism for generalized access to documents. Regardless of the OPENSTUDIO tool used to create the artwork, the rendering server guaranteed an image could be created to display the contents of that file.

3.1.4 Samsung Photo Player - December 2005

One example of an existing product that leverages a very simple infrastructure like the one described here is the Samsung Photo Player device. The device, which is being sold to accompany a photo service provided by Korean Telecom, receives content and commands from a central server. The server, in this case, is responsible for ensuring that the images and videos are sent at the appropriate format and resolution. We have experimented further with the device by adapting a bridging service to our own infrastructure. This permitted the device content to be drawn from the sources described above. In each case, the infrastructure automatically adjusts to the constraints of the screen.

The Photo Player is exactly the type of device to reap the benefits of the system proposed in this thesis. It has limited computation power and memory and is also not able to adapt well to other forms of information as the code on which it runs is firmware-based. However, the use of a network-based infrastructure makes these limitations irrelevant as the computation is done elsewhere in the controlled environment of a data center. Most importantly, the device can rely on the infrastructure to adapt when it can't. This opens up additional opportunities to bring a wide variety of information (i.e. weather, financial, personal messaging) to the device without having to provide any updates to the internal firmware.

The photo player experiment demonstrates the need for both the adaptive design component and the content distribution infrastructure. Consumer electronic devices like the one shown are a rising trend each promising to carry content and information to us in new ways. The diversity of these devices, including platform,



corn
$3
John Maeda
October 2005

[✓] Include in my gallery
set price $3

Figure 3-2: A screenshot from the web interface of OPENSTUDIO

Figure 3-3: Samsung Photo Player displaying photo content delivered from a content delivery service

screen size, network reliability increases the challenge to satisfy the emerging demands for content and information.

3.2 System Overview

3.2.1 Rendering Architecture

The emerging architecture for the rendering system, consists of three primary parts; dispatchers, compositors and rasterizers. Each of the different sub-systems is able to operate on a different computing host taking advantage of the parallelism inherent in some of the image assembly operations. Each of these systems uses the SMPL framework [42] to communicate with the other parts of the system. A basic description of each of the systems is given below.

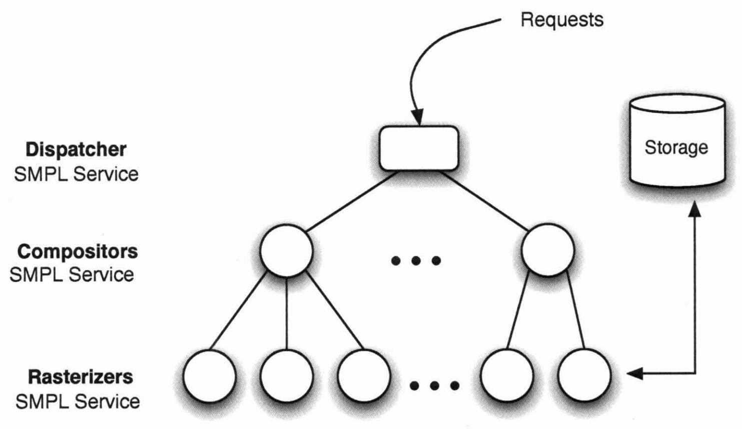


Figure 3-4: Distributed architecture of the rendering system composed of dispatchers, compositors and rasterizers

Dispatcher

The dispatcher provides the primary interface used by clients to make requests. Requests are received over the network using any of a variety of different web service protocols. After receiving a request, the dispatcher is responsible for allocating a suitable compositor and the providing that compositor with a set of rasterizers in order to carry out the construction of the final image. Dispatchers also play the role of load-balancer for the system, since they are responsible for allocating all "downstream" services.

API

```
ID createCanvas(int width, int height);

void addDocument(ID canvasID,
    ID documentID,
    float dstX,
    float dxtY,
    float dstWidth,
    float dstHeight);
```

```
void addDocumentRegion(ID canvasID,
    ID documentID,
    float srcX,
    float srcY,
    float srcWidth,
    float srcHeight,
    float dstX,
    float dstY,
    float dstWidth,
    float dstHeight);
}
```

Rasterizer

As their name suggests, rasterizers are responsible for turning a number of different file formats into a common pixel based image format, or raster. Requests to the rasterizers specify a document, the size of the image to be produced, and the rectangular region to be used in the requested document. Rasterizers are able to handle several document formats and can easily be extended to accommodate more through a plugin system. Currently, the system recognizes PDF [14], SVG [13], and four propriety OPENSTUDIO file formats.

The rasterizer module was originally designed to take upstream requests from the compositor modules. However, they can also receive requests directly from remote clients when a raster representation of a single document is needed. OPENSTUDIO's artwork renderer, for instance, uses the rasterizers directly when it renders only one document in each image (Figure 3-2).

Rasterizers also provide the platform for conducting experiments with adaptive graphical solutions. To carry out the analysis listed below, a special plugin was added to the system which allows certain documents to be specified as adaptive. Adaptive documents consists only of a set of sub-documents, each of which constitutes a design solution optimized for a particular size and aspect ratio. On each request, the sub-documents are analyzed and ranked according to effectiveness for the output characteristics. If all of the documents fall below a particular threshold, the plugin has the capability to notify a human designer of the need for a more fitting solution.

API

```
public Raster rasterize(ID documentID,
    int width,
    int height);

public Raster rasterize(ID documentID,
    int width,
    int height,
    float srcX,
    float srcY,
    float srcWidth,
    float srcHeight,
    float opacity);
```

Compositors

Compositors are responsible for taking the images returned by rasterizers and combining them according to the directives contained in a client's request. For example, a client request might specify that four documents be sized and placed in a grid. The compositor assigned to the task would use a set of rasterizers to render each of the four documents. It would then take the four resulting images and combine them into the proper positions in a single image before returning it to the dispatcher to be sent to the client.

API

```
int createCanvas(int width, int height);
```

```
void addLayer(ID canvasID,
    ID documentID,
    float srcX,
    float srcY,
    float srcWidth,
    float srcHeight,
    float dstX,
    float dstY,
    float dstWidth,
    float dstHeight);

void addLayer(ID canvasID,
    ID documentID,
    float dstX,
    float dstY,
    float dstWidth,
    float dstHeight);

Raster composite(ID canvasID);
```

3.2.2 Hire-a-Designer

The OPENSTUDIO project being carried out by the Physical Language Workshop is an online creative community that provides members the tools for creating digital arts as well as the means to market and sell that art. One of the aims of such a community is to provide a centralized market for creative work so that artists can one day profit from previously unsatisfied creative needs of others. Consider, for instance, an executive in a small company quite adept at configuring the equations in a spreadsheet. In preparing the quarterly report for his company's investors, he looks for an intuitive and elegant graph to emphasize the important parts of the data. Surely there is an information designer out there who would

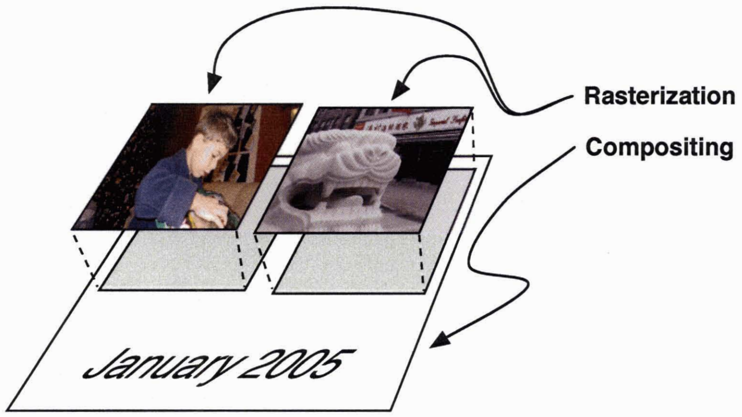


Figure 3-5: The rasterizers turn individual documents into pixel representations that are passed to the compositors to be blended into a single composite image

happily solve the problem for a reasonable price. This is an economic opportunity for both the executive and the artist that is impractical in current working environments and in current software models.

The Hire-a-designer feature of OPENSTUDIO is a system that strives to make these unlikely connections to artists in a creative economy. Hire-a-designer combines a straight-forward commissioning system with a set of Web API's to allow for the bidding, contracting and delivering of creative services from a web interface or directly from other software tools. In the scenario given above, the executive might be able to click on the graphing icon that is typically seen in spreadsheet applications but choose to commission a designer to create the graphic rather than deferring to the application's algorithmic solution for graphing. The requests are all transferred over the network and the OPENSTUDIO infrastructure mediates the transaction from request to delivery.

