Abstract

Virtual environments provide nearly limitless opportunity for creative organization and presentation of digital information. Unfortunately, the current standards of user interface design predominantly revolve around a highly constrained and conventional desktop environment. This project explores alternative, visually rich and interactively dynamic, information environments, and in particular focus on application of such environments to the problem of conceptual sketching in architectural design. To this end, I develop a framework for ‘meta-sketching’, in which metadata and organizational schemes are overlain on the Mental Canvas 3D construction environment developed previously. This framework is built upon of a set of primitive data components that are linked together in a modular, polymorphic fashion. These structures are then organized in a hierarchical set of 2D and 3D environments that draw inspiration from the physical world and break away from the canonical desktop metaphor. As a final step, sketch space visualization and search techniques are developed to fluidly and interactively present a design project to its users, clients, and students. The divergence of the meta-sketching environment from traditional computer aided design (CAD) is motivated by the literature on human visual perception and the specific needs of artists and architects. An implementation of this framework is currently in progress. When complete, this implementation, in combination with the Mental Canvas system I worked on previously, will provide a complete design environment for the conceptualization stage of architectural projects. In addition, the framework is made to be flexible enough that it could theoretically be more applied to a wider range of problems in information management and visualization.

Introduction

Freeform three-dimensional sketching in virtual environments has recently become a major focus of research in computer graphics (e.g. Zeleznik et al. 2006, MacIntyre et al. 2004, and Ju et al. 2007). However, relatively little work has been done on the complementary problem of providing ways to organize and interact with groups of these sketches. Here I present a framework for such interactions, with particular regard to the architectural design process. This paper serves primarily to place in context, motivate, and document an in progress implementation of this framework.

Broad itinerary of which the current project is one part

One principle use of sketching is to visually express the core of an idea. While other graphical techniques (such as drafting, doodling, painting, modeling, etc) may be used primarily for technical, aesthetic, artistic, or didactic purposes, sketching in the present context can be seen as the visual medium that most directly works with the formulation and concise storage of containable ideas. In this light, the present project is part of a broader itinerary to develop virtual environments for general idea conceptualization and organization.
An example of one such system that is presently ubiquitous, is the standard desktop environment and file system of major operating systems. These interfaces organize and present a user’s data. While robust and flexible, such systems are in general quite lacking in their capacity to attach metadata to the organizational structure, and in their ability to render the stored information in task specific ways. Online information organization schemes often to a better job in this regard. This is probably largely because without such metadata, navigating the immensity of the Internet would be impossible. Image tagging, search databases, markup languages, and dynamic webpage parsers have all proven successful steps toward better information management and navigation. As such, these techniques could likely all be usefully incorporated into other virtual environments.

The present work attempts to apply such techniques, as well as new ones, to more graphically immersive environments. The end goal of the broad project, then, would be to develop fluid, graphical landscapes in which to organize ideas and view their evolutions.

**Current project goals**

The current project consists of two primary pieces that compose a virtual environment for conceptual level architectural design. First, 2D and 3D sketching systems are provided for the actual construction of the design. These environments are described in part in my CS 490a report and further in Dorsey et al. (2007).

Second, a ‘meta-sketching’ environment encapsulates the construction interfaces. The goal of this layer is to create hierarchical, modular, and immersive, 2D and 3D spaces in which to annotate, organize, and re-visualize the set of sketches whose union describes a design.

The purpose of the meta-environment is three-fold. First, it organizes ideas and projects for the creator. Second, it presents projects to clients. And third it explicates the creative process behind the design to students and other curious observers.

**Background and Motivating Theory**

The present project draws on multidisciplinary roots in the hope that such a convergence will result in a flexible and widely applicable design environment, which will address the specific needs of all who interact with it. Thus, the design framework is motivated by the requirements and preferences of artists and architects, the organizational lessons of operating systems and software engineering, the general capacities of human cognitive psychology, and the developments of related research in computer graphics.

