Developing Visual Editors for High-Resolution Haptic Patterns

Cuartielles, D., Göransson, A., Olsson, T.  
K3 – Malmö University  
SE – 20506 MALMÖ, Sweden  
david.cuartielles@mah.se  
andreas.goransson@mah.se  
tony.olsson@mah.se  

ABSTRACT
In this article we give an overview of our iterative work in developing visual editors for creating high resolution haptic patterns to be used in wearable, haptic feedback devices.

During the past four years we have found the need to address the question of how to represent, construct and edit high resolution haptic patterns so that they translate naturally to the user’s haptic experience. To solve this question we have developed and tested several visual editors

Categories and Subject Descriptors
H.5.2 [Information Interfaces and Presentation]: User Interfaces – graphical user interfaces (GUI), natural language, prototyping.

General Terms
Design, Languages.

Keywords
Haptic language, haptic editors.

1. INTRODUCTION
A haptic vocabulary is a ‘toolbox’ containing different ways and methods for touching users. The earlier Ertogod installation (2003) [6] used a two-way bodysuit that both sensed users through input and touched them through vibrotactile stimulators. Here three kinds of touch patterns formed its haptic vocabulary. These were i) the ground, basic patterns used in every part of the installations dramaturgy, ii) the designed and specific patterns scripted as specific parts or sequences of the dramaturgy and iii) the random patterns that were generated as response to user’s touch. Patterns were made without a visual editor. To aid the making of specific touches the patterns were first laid out as drawings marking positions and movement before coded directly into the software.

Much like a textual language, a haptic language consists of a hierarchical structure where the lowest layer defines the most basic components of the language, for example the characters of a text. This most basic component is then defined in more complex structures that produce yet another level of understanding; in our textual example this would be words. Multiple second tier constructs are then combined into the communicated message.

In our visual editors we have tried to facilitate both for exact control of single vibrator outputs -analogue to characters- as well as the formation of combinations of multiple outputs that form higher level and meaningful haptic expressions analogue to ‘words’. Haptic languages can be subdivided into the alphabetic using tactile clues to form actual words (Braille, telegraph) and the conceptual, symbolic and non-verbal attempting to form meaning through emotions and embodied sensations (hand gestures and body language) [2]. In this project we focus on the symbolic expression of haptic language. Although less exact it is much faster and experienced more direct for most users.

2. RELATED WORK
In 1998 Fogg et al. developed the haptical entertainment device called HandJive [4]. While developing HandJive the authors also realized the need of a novel haptic language for their device which they named “Tactilese”. Composed of three hierarchical units – Positions, Patterns, and Routines – Tactilese lets the user create haptic messages of varying complexity which are then passed on to another player holding a similar device.

The basic units of Tactilese, Position and Pattern, were simple for users to grasp quickly. However, the more complex Routines had a much steeper learning curve and consequently users had difficulty grasping them, but given enough time they had no problem learning and performing them.

Many other works within the field of haptic interactions consider haptic languages as something immediate, a synchronous communication between two, or more users through devices equipped with haptic actuators [1-3,9,10]. The immediate communication is often initiated by an input device and received, and acted upon, by an output device. An example of this is the InTouch project by The Tangible Media Group at MIT Media Lab. Two identical roller devices in separate locations both record and exchange their movements, thus enabling their users to directly sense each other’s actions.

Similarly, haptic feedback patterns can describe the environment in novel ways [5], actively helping the user to make decisions based on the information acquired by the haptic pattern.

Through this previous research we find that haptic patterns can be recorded, and/or played, in two modalities.

i. Through direct touch on the haptic input device which then interprets that interaction and controls a haptic output device, synchronously or asynchronously. In this construction the haptic input device can be the same as the haptic output device; as in the example of HandJive, TapTap, and Huggy Pyjama.

ii. Through an indirect definition or recording of a haptic pattern which is synchronously, or asynchronously, initiated by the haptic device based on events; originating either from the device status or its context. In the example of Soundcrumbs the device had a statically stored low resolution haptic pattern which was activated by the user in a specific context as navigational aid.
The biggest difference between the two modalities is the freedom to define the haptic language, in i) the user is free to define the vocabulary and its meaning herself, in ii) the meaning is already defined for her.

3. HAPTIC PATTERN RESOLUTION
A user defined haptic language is highly dependent on the number of haptic actuators implemented in the system. Ideally the amount of haptic actuators should exceed 90 to achieve a deep enough immersion into, and correct interpretation of, the high resolution haptic pattern [6].

In our project we define the difference between low-resolution haptic patterns and high-resolution haptic patterns as the combination of the number of haptic channels and each haptic channel’s inherent data-resolution.