Automated commissions

The Hire-a-designer feature of OPENSTUDIO provides the link to the human element in a hybrid computer/human problem-solving system. The creative workforce that is made up of the artists in OPENSTUDIO provide the human intelligence resources by which adaptive design is carried out. The selection engine, which is described in more detail below, attempts to find appropriate designs among those already submitted to the database. If, however, none of the designs are suitable, an artist is automatically commissioned.

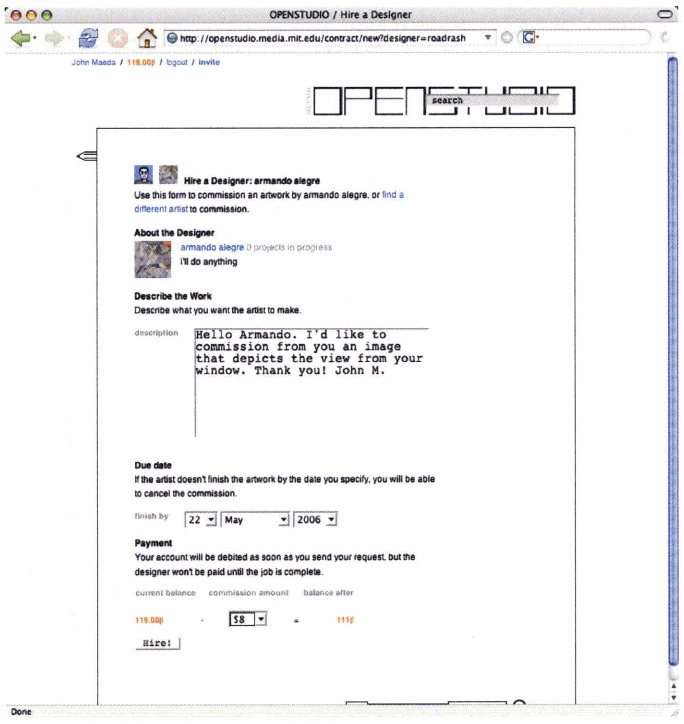
One question that arises when faced with programmatic access to human workers is how the system should proceed after a commission request is made. The time necessary for the design to be completed by the human depends on very many factors and even in the most ideal situation, the delay will be far too long to postpone delivery of the design until the task is complete. Instead, the current system returns the design that fits best (even though it is below the threshold) and also issues the commissioning request. The hope is that while the content that is delivered is suboptimal for this particular request, future requests will be of a much higher quality.

There is also the issue of deciding which designer to commission. A current limitation of hire-a-designer is that it does not allow requests to be broadcast to more than a single designer. It also does not have any notion of a work queue, where available artists can take jobs as they enter the system. Additionally, there are other human factors like trust, cost and delivery time which are not considered in this version. A more detailed discussion of these metrics is included in the next chapter.

Issuing the commission request requires only a single remote procedure call. A new document to hold the new design is automatically generated and made available to the artist under contract. This document is added to the set of documents considered by the design selection engine, but is not considered until the contract with the artist has been finalized. Also, the API does not provide any formalized way to submit the details of the screen context to which the design should be tailored. Since the task is intended for a human, however, the selection engine provides that information as part of the natural language description field.

API

```
ID createJobRequest(
    ID artistID,
    String description,
    Money offeredPrice,
    Date requestedDate);
```



OPENSTUDIO / Hire a Designer

http://openstudio.media.mit.edu/contract/new?designer=roadrash

John Maeda / 118.00฿ / logout / invite

Hire a Designer: armando alegre

Use this form to commission an artwork by armando alegre, or find a different artist to commission.

About the Designer

armando alegre 0 projects in progress

i'll do anything

Describe the Work

Describe what you want the artist to make.

description: Hello Armando. I'd like to commission from you an image that depicts the view from your window. Thank you! John M.

Due date

If the artist doesn't finish the artwork by the date you specify, you will be able to cancel the commission.

finish by: 22 May 2006

Payment

Your account will be debited as soon as you send your request, but the designer won't be paid until the job is complete.

current balance		commission amount		balance after
119.00฿	-	58	=	111฿

Figure 3-6: A screen capture of the hiring interface for the Hire-a-designer system in OPENSTUDIO. Members of the community are able to hire other members to create designs for a negotiated fee. The system takes care of financial transactions and delivery of the artwork.

3.2.3 Design Selection Engine

The primary engineering focus of this thesis is embedded in the final part of the software infrastructure for rendering, the design selection engine. The purpose of this module is to select the best among a set of design alternatives for a particular screen context. Secondarily, the module recognizes when none of the design solutions are sufficient and dispatches a request to a human designer to produce an original and suitable solution. The design selection module represents a new consideration for software acting in a hybrid computer/human system. It does not generate solutions to posed problems, but evaluates existing solutions for appropriate fit. The system is therefore much like the classic generate-and-search approach to AI problems, except the generation is carried out entirely by humans.

The implementation is situated within the larger rendering infrastructure as a plugin inside of the rasterizer module. Document identifiers refer to the collections of design solutions for a particular graphical message. Such a message could be a graph depicting financial information, an advertisement or any other visual message crafted through graphic design. Unlike other rasterizer plugins, like the PDF plugin, which merely converts vector based formats to a pixel buffer, the design selection plugin also has the capability to contact OPENSTUDIO's hire-a-designer feature. This mechanism provides the link between the computing system and the pool of human designers that contribute human creative intelligence when the system determines that none of the available design solutions are sufficient.

At the heart of the design selection engine is a mechanism capable of evaluating the available design solutions and assigning each a score that represents its effectiveness in the given context. The extreme subtleties that contribute to the art of design make it virtually impossible to consider each of the different elements. However, given the tomes of graphic design principles published throughout the fields history and the understanding we have of the human perceptual system, it should be possible to derive a set of design features that generally contribute to the design's ability to communicate a message. Finding the right combination of metrics by which to evaluate a design is an ongoing process, the version of the selection engine presented here relies on a small number of intuitively key selection metrics: deviation from original size, deviation from aspect ratio, percentage of legible features, and change in information content of embedded images.

Metric #1: Deviation from Original Size

The degree to which the size has to be changed is, intuitively, an indicator of how well a design fits the current context. Vector graphic documents have encoded in the format a native size. In PDF, for instance, there are a number of encoded rectangles (generally known as the mediabox, bleedbox and trimbox) that indicate a size for the document. The design selector treats these sizes as target sizes and adjusts the score for the design based on the degree of scaling that must be applied. Since aspect ratio is considered as a separate metric, the size is simply reduced to the difference in both width and height between the context and the document's native size. While size may be the most intuitive of the metrics presented here, it also tends to be the least unreliable. Vector documents are often authored with a native size that is drastically different than the intended viewing size. In fact, most of the popular illustration applications default to a US Letter sized frame. Many authors treat the frame as a virtual canvas and simply scale the document accordingly when placing it in a display context. For this reason, size is considered only as the lowest priority tier. It simply places preference on the design authored at a size closest to the size requested when all else is the same. Many of the metrics presented here score all designs that do not have to be scaled down similarly. It is therefore possible that designs targeted at smaller screens could outscore those that are a closer fit. The use of size is an attempt to rectify this.

The actual numeric metric is calculated by considering the ratio of width and height between the design's original size and the size of the screen context. It is, of course, ideal when this ratio equals 1 for each so the metric is maximal (1.0) at that point. To determine the rate of change for the metric, we introduce a tolerance factor (f) that defines the ratio of design size to screen size which where the metric will be minimal (0.0). The size metric is modeled as a linear function between these two points, as depicted in Figure 3-7. Scores for the width and height are computed separately and the metric is reached by taking the product of these two scores.

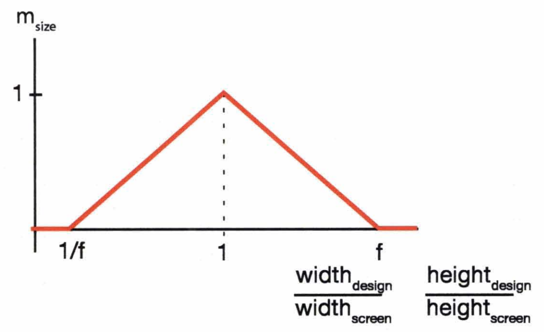


Figure 3-7: The size metric is computed by considering the ratio of both width and height. The metric is maximized when the sizes are equal and falls off in a linear fashion the more they differ.