**Requirements from architectural and artistic design**

Despite widespread adoption of computer aided design (CAD) for drafting, the conceptual stage of architectural remains largely done by hand. This is most likely due to an inverse relationship between the capabilities of virtual tools and the capabilities of physical tools. While computers aid immensely in data storage, sharing, precision, and extensibility, physical art supplies generally remain superior when it comes to ease of use, and they typically require far less overhead to set up and begin using (Zeleznik et al. 2006). Thus, physical tools have been preferable to architects for the rough, conceptual stages of design (during which the traditional capabilities of CAD systems have proven.
not only cumbersome, but moreover superfluous), while CAD systems have been preferable only during the later stages of design.

The present project attempts to address the shortcomings of traditional CAD by offering a low overhead, easy to use construction environment. Thus, focus is placed on intuitive tools and virtual realities that mimic the familiarity of physical space. The meta-sketching layer in particular attempts to identify ways in which a virtual environment could improve on physical sketching. A frequent problem in the physical organization of designs is finding a specific lost sketch and viewing it in its original descriptive context. A desired sketch may be hidden in an archive of undifferentiated sketchbooks, and, when found, the sketch may turn out to be separated from the context in which it was created. The meta-sketching environment addresses this problem by organizing sketches on a bounded, spherical surface, which is annotated with metadata to create a rich environment for intuitive retrieval of sketches in a descriptive context. In addition, the metadata of the virtual system provides a substrate on which automated search systems could be built.

One of the most important advantages pen and paper have over standard CAD systems is that they allow free-form annotation of designs. This annotation frequently consists of written notes, diagrammatic symbols, and complementary sketches. Photos, typed documents, and other multimedia inputs are also frequently filed adjacent to sketched designs. Current CAD programs do a poor job integrating such multimedia information. Their tools are often too precise and cumbersome for efficient handwritten annotation, and the programs often do not have simple facilities for loading textual documents and photographic imagery. Unlike traditional CAD systems, the meta-sketching framework described here places central importance on the flexible annotation of designs.

**Insights from operating systems and software engineering**

Data organization concepts have reached a high degree of sophistication in software engineering and especially in the field of systems programming. Just as programming environments provide an abstracted framework in which to construct applications, the meta-sketching environment provides an abstracted framework in which to construct architectural designs. Therefore, the lessons of software organization might well be applied to meta-sketching. Thus, the same paradigms that guide the coding of the meta-sketching application are provided as paradigms for designs within the application. Canonical programming concepts such as modularity, polymorphism, and hierarchical object oriented organization are all incorporated into the meta-sketching design environment.

Robust information organization reaches perhaps its pinnacle in the underlying structures of operating systems. In view of this, the meta-sketching information environment borrows the concept of pipes from Unix, and may eventually, if it is adapted to support multiple networked users, be extended to incorporate the systems concept of concurrency control.

**Insights from the cognitive science**

Psychological studies have revealed a large number of rules and parameters that govern human perception, learning, and executive function. Consideration of issues from
a psychological perspective is essential to designing systems with good usability (Frøkjær and K. Hornbæk 2008). As a graphical system, the meta-sketching project stands to gain considerably by presenting information that is easily consumable by the human visual system. Additionally, as an interactive system, it stands to gain by providing controls that are intuitively tractable by the human executive system.

Numerous studies (e.g. Atkinson et al. 1996, Scholl and Xu 2001) have shown that humans can on average process at most four items of any given set at once. This ‘magic number four’ should be taken into consideration when developing virtual environments. In the present meta-sketching system, it is incorporated as an ideal (though not strictly enforced) limit on the number of tools available to a user at any given time, and on the number of types of module the user will interact with at once. To this end, the meta-sketching system assigns all methods of interaction to the particular modular context in which they occur. Thus, the user will only ever have available a few of contextual controls at once. However, by encapsulating tools into hierarchical blocks, it will be possible for the user to be aware of many more options for action without overloading his or her cognitive system. This is because grouping items into a categorical set causes them to be cognitively processed as a single unit. Thus, only four categories will be present at a given level of abstraction, but many embedded abstractions will be available for implicit consumption.

