# The Role of Prototyping Tools for Haptic Behavior Design

Colin Swindells[1], Evgeny Maksakov[1], Karon E. MacLean[1] Victor Chung[2]

[1] Computer Science, University of British Columbia
201 – 2366 Main Mall
Vancouver, BC V6T 1Z4
{swindell, maksakov, maclean}@cs.ubc.ca

[2] Computer Science, University of Toronto
3302 – 10 King's College Road
Toronto, ON M5S 3G4
vchung@cs.toronto.edu

## Abstract

*We describe key affordances required by tools for developing haptic behaviors. Haptic icon design involves the envisioning, expression and iterative modification of haptic behavior representations. These behaviors are then rendered on a haptic device. For example, a sinusoidal force vs. position representation rendered on a haptic knob would produce the feeling of detents. Our contribution is twofold. We introduce a custom haptic icon prototyper that includes novel interaction features. We then use the lessons learnt from its development plus our experiences with many haptic devices to present and argue high-level design choices for such prototyping tools in general.*

## 1. Introduction

Haptic behaviors are touch-based interactions that represent some kind of meaning to the user. For example, a vibrotactile pattern from a cell phone can indicate a caller's identity, or a unique feeling on a haptic radio tuning knob can convey a station's genre of music. A general prototyper tool can facilitate design of many types of haptic interfaces. Different types of interaction, degrees of freedom (DOF), and dynamism could theoretically be included in an optimal haptic behavior designer.

In this paper, we show our custom haptic icon prototyper to illustrate a collection of issues related to the development of a haptic behavior design tool. We then use our lab's extensive haptic experience, and the insights gained from using this icon prototyper, to develop a set of general guidelines for an optimal icon prototyper. But first, we briefly summarize some related work.

## 2. Related Work

Many approaches can be used to prototype haptic behaviors. Collections of tactile surfaces can aid exploration of haptic behaviors for a tactile interface. For example, one could use surfaces of silk, wood, sandpaper, and metal to express how a mechatronic tactile system might feel when in particular system states. Collections of mechanical assemblies could similarly represent how a kinaesthetic haptic behavior might feel. A more theoretical approach could involve sketching waveforms or, more generally, expressing mathematical representations, of a haptic behavior over space and/or time.

We were inspired by previous work for audio and visual modalities because they contain many user interaction concepts that are relevant to haptic prototyper tools. For example, many video and audio editors offer movable and modifiable graphical icons that represent audio / visual data elements and data streams (e.g., Apple iMovie [1] and Adobe Soundtrack Pro [2]).

Waveforms representing haptic behaviors can be edited in similar ways to audio waveforms. For example, the audio icon work pioneered by Gaver [8], and documented further by Buxton et al.[5], help provide a starting point for haptic icon development. People have successfully used audio tools to create haptic effects. For example, Chang and O'Sullivan [6] used audio waveform tools to create recorded haptic icons to be played back through vibrotactile actuators in cell phones. Nevertheless, the fundamental differences between audio and haptic modalities necessitate the creation of tools tailored for haptic development.

Many movie editors and flowchart tools use palettes of icons to represent data streams and elements. For example, stencils in Microsoft Visio [3] contain rectangle, circle, and triangle shapes for building flowcharts, and palettes of icons represent film clips in Apple iMovie [1]. A haptic prototyper could similarly utilize graphical icons to represent and organize collections of haptic behaviors. Additional user interaction techniques from movie editors such as fading, merging and filtering effects are also relevant to haptic behavior design.

Although graphical user interface (GUI) tools are usually discussed when describing haptic icons development, using custom physical interfaces (i.e., tangible media) may often be more appropriate – especially in early design stages. For example, many rich physical interactions, such as physical splicing and pasting, were lost when movie editor technicians switched from editing physical rolls of film to GUI video software. Snibbe & MacLean [12] describe a set of prototyping techniques for haptically manipulating digital media.

Often ignored are visualizations of physical device limitations, or the user's psychophysical boundaries, and how these constraints might affect the rendering and perception of haptic behaviors. For example, Weir et al. [13] developed the Haptic Profile concept to visualize subtleties of switch movements (e.g., friction resistance profiles over position). Such techniques are especially important for examining differences between haptic models and physical realities, including unintended hysteresis or backlash. An example perceptual design technique is the use of 2-D multidimensional scaling plots to quantitatively show perceptual differences between several haptic icons [11]. Although perceptual sensitivity would be a useful addition to a haptic prototyper, there are many unknowns within the perceptual limitations of haptic behaviors. So, currently, the best approach for haptic prototype design is to perform perceptual user studies to compare several designed haptic behaviors after they have been developed. For example, Lee & Hannaford [10] explored haptic thresholds of a person's index finger when using a pen based haptic display under various force feedback conditions.

## 3. Example Prototyping Tool & Interface

Previous haptic icon prototypers, such as the Hapticon Editor [7] and Immersion Studio [9], are specialized GUI tools for editing torque waveforms in space or time. Within the next few sections, we describe our new custom haptic icon prototyper tool and then use it as a concrete example to support discussion of more general haptic prototyper design principles. Our icon prototyper is primarily designed for creating fast prototypes for 1 DOF haptic actuators such as knobs, sliders, pressure actuators, or temperature actuators. It also contains three major enhancements to previous work:

- The concept of 'haptic tiles' to arrange and organize collections of haptic icon primitives.
- Streamlined interaction sequences for faster iterations between haptic renderings.
- Support for dynamic haptic properties and interactions.