Metric #2: Deviations in Aspect Ratio

Aspect ratio is a major metric in determining the fit of a particular design. Graphics configured for a landscape orientation do not often remain effective when transformed non-uniformly into portrait. However, our Gestalt tendency to simplify what we see does give us some leniency in applying slight non-uniform scaling. The degree to which a particular design can be distorted will certainly depend almost exclusively on the content, but for the sake of a scoring metric we can assume that the best solution is the one that deviates the least from it's original aspect ratio. Like the size metric, the original or native aspect ratio is taken from the vector graphics encoded framing rectangle [1]. The metric itself is computed as being ideal (or 1.0) for a range of distortion the eye is unlikely to notice. As the aspect ratio deviates from the ideal range, the score falls off with exponential decay. This permits the score to fall off quickly outside of the ideal band, but still favor designs that are closest to matching the aspect ratio. Without the exponential fall off, all non-ideal designs would receive the same score for this metric which could lead to selection of a design that has a badly misappropriate aspect ratio.

[1]For PDF, the media box is used as the frame. For SVG, the width and height elements are used from the document element

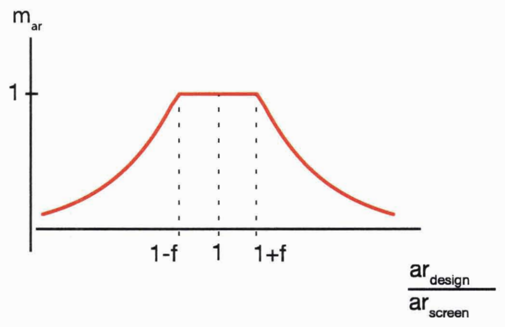


Figure 3-8: The metric for aspect ratio considers an ideal band outside of which the score decreases exponentially.

Metric #3: Percent of Appropriately Sized Objects

Another telling metric considered by the selection engine is captured in the data structures that are used to carry out the rendering itself. Vector graphics are generally expressed as a collection of graphical primitives. To facility the layout and rasterization, the objects are arranged in a *render tree* and transformed to the appropriate size and position. Rasterization is then accomplished by traversing the tree recursively drawing each object as pixels into the image. At the completion of the tree traversal, the image will contain the entire vector graphic rendered as pixels. Since the render tree contains each of the objects at the sizes they will be depicted in the resulting image, the information is valuable in selecting the most appropriate design.

Direct calculation of this metric is carried out by simply walking the render tree and analyzing the geometric bounds of every leaf node (primitive graphic object). If the area of a particular object is less than a configured threshold, then the object is considered unlikely to be discernible to viewers of the graphic. It is not enough, however, to invalidate an entire document based on a single object being below the threshold. Many author mistakes in vector graphic editing tools result in negligible sized objects being unnoticeably created within documents. We, therefore, consider the percentage of objects that fall within tolerable size guidelines. The tolerance factor is a factor of the device's dots-per-inch (DPI) and is generally taken as the smallest number of pixels an object has to span to be distinguishable. This, of course, does not take into account two other important issues. First, complex objects might require a larger span than simpler objects. And secondly, there is no consideration for occluded or clipped objects.

Metric #3: Entropy Change in Raster Images

The final considered metric tries to contend with the issue of embedded images by considering changes in the theoretical information content. Information (or Shannon) entropy provides a description of the amount of randomness, or uncertainty, that exists in a signal. Alternatively, the concept can also be said to indicate the amount of information carried by a signal. If a communications signal is viewed as being a series of discrete events, where each type of event i has a probability (p_k) of occurring, the entropy (H) of that signal is given by [43],

$$H = -\sum_{k=1}^{n} p_k \cdot \log_2 p_k \tag{3.1}$$

In image processing, the symbol types are typically the possible pixel values and their probabilities are simply calculated through the color histogram of the image itself. Thus, entropy gives a ratio of bits to pixels. What we are most interested in is the effect that scaling an image has on the entropy. As an image is transformed to display in different sizes, we rely on changes in the entropy calculation as an indication of readability. Scaling to lower sizes is primarily an information reduction operation, taking a number of symbols and performing some form of combination to generate a single a new symbol. In the simplest case, a cluster of pixels is averaged into a single pixel in the scaled down version. As an illustration of this process consider what becomes of the letter "a" in Figure 3-9 as the information is reduced with a scaling operation.

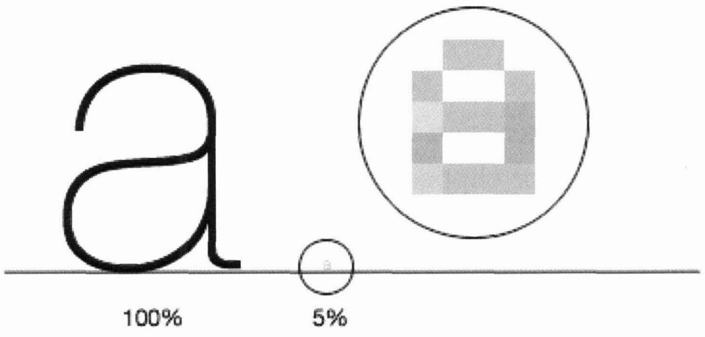


Figure 3-9: Information content is reduced as features are scaled to smaller contexts.

While the use of entropy provides a metric that works on any arbitrary image, it has the downside of varying greatly depending the contents of that image. As is illustrated in Figure 3-10, graphics with vastly different content can have very different amounts of information content. We are, therefore, unable to rely on any absolute value of entropy as a legibility cutoff. Instead, we consider the change in entropy from the raster of the native size and that of the context size. In other words, we compare the ratio of information present in the current rendering to that of the original size. The plots for the images below are typical with entropy steadily decreasing as the image is scaled to smaller sizes and finally falling off sharply at the point where whole image features disappear into single pixels. Given that the drastic drop reflects the loss of information rendered into the resulting image, it is expected that the value should have a predictable relationship to the legibility of the image's contents. The design selection engine uses the ratio of the entropy computed at the transformed size to that of the original size directly.

Overview of the Design Selection Engine

Each of the metrics described above have been normalized to a value between 0 and 1 indicating, roughly, the likelihood of how each of the metrics will impact the effectiveness of the graphic. The remaining part of the selection engine is to

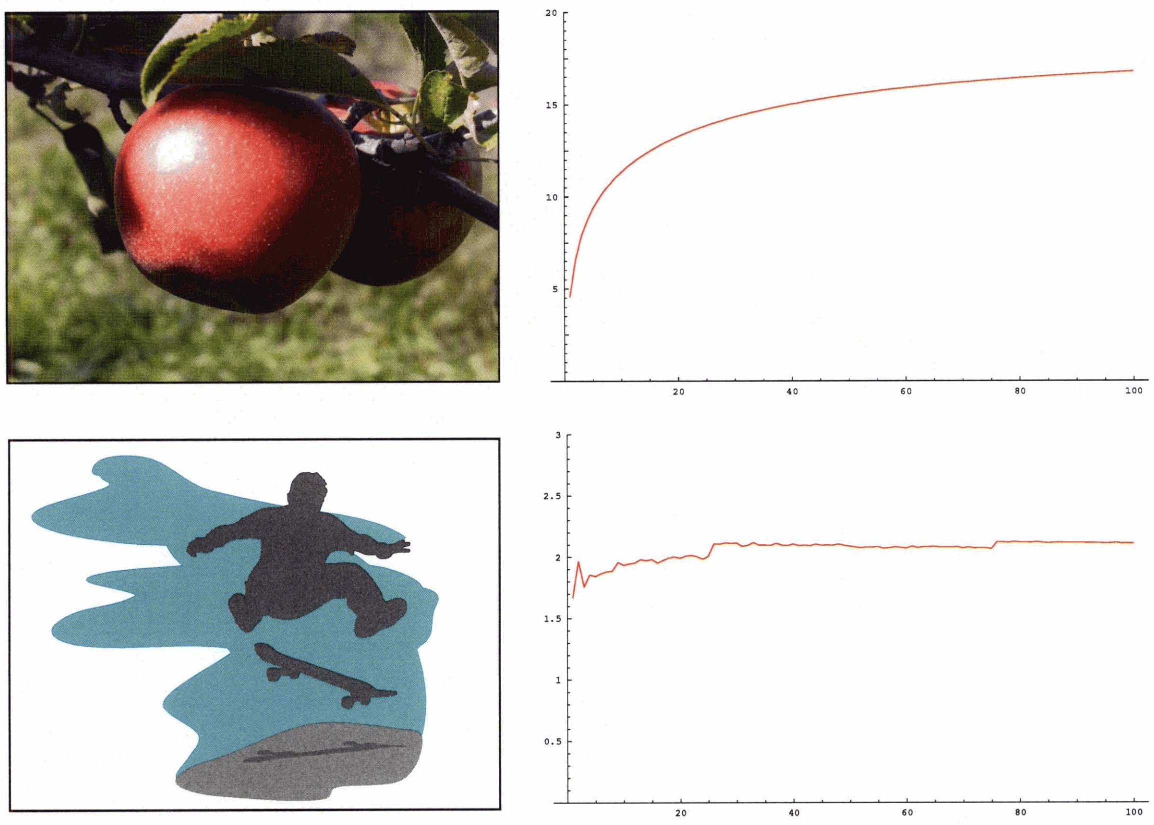


Figure 3-10: Entropy calculations change differently as different types of images are scaled down. Photographs tend to have less noisy curves as number of symbols present create very gradual loss of information.

combine the metrics into a single numeric score that correlates to the effectiveness of that graphic at that particular screen size. Selecting the appropriate design then becomes only a matter of taking the design with the highest computed score. If the highest score is below a configured threshold, the selection engine deems each of the designs as being inappropriate, renders the highest rated version, and dispatches a commissioning request to OPENSTUDIO's commissioning system with the details of the requirements.