The garments developed in this project each had a varying number of haptic channels, ranging from 6 (Sweet) and 64 channels (Blind Theatre). Each channel had a data resolution of up to 128bit allowing us to precisely adjust the strength of the vibratory output of each channel.

4. THE PROTOTYPES
In this section we discuss the three latest iterations of the visual editor for high resolution haptic patterns. We also introduce our next version of the editor, currently in development.

4.1 First Generation Editors
The Blind Theatre editor (2009) was heavily influenced by the DMX GUI’s used by technicians at the theatre. Featuring a total of 64 haptic actuators, and with little to no natural interactions planned in the user interface this editor had a very high threshold for beginners. While the interface was very complex, it also offered a high degree of control for each haptic actuator, and therefore an overall control of the entire suit.

4.2 Sweet Editor
Sweet marked the beginning of the second generation editors. Here we made a conscious move towards a much lower resolution haptic pattern compared to what was used in previous iterations, we settled with 6 channels because our aim in this project was not to explore haptic interactions but rather develop the concept of a haptic toolkit for designers.

The environment developed in Sweet should follow the basic design principles of user interfaces: among others it should be clear, non-intimidating, and intriguing for the designer. Another important principle for Sweet was the one defined by Bret Victor in which he defines that creators need an immediate connection to their creations [8].

Therefore the visual environment of the Sweet Editor should:

i. Resemble the outcome of the wearable device on which the high-resolution haptic pattern would be applied.

We envisioned that the Sweet Editor would allow the designer to insert a drawing or a photo of the garment giving the designer an immediate, and visual, connection between the garment and the haptic pattern. The Sweet environment should also:

ii. Offer immediate connection to the outcome of the created haptic pattern.

Through the interface of the Sweet Editor the designer would be able of instantly installing, and testing, the pattern which is currently being edited. Another important concept of the Sweet Editor was also to move towards a more natural way of haptic interaction inside the programming environment – through touch.

iii. The editor should be based on a touch-enabled platform allowing for a more natural interaction when creating the haptic pattern.

As there were no suitable touch-enabled platforms to sport the interface at the time of creation we tested functionality by mimicking touch-interactions through a standard WIMP interface. This enabled us to test basic functionality, but also caused confusion for test subjects.

![Figure 1 Blind Theatre Editor](image1.png)

![Figure 2 The Sweet Editor](image2.png)

4.3 Sense Memory Editor
As cloud computing and streaming media is becoming the norm for accessing applications on mobile interfaces, we decided to move our tools and data online. This removed the need to actively install new geo-located high resolution audio-haptic patters on the mobile device and instead issue a pull command to a web database when needed.

Also notable is that we decided to temporarily abandon the touch-based editing paradigm implemented in Sweet as that caused some confusion when applied to a WIMP style interface. The detailed time-line editor offered sufficient control for initial test, but represents a time consuming activity demanding several iterations of haptic pattern testing and re-editing.
4.4 Next Generation Editor

All our editors gain detailed control over the haptic channels in combination with an interactive timeline graph. However, as haptic stimulus must be edited one after the other, this represents both a slow and non-transparent approach. In realizing this we decided that our next generation editor should lessen detailed control over the haptic pattern through a timeline, and design towards a more natural touch interaction paradigm. Our new editor creates haptic patterns by touching iconicographic representations of the body on a touch screen interface.

5. EDITOR ANALYSIS

The graphical editor iterations have been tested in several projects. Project group discussions and interviews have constructed the base for our iterations throughout the project, and from them we’ve gathered the following guiding principles for our next generation editor.

i. The editor should support both modalities of creation and consumption of haptic patterns, and support this through both through asynchronous and synchronous communication.

ii. Provide a direct feedback connection for designers working with the haptic pattern.

iii. Direct feedback to haptic pattern generation on the small smartphone screens can be enhanced both through sound and vibration.

iv. Visual editors of full body haptic patterns should include iconography representing the whole body, both in front and rear view.

6. CONCLUSIONS AND FUTURE WORK

Developing visual editor software for high resolution haptic patterns and multimodal audio-haptic sculptures is a complex and difficult task. Possible solutions should focus on natural, touch based input to form and edit haptic patterns. Our intentions are to continue this project; developing visual editors for high resolution haptic patterns which are easy to use, portable between multiple systems and provide high-resolution haptic patterns in the three most common textual data formats – XML, JSON, and CSV.

7. ACKNOWLEDGMENTS

Our thanks to ACM SIGCHI for allowing us to modify templates they had developed.

8. REFERENCES