Another cognitive mechanism motivates the layout of the sketch landscape. Contextual cueing is a method by which quick recall is promoted when objects are viewed in repetitive, feature rich spatial contexts (Chun and Jiang 1998). In light of this capacity, the meta-sketching environment provides a physically based layout in which modules placed by a user will remain in the same spatial context until possibly moved by the user at a later time. This structure differs from standard file managers, in which windows pop up out of nowhere into variable spatial contexts, and files are frequently automatically rearranged in directory lists with sparse spatial cues.

**Related work in computer graphics**

There has recently been significant research activity on sketch-based graphical systems. This work has applied sketching to a wide range of topics, including quick 3D modeling (e.g. Zeleznik et al. 2006, Nealen et al. 2007) and model editing (Ju et al. 2007), augmented reality (Maclntyre et al. 2004), 2D sketch parsing and editing (Saund et al. 2002), user input (LaViola and Zeleznik 2004), and image search (Fonseca and Jorge 2003).

These and many other papers thoroughly explore the problems and capabilities of individual sketches and sketch-based interfaces. Less work has tackled the meta-sketching problem of designing comprehensive environments for sketch organization and conceptual project management.

At the same time, user interface work has progressed significantly away from the standard desktop metaphor. Multitouch zooming user interfaces have broken out of academia (e.g. Han 2005) and into commercial devices (e.g. the Apple iPhone and the Microsoft Surface). At the same time, physically-based interfaces (e.g. Agarawala and Balakrishnan 2006) and polymorphic spatial modules (Bederson et al. 1996) have been advanced to improve upon the standard desktop metaphor, and haptic interfaces have been explored to extend interaction beyond vision and audition (Harrison et al. 1998).
The current meta-sketching project sits at the intersection of the work on sketch-based design systems and dynamic user interfaces. As such, it draws principles from both sets of literature.

**Overarching Design Principles**

The above motivations have lead to a number of overarching design principles for development of the meta-sketching framework.

**Putting the human in the loop**
First, as much as possible, the human user will be kept in the meta-sketching loop. Automated solutions often do not match users’ intent. As an alternative to bulky and clumsy automation, this project attempts to give the user as much control as possible in structuring the design environment. Tools are polymorphic and flexible. Design layout is fully and explicitly specified by the user. And 3D forms in the underlying Mental Canvas construction environment coalesce into coherent structures only in the user’s mind (the 3D forms are represented as collages of 2D billboards; see Dorsey et al. 2007).

**Modular, polymorphic primitives and devices**
Second, in order to ensure maximum flexibility in addressing design problems of all types, interactions with the meta-sketching system are made modular, and separate interactions are as often as possible made polymorphically linkable. Traditional CAD suites also often aim to make themselves applicable to a wide range of design problems. Unfortunately, the solution arrived at in these systems is all too often to hack together as many problem-specific tools as possible into a single application. This leads to the bloat that plagues many modern CAD programs. They provide too many distinct tools at once, inducing cognitive strain as users subconsciously enumerate options well beyond the ease-of-use limit of four. Additionally, the individual tools of CAD suites often are quite dissimilar from each other. This phenomenon, in combination with the bloated number of tools and settings available, leads to the vast overhead required to learn and start up traditional CAD applications. As a primary goal of the present system is to rival the intuitive and quick usability of paper and pen, the current project eliminates such bloat wherever possible. As an alternative, the meta-sketching system provides a highly hierarchical and contextual set of tools and interfaces.

**Physically-based interactions**
Cognitive research (e.g. Chun and Jiang 1998) and an analysis of successfully easy to use user interfaces (e.g. LaViola and Zeleznik 2004, Han 2005, and Agarawala and Balakrishnan 2006) reveal that physically-based virtual environments provide an intuitive and powerful way to interact with designs. Such a physical basis is incorporated into the meta-sketching framework in the following ways. First, all information is spatially laid out in at least two dimensions. The confusing and feature poor metaphors of standard file systems are left out in place of a consistent spatial data hierarchy. Second, input commands are made gestural where possible, and familiar physical cues such as inertia and solidity are preserved in the movements of the virtual interfaces.
The Meta-Sketching Framework

Design Overview
The meta-sketching framework consists of several modular structures that interact in well defined ways, and combine to allow the user to build hierarchical design spaces, organize sketch content within these spaces, and annotate and integrate the sketch content. These basic structures are ‘worlds’, ‘interfaces’, and ‘data primitives’. Worlds define a coordinate space in which interfaces are placed. Interfaces are modular, linkable portals to embedded modules. These embedded modules may either contain further worlds, or leaf-level data primitives. Finally, such data primitives contain the actual strokes and annotations that make up the content of the design.