The function for combining the metrics into an aggregate score must be considered carefully. It is not desirable, for instance, for a design to be immediately considered inappropriate based entirely on a zero score for the size metric. Aspect ratio, on the other hand, may be a stronger indication. For that reason, a weighted sum of metrics is used to reduce the values to a single score. The score for each design in a particular screen context is given as,

$$score = \sum_{i=1}^{n} f_i m_i \tag{3.2}$$

where m_i and f_i are the computed metric and a weighting factor, respectively, for each design alternative, i.

Chapter 4

Analysis

In this section, we will take a reflective look at the system presented in order to better understand its strengths and weaknesses. In particular, we consider two questions that form the crux of the thesis assertion. The first considers the assertion that the computing side of hybrid computer/human systems can more effectively evaluate existing solutions (in this case designs) than solve problems out right. Here a group of design alternatives are applied to various different screen contexts and the results of the design selection engine are evaluated for accuracy. The second part presents a thorough analysis of the design considerations that emerge when humans become the creative problem solving element in an automated system.

4.1 Design Selection and Evaluation

Evaluation of the design selection engine is carried out in two parts. First, we consider a couple of design examples in more abstract terms. Here we compute the effectiveness scores of the sample designs for a very wide range of screen sizes. This will allow us to examine the engine almost in terms of a mathematical function confirming that the trends that are present in the results match expected values.

In the second part, a number of real-world designs are used to validate the accuracy of the system for common screen sizes. This section should also give a good indication of how well the engine is able to determine the need for human redesign.

4.1.1 Evaluation Results

Here, we consider a design (shown in Figure 1-2 and also in Figure 4-4) that was presented early on in this document as an example of the type of redesign possible for different contexts. The design is presented as two different design alternatives targeted at two different ranges of screen size. For this experiment, these two design alternatives have been analyzed at all screen sizes where both the width and height are between 0 and 800 pixels. The plots are shown below (Figure 4-1 for the larger design and Figure 4-2 for the smaller one) for the effectiveness score computed by the design selection engine at each size.

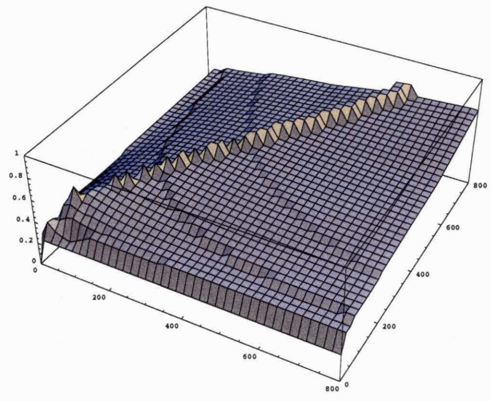


Figure 4-1: Score vs. screen size results for Design #1

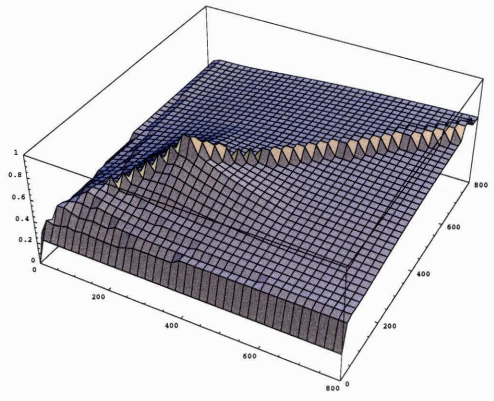


Figure 4-2: Score vs. screen size results for Design #2

Examining the results of each, a number of specific features of the plots are noticeable. First, the best fits for each design are along diagonals where the aspect ratio of the screen is close to the document's native aspect ratio. There is also a circular section in the middle of the plot for design #2 where the size metric is particularly high. This can be seen on the plot for design #2 very easily. The plot for design #1 has a similar hump, but it is not in the range of the plot as the native size for that design is 1024×768. As desired, the document is of best fit where metrics #1 and #2 are both high. The effect of metric #3 can be seen as either the width or height decrease. There is a slight slope at the higher sizes which gets very steep as either the width or height get excessively small. This, of course, eliminates sizes where features would be too small to be discernible by the viewer. Since there are no embedded images in either of these two designs, metric #4 is constant in all cases except when either width or height are exactly zero[1]. Comparing the results of the plots to what we know of the limitations of each design seems to indicate that the engine has captured a reliable set of evaluation rules.

4.1.2 Test Cases

To further evaluate the degree to which the design selection engine is capable of accurately selecting the most effective design and recognizing the need for human designer intervention, a set of test cases have been examined. Each of the designs below are based on actual designs that were used elsewhere in the last two years. In each case, the designs were added to the system as a set of design alternatives. We then dispatch requests to the rendering system asking for an image of each design at 7 different screen sizes, 1024×768, 800×600, 800×450, 400×300, 400×225, 800×300, 320×240, 40×30. Figure 4-3 gives a visual comparison of each of the different screen contexts used. The eight contexts were chosen to isolate certain situations which should give reasonably good indicators of the engine's accuracy.

1024×768 The largest screen size considered in the tests is a common industry standard known as XVGA.

800×600 This size was selected due to its real-world popularity. It also has an aspect ratio of 4:3 which is still the most common screen aspect ratio found on digital devices.

[1]These cases are by definition zero

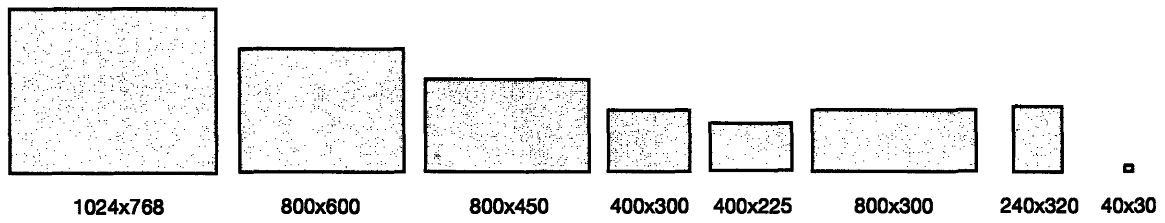


Figure 4-3: Seven different screen sizes are used for the test cases. These screens test both common screen configurations and isolated extremes to ensure that the selection engine performs well over a wide range of inputs.

800×450 Another popular aspect ratio is 16:9, this size serves as a direct comparison to the 800×600 case.

400×300 Also a size with a 4:3 aspect ratio. We consider this size as it is half the size of 800×600 case.

400×225 The 16:9 aspect ratio comparison against 400×300.

800×300 This size is taken to cover the extra wide case. This design should give a good indication of whether designs can be fit by aspect ratio.

240×320 A taller case is similar to the previous case, except in the vertical direction. This resolution was used by the very popular iPaq PDAs and is used in other mobile devices as well.

40×30 This case is considered as an indicator of how well the engine is able to determine that a design should not be used. Given the size of the documents in the test cases (Figures 4-4, 4-5, and 4-5), it is not expected that any of the designs will be effective at such a small size. The engine's results should agree.

