AUDIO CUES FOR GRAPHIC DESIGN

Solange Karsenty[*]*, James A. Landay* [†] *and Chris Weikart*[*]

[*]Paris Research Laboratory - Digital Equipment Corporation
85, Avenue Victor Hugo, 92500 Rueil-Malmaison, FRANCE
{karsenty,weikart}@prl.dec.com

[†] School of Computer Science - Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213, USA
landay@cs.cmu.edu

ABSTRACT

Rockit is a system for the inference of graphical constraints, embedded within a graphical editor as part of larger project on applications development. The graphical editor allows the application designer to draw application-domain-specific objects and define constraints among them. On the basis of direct manipulation techniques, Rockit can infer the most likely graphical constraints as the designer manipulates the objects. Both graphical and audio feedback are used to guide the designer towards their choices. In this short document, we present our experience in designing and using audio, as a complement to graphical feedback, for this purpose.

KEYWORDS

Sonic feedback, Audio, Geometric Constraints, Graphical Editors, Inferencing, Interaction Techniques, Direct Manipulation, User Interfaces

INTRODUCTION

Rockit [2] stands for Rapid graphical Object Constraint identifier using Knowledge Inferencing Techniques. Rockit's purpose is to automatically identify the possible graphical constraint scenarios between objects within a scene and then allow the application designer to easily apply their choices. *Rockit* is embedded within a graphical editor for 2D application construction, focusing on the "data view" of an application. The graphical editor allows the designer to create both primitive and composite (hierarchical) application objects, and establish constraints among them, by direct manipulation. Typical application objects include diagrams, circuits, and flowcharts; the targeted applications are graphical editors for these kinds of objects.

Rockit uses a rule system which encodes some simple knowledge about how graphical objects are typically associated, and thus how they might be constrained amongst one another, as in *Peridot* [4]. Object-defined *gravity fields* are used to filter out unlikely constraint scenarios between the object the user is currently manipulating and other objects in the scene. The remaining candidates are passed on to the rule system, which returns information about possible constraints. As the object is manipulated, it will *gravitate* towards the most likely scenario. If the designer is not satisfied, and wishes to see other possibilities, they can cycle through the other scenarios by clicking a mouse button, as in [3]. The designer is thus guided within, but remains firmly in control of, the design situation.

Objects, according to their type, contain a number of *constraint sites*. Each site has an associated gravity field, defined by a region (boundary) and a potential function active within this region. When the user clicks on an object to begin a drag operation, the position of the click determines which site will be considered active during the subsequent drag. As an object is dragged through some other gravity field, it is attracted (or repelled, as the case may be) according to the gravity function. In the general case, dragging an object through multiple, overlapping fields results in a composite displacement for each mouse move, according to the vector sum of all the contributions. The user intuitively senses the object gravitating towards the most likely scenario. During this process, the affected constraint sites are highlighted. In addition, two types of feedback objects are created for indicating the possible constraints, and that which is finally accepted. The system currently attempts to identify six different kinds of constraints: connectors, spacers, repulsers, annotators, containers and aligners.

COMBINING SONIC AND GRAPHICAL FEEDBACK

Purely graphical feedback is problematic, largely because it is not always easily distinguishable from the graphical objects themselves. There is no graphical style attribute, such as line thickness or dash style, "reserved" for feedback objects only. Since the nature of the task at hand is to construct graphical applications, we needed an orthogonal, sensorial dimension: audio.

Thus, we have associated a sound property to each feedback object. This property is activated by default when the object appears on the screen. Gravity fields produce a continous sound while an object is dragged through them. When a constraint is accepted, for instance when an object is snapped to the edge of another object, a brief "clack" sound is played. The targeted objects are still graphically highlighted, but the type of constraint – which may have been symbolized as a dashed line – is now also represented by a sound. With six constraint types and two states for each, there must be twelve unique sounds. Naturally, the two sounds associated to a specific constraint type are related to each other. Therefore there has to be a sonic similarity between the two of them.