Design Details

Root globe
A root globe is the standard starting world for design. The presence of a unique root helps orient the user in an otherwise potentially disorienting series of embedded spaces.

Desirable properties of a root environment for sketch organization include the following. First, there should be no limit to the size of the space. It should be possible to always add more content in any given local setting. This requirement suggests the use of a multi-scale, highly zoomable space. Second, the user should be able to view the entire space more or less at once. Lacking this property, it would be easy to quickly get lost in a large design space. As a corollary to this requirement, the design space should be bounded. Otherwise it would not be possible to view the entire space at once, and the designs themselves could get lost. Thus, it should be possible zoom in to arbitrarily constrained spaces and zoom back out to view the entire meta-sketching environment. These considerations suggest the use of a zoomable globe for the root world.

A globe is an intriguing layout environment for other reasons as well. In particular, a globe provides a familiar physical metaphor. Humans are accustomed to locating data in a geographical reference frame, and, as a two-dimensional surface, a globe is a cognitively simple space in which to organize sketch content. At the same time, a globe’s surface curvature in three-dimensional space gives interaction with it a desirable physicality. Additionally, this curvature lends some of the advantages of three-dimensional organization. Specifically, observing the globe from one point of view, the user can cognitively separate the local information from sketch content on the opposite side of the globe.

The root globe is rendered from the perspective of a world-in-hands style camera. The user may draw on and deform this globe, and may pin windows on its surface. These windows are polymorphic portals into embedded modules. For example, the user may load a scanned sketch in a window, or may make a windowed view into a 3D construction space.

By drawing on and deforming the globe, the user creates a physical landscape in which to organize and locate his or her ideas. The local landscape around content serves to cue familiar interactions with that content, and explicates the relationship of that content to adjacent data.
Hierarchical, embedded content

Each window on the globe is an interface for an embedded world. More generally, within any type of world, windows serve to embed further modules. The hierarchical structure that arises is designed to be flexible and highly customizable. Embeddable worlds include 3D construction environments, 2D zoomable surfaces, and further embedded globes. The user interacts with all these worlds in the same general ways. Each window displays a toolbar of tools that can be used on the object stored inside the window. Thus, there are no global tools – all tools are contextual in that they are attached to the specific data they manipulate. At the same time, the tools for one type of data are designed to be very similar to the tools for other types of data. This ensures an easy learning curve for tool use.

The Mental Canvas 3D construction environment is fluidly integrated into this hierarchical scheme. Within this environment, users build 3D form out of a collection of 2D canvases oriented in 3D space. The meta-sketching framework adds to Mental Canvas by treating canvases in the same way as it treats windows. Thus, canvases become interfaces to embedded modules. Typically these embedded modules will just contain strokes and texture maps, but it is worth exploring whether or not more general canvas data is useful.

Data primitives

Data primitives provide the basis on which higher level visualization and search applications function. The following data primitives are intended: timestamps, identification numbers, strokes, parameter tables, semantic annotation mapping tables, texture maps, color and brush selectors, text boxes, command prompts, and links. These primitives are interfaced by windows and canvases.

Links

Links are perhaps the most important type of data primitive. These structures are used to couple together any one module to another. A link is drawn by the user as a quick, gestural pen stroke between the objects to be linked. These links may be drawn in variable styles to indicate the particular type of linkage intended. For example, drawing a directed arrow would specify a data pipe link from one module to the next, whereas drawing a line between two modules would specify a simple grouping link. Other types of links intended include tagging links, graphing links, and annotation links. When a link is formed, the system establishes an intuitive connection between the linked objects (if one exists) based on the types of the objects and the type of the link, and then renders a completed link between them.