A description of each of the cases is given below along with a table of the scores generated for each alternative using each screen size. The tables also indicate whether a request was issued to OPENSTUDIO to have a human design another alternative for the requested screen size. For the tests below, the threshold used for determining effectiveness was 0.6. Scores below that threshold are considered ineffective. Thus, any screen size where none of the designs scored higher than 0.6

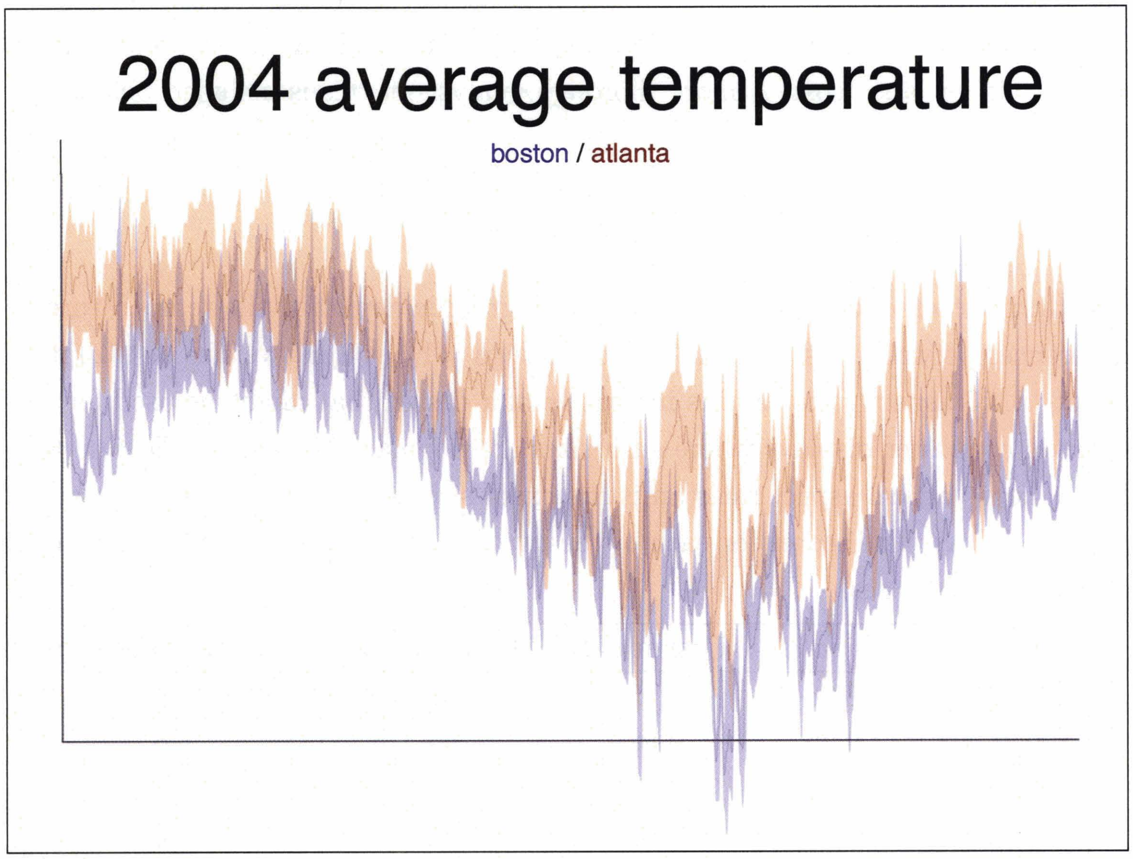
would result in a commission request being issued to OPENSTUDIO. The use of 0.6 is somewhat arbitrary as the threshold effectively becomes a quality metric. Higher thresholds will result in a lot of highly customized design, but will obviously come a higher labor cost. A more thorough analysis of this is included below.

Test Case #1: Weather Information Graphic

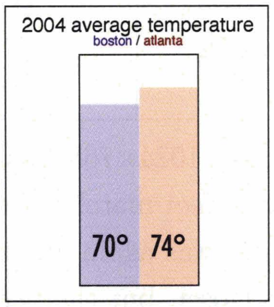
The first test case, considers an information graphic depicting the temperature differences between Boston and Atlanta. Design #1 has a native size of 1024pt × 768pt (a 4:3 aspect ratio) and is indicative of a design that might be used for desktop computer displays where such sizes are popular. The smaller Design #2 has a more vertical aspect ratio an is the exact size of the screens for a number of popular PDAs. The scores for each of the two designs is given below for each of the different screen sizes.

Size	Design #1	Design #2	Selected Design	Human Requested?
1024×768	0.96	0.62	#1	No
800×600	0.79	0.62	#1	No
800×450	0.61	0.59	#1	No
400×300	0.69	0.66	#1	No
400×225	0.56	0.62	#2	No
800×300	0.55	0.56	#2	Yes
240×320	0.69	0.71	#2	No
40×30	0.50	0.44	#1	Yes

The results are mostly as expected. 1024×768 was best filled by the design #1 as the aspect ratio and size were a perfect match. The 800×600 case also matched well with design #1 as the aspect ratio was again an exact match. The request for 400×300 brings the size metrics closer, but the perfect fit for the aspect ratio is enough that design #1 is still preferred. This is not the case with the comparable 16:9 option where design #2 is favored. When the request is made for the wide screen (800×300), the engine determines that neither is suitable and issues a request to OPENSTUDIO. This seems logical as neither of the designs are a good



Design #1: 1024pt x 768pt



Design #2: 240pt x 268pt

Figure 4-4: Design alternative set used for test case #1

fit either in terms of aspect ratio or native size. The taller screen, however, can be accommodated as it is similar in aspect ratio to design #2 which was originally designed for the portrait layout screens of PDAs. Finally, the extra small cases proves ineffective for both designs thanks to metric #3. It should be noted also that for the 40×30 screen design #1 still fits perfectly in terms of aspect ratio. However, the difference in size and the percent of features that have become too small ensures that it evaluates as ineffective.

Test Case #2: Graphic with embedded images

The second test case considers a poster for our research group and a web graphic that was later created to mimic it in smaller form (Figure ??). This design is also considered because it contains mostly embedded images, so it should give some indication if the entropy metric is performing as expected.

Size	Design #1	Design #2	Selected Design	Human Requested?
1024×768	0.54	0.26	#1	Yes
800×600	0.70	0.26	#1	No
800×450	0.43	0.26	#1	Yes
400×300	0.58	0.26	#1	Yes
400×225	0.50	0.27	#1	Yes
800×300	0.32	0.29	#1	Yes
240×320	0.56	0.25	#1	Yes
40×30	0.31	0.26	#1	Yes

Again, the engine performed well within expectations. The only context that can be accommodated is the 800×600 base case. In scaling to the 400×300 size, the entropy of the images drops just enough to invalidate design #1. Design #2 does quite poor in all the contexts which is to be expected given its odd aspect ratio and size. However, it is worth noticing that it does best on the extra wide screen. And again, the 40×30 case yields a human request and very low scores.

Test Case #3: Advertisement

The final test case uses a design with three alternatives. The design was used to advertise a programming contest and was published in a number of different places, both electronic and print (Figure 4-6). The designs vary greatly in size as two of the designs were originally intended for print, design #1 on letter size paper and design #3 on tabloid. The use of advertisements here demonstrates another very useful application. Of all the places on the web where a message needs to be reconfigured to work in a number of different sizes and formats, advertisements generally require the most work by designers. There also exist many different well-established content delivery infrastructures for delivering ads based on many different considerations. Those in the business of delivery ads are certainly faced increasingly with the problems posed in this thesis.

Size	Design #1	Design #2	Design #3	Selected Design	Human Requested?
1024×768	0.59	0.63	0.69	#3	No
800×600	0.64	0.64	0.71	#3	No
800×450	0.58	0.62	0.62	#3	No
400×300	0.59	0.81	0.62	#2	No
400×225	0.57	0.71	0.59	#2	No
800×300	0.53	0.57	0.55	#2	Yes
240×320	0.84	0.61	0.65	#1	No
40×30	0.34	0.47	0.37	#2	Yes

The results of the final case are in line with the previous results. The 800×600 and 1024×768cases here can actually be accommodated by all the designs, although design #3 is a clear winner in terms of aspect ratio. It is questionable whether design #2 would provide a decent design solution as it both the size and aspect ratio are not well suited. It is, however, very close to the threshold. And as discussed earlier, the threshold can be increased to ensure higher quality content is delivered. The only context that is not accommodated by any of the designs is again the extra wide screen. This is not an entirely unexpected situation. In fact, considering the designs in all the test cases it is clear that not many designs were suited to the extra wide scenario. Also, as we had hoped, the 40×30 case is clearly not a fit for any of the designs.

4.2 Hybrid Computer/Human Systems

The second part of the analysis will focus on the human side of hybrid systems. The notion of computers and humans working as part of the same system introduces a new sets of dynamics in terms of cost, performance and quality. Below, each of these concepts is analyzed based on experiences of assembling and participating in such a system.