AUDIO EXPERIENCES

While it is obvious that sound conveys useful information in everyday life, it is not always clear how audio can be used in the computer to improve user interfaces. It is still often considered annoying, especially in a group context. Nevertheless, within the context of our experiment, audio with graphics was a clear winner over the entirely graphical alternative.

Since our paradigm does not correspond directly to a "real-world" situation, as in Arkola [1], we needed to make decisions about which sounds seem to be the most helpful for the largest number of users. Obviously, such subjective judgements can only be vindicated by experience. We have collected sounds from two sources: some natural sounds were sampled, others were recorded from a synthetizer. Among sampled sounds, we started by recording speech because it was an obvious kind of feedback that did not require any learning. For instance we used words such as "annotation!" or "spacer!". This technique revealed two problems: such sounds were not intelligible when mixed, and sustain was not possible unless looping on the word. In our system, a sustained sound is required while an object remains in the region of a gravity field.

Mixing sounds caused a major difficulty. Designers were observed to be frequently activating several potential constraints, triggering several sounds simultaneously. By carefully choosing the sounds, and ensuring that pitches combine consonantly, we can make the feedback understandable and pleasant to listen to. We can also adjust the volume to render the intensity of the gravity field (which usually varies with distance). This particular aspect was never reflected clearly through the graphics alone, although it could be detected through the "feel" of the gravitation towards the feedback objects. As a result, we have noticed among our users a significant increase in the complexity of their graphics manipulations, and therefore the scenes being designed. Audio allows the designer to go about their business in a more natural way.

In a previous implementation, we experimented with sound interleaving instead of sound mixing. That is, sounds were played successively and repeatedly for a time proportional to the intensity of the gravity field. This technique was abandoned after user testing: interleaved sounds were not accurate enough. Futhermore, the human ear can easily capture and understand mixed sounds if they are carefully designed, as discussed above. We found, for instance, that the attack, sustain and decay can express fine grained states of interaction that can help distinguish mixed sounds.

FUTURE RESEARCH AND CONLUSION

We are happy with the dimension that audio has added to our interface. There are, nevertheless, more interesting possibilities. We imagine that quadriphonic audio could easily be mapped to a 2D workspace, the screen, by associating an audio source to each corner. Proper mixing would then allow the user to localize sounds as if they were emitted by specific objects. Another option would be to use a method such as [5], which allows localization with only a stereo source. However, since we are not trying to localize within a 3D space, this technique is probably too sophisticated for our needs.

Our implementation is based on the DECaudio board, which gives us either phone quality sound or high quality digital audio sound (48kHz) in stereo. Our experience encourages us to extend the use of audio throughout the entire project, thus allowing us to provide tools to build sonic applications.

REFERENCES

1. William W. Gaver, Randall B. Smith, and Tim O'Shea. Effective sounds in complex systems: The arkola simulation. In *CHI'91 Proceedings, Human Factors in Computing Systems*, pages 85–90. Addison Wesley, 1991.

2. Solange Karsenty, James Landay, and Chris Weikart. Inferring graphical constraints with Rockit. Research Report 17, Digital Equipment Corporation, Paris Research Laboratory, March 1992.

3. David L. Maulsby, Ian H. Witten, and Kenneth A. Kittlitz. Metamouse: Specifying graphical procedures by example. *Computer Graphics*, 23(3):127–136, July 1989. ACM SIGGRAPH '89 Conference Proceedings.

4. Brad A. Myers and William Buxton. Creating highly-interactive and graphical user interfaces by demonstration. *Computer Graphics*, 20(4):249–258, August 1986. ACM SIGGRAPH '86 Conference Proceedings.

5. Elizabeth M. Wenzel, Frederic L. Wightman, and Doris J. Kistler. Localization with non-individualized virtual acoustic display cues. In *CHI'91 Proceedings, Human Factors in Computing Systems*, pages 351–359. Addison Wesley, 1991.