Linkage attaches metadata to sketches. In particular, tagging links label a sketch with linked images, summary sketches, and textual fragments. Annotation links function similarly to tagging links except specify that the linked data should be overlain as a layer on top of the sketch. Then, users may paint further annotations on top of the sketch and, using the linked annotation module, tabulate which colors of paint denote which types of semantic information. Grouping links simply assign modules to a set such that they are selected as a unit and when they move they preserve constant relative positions. Graphing links are used to show modules as part of an abstracted graph structure (for example, a
tree structure could be imposed on a set of sketches in this way). Lastly, data pipe links are used to send output from one module into the input stream of another module.

**Gadgets**

Data pipe links can thus be used to create small gadgets that populate the design environment. These gadgets function much like Unix pipelines. A simple example of how these powerful gadget can be created with little automated overhead is as follows: the user creates a window, sets it as a color selector, selects a color, links the window to a 2D sketching window, and then begins drawing with the pen tool in the 2D sketching window. This results in the pen adopting the selected color. If subsequently, the user links the same color selector to a canvas creation tool in a 3D construction world, the new canvases created would be rendered in the same, previously chosen color. The color selections for each module thus remain modularly linked to those modules and can be later referred to and edited. To avoid cluttering the design environment with excessive modules of this sort, many small modules can be hidden. However they can always be later retrieved for further manipulation.

**Visualization and search applications**

Visualization and search applications are applied as a final stage on top of the constructed worlds and linked data primitives. On globe surfaces, a height map visualization is to be provided. To set this up, the user must initially link parameter data tables to the windows to be included in the map. Alternatively, the user may setup the visualizer to automatically calculate a parameterization based on automatic extraction of features within the windows. Then, based either on the values in data tables or on automatically extracted features, the windows are extruded variable distances off the globe. This visualization allows the user to view the landscape of his or her design with respect to any given parameterization.

Such visualization techniques may be applied to sketch search. For example, if the user wants to find all the sketches that are tagged as involving the design of a particular door knob, the user may setup the visualization to height map the globe based on the presence and density of these tags. The user would then be able to quickly scan the world for high points that indicate concentrations of sketches involving the door knob in question. A key advantage of this search approach is it allows the returned results to be viewed in a consistent spatial context. Unlike most traditional search applications, which return an out of context list of matches, this method ensures that the sketches remain understood in the context of their local landscapes.

A second visualizer displays the timeline of design fabrication. The user may save the global project at any time, and these saved states are stored on disk. The timeline visualizer then accesses these saved states to display a cross section of the globe with older saved globes concentrically placed inside the current globe, with the oldest at the core and younger globes further out. The user may then doubleclick on a shell to be taken to that saved state and explore it in detail.

Additionally, the globe shells may, while still in the timeline view, be height mapped as described above. This combination produces 'geological column' view of the history of the project. The progression of particular individual modules through the
global section can be traced with overlain strokes, and in this way, the parametric evolutions of these modules and their local environments can be viewed.

One last feature of the timeline view is that the user may excavate through the shells while viewing a globe from the perspective of the full 3D world-in-hands camera. Starting with an outer shell, the user would peel away particular bits of the surface to view the state of the world at the last save point underneath, and the user could continue in this way to arbitrary depths. Of course, rendering all these shells will be intensive and various forms of culling and graphical simplification will be used.

**Implementation**

An implementation of the above described meta-sketching framework is currently in progress. The design environment in many ways directly reflects the code organization. The code is categorized into four basic classes, which are supplemented by various utility, rendering, and system hooking objects. The four base classes are Controller, World, Interface, and Data Modules. Controllers are modular, polymorphic ways of controlling Worlds. Worlds organize Data Modules and Interfaces. Interfaces interface onto embedded worlds. And Data Modules provide the leaf storage of the data primitives described above. Documentation of this architecture is provided in an accompanying presentation.

**Conclusion**

The meta-sketching framework here presented offers novel ways to create highly organized design environments. Such intricate organization lends itself to parsing by visualization and search mini-applications. Several such applications were presented above, but this initial exploration is by no means exhaustive. A primary goal of the in progress implementation of the meta-sketching framework is thus to provide convenient interfaces though which a vast range modular applications might eventually be developed to hook into the environment and re-present it in ways that match the particular requirements the many specific problems in conceptual level architectural design.

**References**