4.2.1 System Response

With pure computing systems, response time is generally a measure that can be predicted within reasonable bounds by considering the computational complexity of the task and the available processing resources upon which the task will be carried out. When humans begin to serve as parts of the system, however, the bounds become wider and more erratic. We consider the possibility of a response not in terms of a Gaussian distribution over a few milliseconds, but one that could expand over weeks perhaps even years. For systems with this much variance in response, we are left with two options. Either the link between human and computing systems is synchronous or asynchronous. It is also important to keep in mind that synchronous and asynchronous describes the link between human and computer in the hybrid system. The link to clients which can also be synchronous or asynchronous in either case.

Synchronous Systems

The synchronous system keeps the computing and human systems in lock-step. The computing system is unable to return a response until it is provided by the human system. As a result, clients to synchronous systems must contend with highly indeterministic response times. As a result, these systems require a new type of system thinking with uncertain execution time. MIT Media Lab researcher Brent Fitzgerald, for instance, has proposed a new paradigm of "Slow Computing" [15]. In such a system, the expectation is that execution will take place slowly, spanning months or years.

While synchronous systems seem impractical for our current models of computation, it may be well suited to other applications. Consider fields like bioinformatics or number theory where it may be reasonable to develop systems that depend on unanswered research questions. Such a system, set in motion, may remain in a suspended state for an unknown amount of time before systematically receiving a response and proceeding to the next step in execution.

Asynchronous Systems

The more practical model for hybrid systems is to decouple the human and computer systems so that the computer is free to respond in a time frame reasonable by our current system standards. The system described in this thesis is such a system as it will respond with the best design available and prompt a human to redesign rather than waiting for the human to complete the design. Both the Peekaboom [49] and Phetch [48] systems described above are also asynchronous hybrid systems as they use the humans to construct data for training and direct use by a computing system.

Asynchronous systems try to mitigate the uncertainty for the clients by keep it internal to the system. The primary trade off, however, is quality. The design selection engine, for instance, returns design solutions that it knows to be ineffective to guarantee a reasonable response time.

4.2.2 Quality Guarantees

The other side of the trade off is that of quality. This can mean different things for different types of systems. In the selective design engine, quality is the effectiveness of the design returned by the system to the screen context. In the Pfetch system, it is the descriptiveness of the natural language explanation of the image. For hybrid systems, the quality is a characteristic that is drawn from the human system. It is, in fact, the primary reason for which humans have been introduced at the system level.

4.2.3 The Economy of Hybrid Systems

The consideration of quality vs. responsiveness give rise to an economy of hybrid systems, a framework for understanding the cost structure between the two axes. As mentioned previously and depicted in Figure 4-7, quality is in the realm of the human system and responsiveness is in the realm of computing. The idea of a hybrid system is an attempt at a compromise that gives greater responsiveness to human systems and higher quality to computing systems.

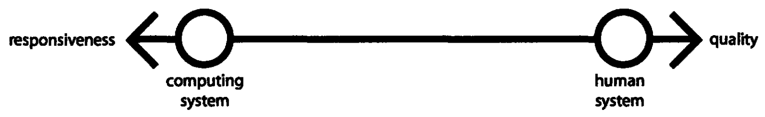


Figure 4-7: Hybrid systems must balance the trade offs of quality and responsiveness. Human systems ensure a higher level of quality whereas computing systems have more predictable responsiveness.

The economic factors enter when we consider the cost to increase either the responsiveness or the quality of the system. We argue that the two metrics cannot be moved independently without introducing a cost increase. So, for instance, the response time of the system can be increased but added cost will be introduced to maintain the current level of quality. Increasing quality without impacting response time involves keeping experts available to solve problems, keeping a pool of workers available to respond, or leveraging a trust system with human review. This situation is depicted in Figure 4-8.

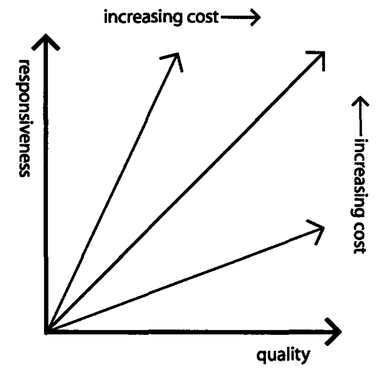
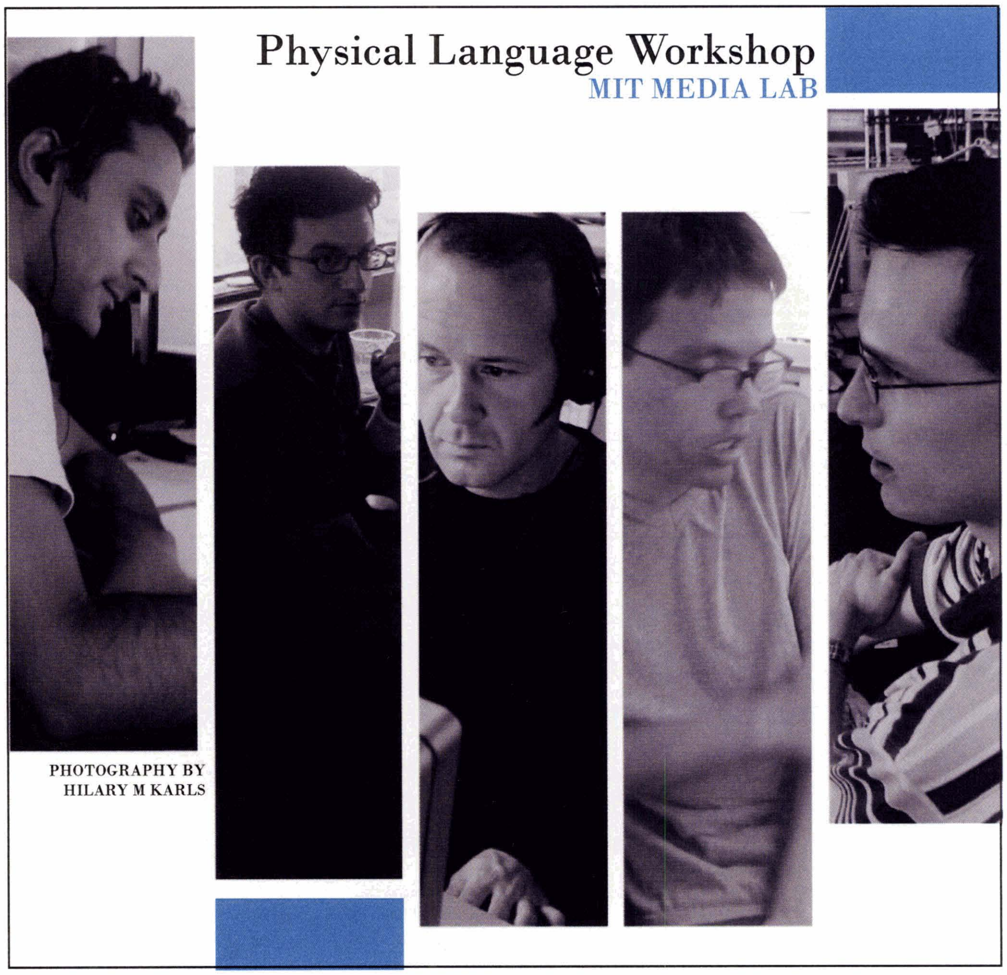


Figure 4-8: Increased cost is introduced into the system whenever quality or responsiveness is increased without effecting the other.

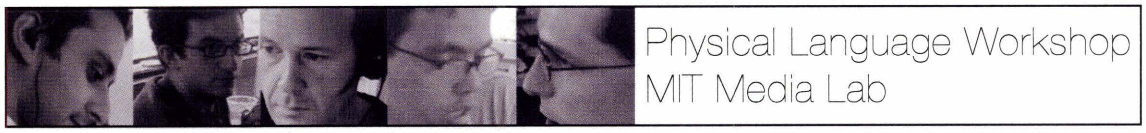
4.3 Summary

The results here demonstrate two important concepts relevant to this thesis project. First, we demonstrate the effectiveness of an algorithmic approach to evaluating creative design solutions for their effectiveness on particular devices screens. The algorithm was able to accurately select appropriate designs based on screen size and detect the cases where none of the designs are appropriate. This demonstrates the role of the computing system to mediate problem solving in an asynchronous hybrid system.

Secondly, a more detailed analysis of the economics of hybrid systems is presented in an effort to understand the elaborate cost dynamics imposed on such systems. In such systems there are two mutually constrained goals, responsiveness and quality. The cost of increasing one apart from the other introduces increasing cost to the system.

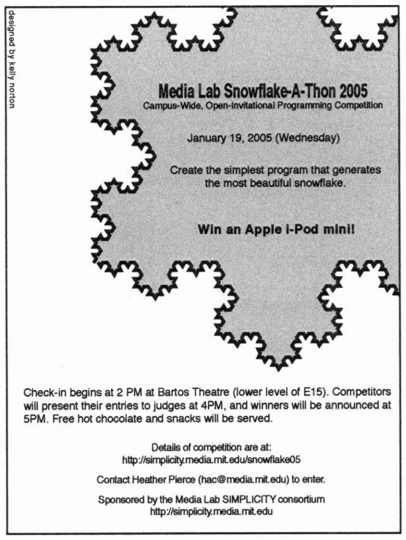


Design #1: 612pt x 590pt



Design #2: 700pt x 73pt

Figure 4-5: Design alternative set used in test case #2



designed by kelly norton

Media Lab Snowflake-A-Thon 2005

Campus-Wide, Open-Invitational Programming Competition

January 19, 2005 (Wednesday)

Create the simplest program that generates the most beautiful snowflake.

Win an Apple i-Pod mini!

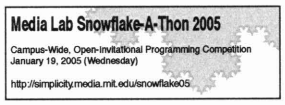
Check-in begins at 2 PM at Bartos Theatre (lower level of E15). Competitors will present their entries to judges at 4PM, and winners will be announced at 5PM. Free hot chocolate and snacks will be served.

Details of competition are at:
http://simplicity.media.mit.edu/snowflake05

Contact Heather Pierce (hac@media.mit.edu) to enter.

Sponsored by the Media Lab SIMPLICITY consortium
http://simplicity.media.mit.edu

Design #1: 432pt x 576pt

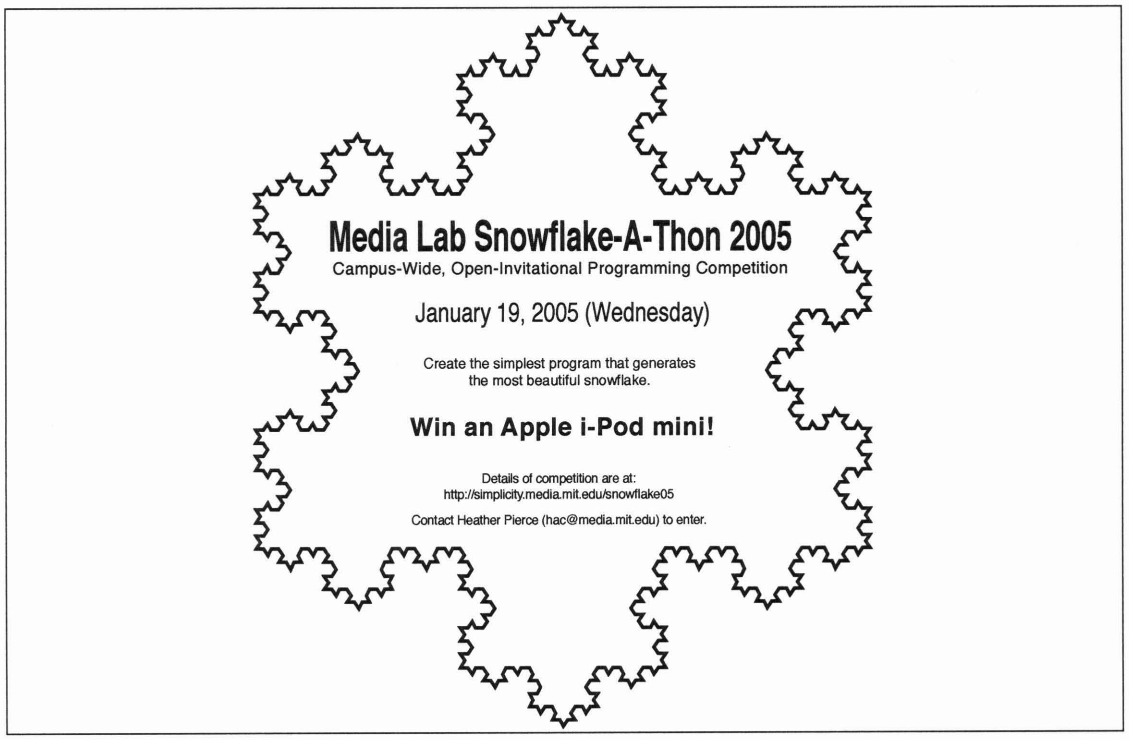


Media Lab Snowflake-A-Thon 2005

Campus-Wide, Open-Invitational Programming Competition
January 19, 2005 (Wednesday)

http://simplicity.media.mit.edu/snowflake05

Design #2: 300pt x 100pt



Media Lab Snowflake-A-Thon 2005

Campus-Wide, Open-Invitational Programming Competition

January 19, 2005 (Wednesday)

Create the simplest program that generates
the most beautiful snowflake.

Win an Apple i-Pod mini!

Details of competition are at:
http://simplicity.media.mit.edu/snowflake05

Contact Heather Pierce (hac@media.mit.edu) to enter.

Design #3: 1224pt x 792pt

Figure 4-6: Design alternative set used in test case #3

Chapter 5

Conclusions

5.1 Summary

The presence of a vast number of electronic devices for displaying digital content has made the idea of universal design for every screen size an impossibility. As the number of mobile and consumer electronic devices that leverage the stream of content generated by electronic publishers continues to increase, the demands on those publishers increases exponentially. In order to keep up with the demands imposed on content production, new systems are required. While there has been much past work on making such systems purely automated computing systems, their performance as evaluated from a graphic design standpoint has been disappointing except in overly specific cases. In proposing a solution to this problem, we seek the quality of work only possible by direct human creativity along side the formalized and systematic responsiveness of an automated computing system.

This thesis presents a system for automated adaptation of the design of digital documents that leverages human creativity as the system intelligence. A system, which is built as part of a network-based content delivery system, receives requests to render documents for a particular screen-based device. The system then uses an algorithmic approach to determining the best design among a set of design alternatives for that particular screen. When none of the designs are appropriate, the system is able to recognize this and tie into a commissioning system in order to engage human labor to prepare a suitable design for the screen for which there was previously no suitable alternative. The work here presents and validates the design evaluation algorithm and demonstrates the use of OPENSTUDIO as a creative community of human artists available to accommodate the creative labor needs of the system.

A more general discussion of the adaptive content delivery system as a new type of hybrid computer/human system is also discussed. Such systems pair computing resources with human resources into a single system that operates on the formalized input of computing systems but leverages the creative intelligence of humans. Hybrid systems either favor conventional computational models and make reasonable guarantees on response time, as in asynchronous systems, or they favor the additional quality of response introduced by human intelligence, in synchronous systems. These two axes, responsivness and quality of response form the basis of a cost economy in such systems.

5.2 Lessons

When early development of the rendering infrastructure began, there was a push to support elaborate imaging processes behind the scenes. The distributed nodes for compositing are an example of one of these processes that actually made it to the implementation stage. As users began to rely on the system for the generation of images, it was apparent that the more complex parts of the system were under utilized. In many cases, developers implemented redundant code to carry out operations that could be easily requested from the renderer. As it turned out, the fact that the complex system was veiled by its own isolation on a server hidden away in the closet, made it more difficult for developers to understand the more complex parts. The lesson here is that the isolated and remote nature of server-based software often forms a barrier to adoption. It is important to find clever ways to communicate clearly the mechanisms that are at work so that users fully understand the capabilities.

Along those same lines, we found that any extra effort that can be put into reducing the barrier to entry for using services like the renderer is time well spent. When the renderer first went live, driving the mobile photo browser and later the eReader software, it required the use of SMPL protocols to make requests to it. While this seemed an appropriate enough situation at the time, it later proved to be the least used means of interfacing to the system. Instead, users connected using a PHP-based page that could simply be accessed through a standard browser. Access in this manner limited what could be requested from the system and added extra latency to the request, but its simplicity coupled with its portability made it an immediate hit with other members of the team. Sometimes the thing that is so simple that it borders on dump is the thing that users most appreciate.

Along the way, we also learned a lot about humans as problem solvers. When the design selection engine was emerging, it was not a selection engine at all. In fact, in a lot of ways, it strived to be another system for automated design. The design always leveraged human invention, but early revisions called for humans only after evaluations of automated solutions had failed. We later realized that we were paying homage to a techno-ideology of trying to automate everything in the world and that human inventiveness was the real win. In fact, this taught us something about problem solving in general. Too many times, problems are solved by persistent head wringing. One finds a quiet place, a scratch pad and tries to attack the problem with more vigor than ever before. Such efforts usually lead to incremental improvements in the world around us. To find breakthrough solutions, one must abandon focus and instead look at the things that are taken for granted. The obvious, in our case, is that humans do better graphic design than computers. Why then are we so insistent that computers do it for us?

5.3 Future Work

While this thesis reflects on this project at a crucial milestone, there are a number of areas we would like to continue to explore. These include both incremental improvements to the rendering system and further explorations in the use of hybrid systems.

Much of the work to add the adaptive design module to the rendering infrastructure required rapid prototyping with parts that were already in play as part of the larger body of work being carried out by the Physical Language Workshop. Hire-a-designer is a key example. It provided a general commissioning system which was very easy to tie into as a means to validate our ideas. Its generality, however, makes the practical use of the system tedious. Many of the parameters of the request made to humans had to be encoded in natural language descriptions. As a result, we had no way of validating that the response did in fact match the request. For many of these general systems, a specific instance is need to address the special needs of the hybrid system.

The use of hybrid systems to solve other problems is also an interest for further exploration. We are convinced that these models apply to many more problem domains and we're eager to try them out. One opportunity for exploration is a project being lead by Brent Fitzgerald to formalize social contracts into executable objects that are maintained by a computing system. His ideas stretch the hybrid model in revealing new ways and should teach us more, in general, about how to go about making such systems successful and practical.

Bibliography

[1] `http://www.guru.com/`.

[2] `http://www.smarterwork.com/`.

[3] `http://www.elance.com/`.

[4] Apache http server project. `http://httpd.apache.org/`.

[5] Cocoa. `http://developer.apple.com/cocoa/`.

[6] Flickr. `http://www.flickr.com/`.

[7] Google answers. `http://answers.google.com/answers/`.

[8] Java foundation classes (JFC/Swing). `http://java.sun.com/products/jfc/`.

[9] Microsoft .NET framework windows forms. `http://www.windowsforms.net/`.

[10] One laptop per child. `http://laptop.org`.

[11] Openstudio: An experiment in creativity, collaboration and capitalism. `http://openstudio.media.mit.edu/`.

[12] Opera's small-screen rendering^{TM}. `http://www.opera.com/products/mobile/smallscreen/`.

[13] Scalable vector graphics (SVG). `http://www.flickr.com/`.

[14] Why PDF? `http://www.adobe.com/products/acrobat/adobepdf.html`.

[15] Slow computing. `http://www.brentfitzgerald.com/2006/04/17/slow-computing/`, April 2006.

[16] R. Beach and M. Stone. Graphical style – towards high quality illustrations. In *SIGGRAPH '83: Proceedings of the 10th annual conference on Computer graphics and interactive techniques*, pages 127–135, New York, NY, USA, 1983. ACM Press.

[17] R. J. Beach. *Setting Tables and Illustrations with Style.* PhD thesis, Department of Computer Science, University of Waterloo, 1985.

[18] S. M. Casner. A task-analytic approach to the automated design of graphic presentations. In *ACM Transactions on Graphics*, volume 10, pages 111–151, April 1991.

[19] Jinlin Chen, Baoyao Zhou, Jin Shi, HongJiang Zhang, and Qiu Fengwu. Function-based object model towards website adaptation. In *World Wide Web*, pages 587–596, 2001.

[20] Yu Chen, Wei-Ying Ma, and Hong-Jiang Zhang. Detecting web page structure for adaptive viewing on small form factor devices. In *Proceedings of the 12th International conference on World Wide Web*, pages 225–233, 2003.

[21] W. S. Cleveland and R. McGill. Graphical perception: Theory, experimentation, and application of the development of graphical methods. *Journal of American Statistical Association*, 79(387):531–554, Sept. 1984.

[22] I. Cooper and R. Shufflebotham. PDA web browsers: Implementation issues. Technical Report 11-95*, University of Kent, Canterbury, UK, 1995.

[23] R. D. Coyne, M. A. Rosenman, Radford A. D., M. Balachandran, and J. S. Gero. *Knowlege-Based Design Systems*. Addison-Wesley, 1990.

[24] A. Fox, S. Gribble, Y. Chawathe, and E. Brewer. Adapting to network and client variation using infrastructural proxies: Lessons and perspectives. *A special issue of IEEE Personal Communications on Adaptation*, 1998.

[25] K. Gareis. elancing – the future of work?
`http://www.janus-eu.org/Documents/EBEW/eLancing.pdf`, 2005.

[26] S. Gnanamgari. *Information presentation through default displays*. PhD thesis, Department of Decision Sciences, The Wharton School, University of Pennsylvania, Philadelphia, PA, May 1981.

[27] L. Gonick. *The Cartoon Guide to Computer Science*. Barnes and Noble Books, New York, 1983.

[28] X. Gu, J. Chen, W. Ma, and G. Chen. Visual based content understanding towards web adaptation. In *Proceedings of AH-2002*, 2002.

[29] Lynda Hardman and Jacco van Ossenbruggen. A graphic design perspective on processing delivery context.

[30] R.G. Kermode, H.N. Holtzman, and A.B. Lippman. The media bank: A fault tolerant heterogeneous distributed pull based architecture for the delivery of object oriented multimedia across packet switched networks. In *Proceedings of the 7th International Workshop on Packet Video*, Bisbane, Australia, March 1996.

[31] R.J. Laubacher and T.W. Malone. Retreat of the firm and the rise of the guilds: The employment relationship in an age of virtual business. Working Paper 033.

[32] R.J. Laubacher and T.W. Malone. Two scenarios for 21st century organizations: Shifting networks of small firms or all-encompassing 'virtual countries'? *MIT Sloan Management Review*, 1997.

[33] H. Lie and B. Bos. Cascading style sheets. WWW Consortium, September 1996.

[34] S. Lok and S. Feiner. A survey of automated layout techniques for information presentations. In *Proceedings of SmartGraphics 2001*, Mar. 2001.

[35] Conlin M. And now, the just-in-time employee. *Business Week*, 28 August:94–95, 2000.

[36] W. Ma, B. Shen, and J. Brassil. Content service networks: The architecture and protocol. In *Proceedings of the WCW '01*, Boston, MA, 2001.

[37] Jock Macinlay. Automating the design of graphical presentations of relational information. In *ACM Transactions on Graphics*, volume 5, 1986.

[38] T. Malone and R.J. Laubacher. The dawn of the e-lance economy. *Harvard Business Review*, 76(5):144–152, September-October 1998.

[39] M. McLuhan. *The Essential McLuhan*, volume 13, chapter 13, pages 53–74. Basic Books, New York, 1995.

[40] J. Müller-Brockmann. *Grid Systems in Graphic Design.* Arthur Niggli Publishers, Niederteufen, Switzerland, 1981.

[41] A. Ortega, F. Carignano, S. Ayer, and M. Vetterli. Soft caching: web cache managment techniques for images. In *IEEE First Workshop on Multimedia Signal Processing*, pages 475–480, Princeton, NJ, June 1997.

[42] C. Rocha. Smpl: A network architecture for collaborative distributed services. Master's thesis, Massachusetts, 2005.

[43] John C. Russ. *The Image Processing Handbook.* CRC Press, 2nd edition, 1994.

[44] J.R. Smith, R. Mohan, and C.S. Li. Scalable multimedia delivery for pervasive computing. In *Proc. ACM Multimedia*, pages 131–140, Orlando, Florida, October 1999.

[45] L. Teodosio and W. Bender. Salient video stills: content and context preserved. In *Proceedings of the first ACM international conference on Multimedia*, pages 39–46, 1993.

[46] E. R. Tufte. *The Visual Display of Quantitative Information.* Graphic Press, Cheshire, CT, 1983.

[47] L. von Ahn, M. Blum, N. Hopper, and J. Langford. Captcha: Using hard ai problems for security. In *Proceedings of Eurocrypt*, pages 294–311, 2003.

[48] L. von Ahn, S. Ginosar, M. Kedia, R. Liu, and M. Blum. Improving accessibility of the web with a computer game. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 79–82, 2006.

[49] L. von Ahn, R. Liu, and M. Blum. Peekaboom: A game for locating objects in images. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 55–64, 2006.

[50] L. Weitzman and K. Wittenburg. Relational grammars for interactive design. In Ephraim P. Glinert and Kai A. Olsen, editors, *Proc. IEEE Symp. Visual Languages*, pages 4–11. IEEE Computer Society, 1993.

[51] L. Weitzman and K. Wittenburg. Automatic presentation of multimedia documents using relational grammars. In *ACM Multimedia*, pages 443–451, 1994.

[52] K. Wittenburg and L. Weitzman. Relational grammars: Theory and practice in a visual language interface for process modeling. In K. Marriott and B. Meyer, editors, *Visual Language Theory.* Springer-Verlag, 1998.