### 3.1. Haptic Icon Prototyper Summary

Figure 1 illustrates a screen capture of our haptic icon prototyper. It contains three main interaction regions:

*1* ***Waveform editor.*** Represents the magnitude of a haptic signal vs. space or time.

*2* ***Icon palette.*** Contains icons representing basic haptic effects.

*3* ***Tile pane.*** Enables combining basic haptic icons into more sophisticated icons.

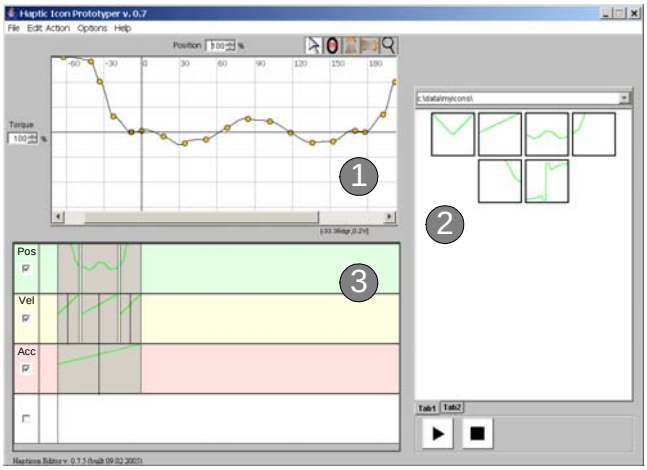


**Figure 1: Screen capture of a haptic icon prototyper illustrating the 3 main interaction regions: 1) waveform editor, 2) icon palette, and 3) tile pane**

### 3.2. Prototyping Example: A Fan Knob

A specific example of designing a knob behavior for a four setting fan is used to clarify the interaction design concepts. We chose a fan knob because it is a simple, easily understood control that suits our needs of communicating usage of the interaction design concepts (other examples might be more likely commercial targets, but are harder to explain prototyping tool concepts).

Suppose we have a fan knob as illustrated in Figure 2. When rotating the knob, the user will feel 'clicks' at the *Off*, *Lo*, *Med*, and *Hi* setting angles. Also, the knob has particular damping and inertia.

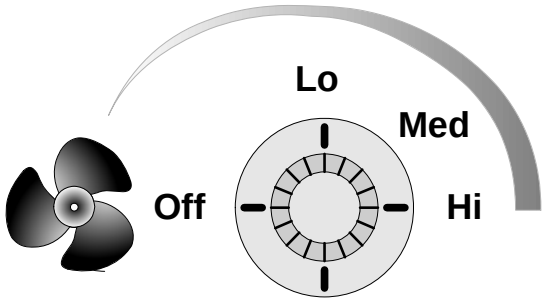


**Figure 2: Example fan knob for illustrating icon prototyper interactions**

### 3.3. Hardware Example: A Force Feedback Knob

To render the haptic behaviors designed with our haptic prototyper, we used the custom haptic knob shown in Figure 3. The knob operates with an update rate of 10 kHz, 0.001° positional accuracy, and 180 mNm maximum continuous torque when connected to a real-time Linux PC via an I/O board. These functional specifications enable a

wide variety of haptic behaviors, and dynamic simulations, to be rendered effectively.

## 4. Mathematical Representations

The underlying representations used to store the haptic behaviors will constrain and shape any haptic prototyping tool. For our tool, we use Equation 1 as our underlying representation for position-based icons and Equation 2 for our time-based icons. These equations represent the reactions of the actuator to the user's hand position, velocity, and acceleration over space (Equation 1) or time (Equation 2). Matrices of such equations and/or higher order effects could be used to handle multiple-DOF and higher fidelity actuators. Nevertheless, the approach of our one DOF prototyper is generalizable to higher DOF and/or higher fidelity haptic behavior design.

$$\tau = \sum_{i=1}^{N_i} f_p(\theta)\,\theta + \sum_{j=1}^{N_j} f_v(\theta)\,\dot{\theta} + \sum_{l=1}^{N_l} f_a(\theta)\,\ddot{\theta} \qquad (1)$$

| | |
|---|---|
| $\tau$ | Torque applied to the haptic actuator |
| $\theta, \dot{\theta}, \ddot{\theta}$ | User's hand position, velocity, and acceleration applied to the actuator |
| $f_p$ | Position dependent functions (i.e., springiness) |
| $f_v$ | Velocity dependent functions (i.e., damping) |
| $f_a$ | Acceleration dependent functions (i.e., mass) |
| $N_i$ | Number of position dependent functions |
| $N_j$ | Number of velocity dependent functions |
| $N_k$ | Number of acceleration dependent functions |

$$\tau = \sum_{i=1}^{N_i} f_p(t)\,\theta + \sum_{j=1}^{N_j} f_v(t)\,\dot{\theta} + \sum_{l=1}^{N_l} f_a(t)\,\ddot{\theta} \qquad (2)$$

| | |
|---|---|
| $\tau$ | Torque applied to the haptic actuator |
| $t$ | Time |

## 5. Example Design of a Fan Knob

The following two sections illustrate design of primitive icons using waveform editing, and how primitive icons can be organized and modified into more sophisticated haptic icons. These topics are illustrated by walking through the creation of the haptic behavior for a haptic knob using our haptic icon prototyper.

### 5.1. Waveform Editing

To build up the spring, damping, and inertia settings, we build the following torque waveforms based on the user's hand position, velocity, and acceleration, respectively. For our fan example (see Figure 2), the user will rotate the knob with their hand along a 1-D rotary path between the *Off*, *Lo*, *Med*, and *Hi* settings. The underlying representation of position-based Equation 1 is therefore more appropriate than time-based Equation 2 for building up the desired haptic behaviors.

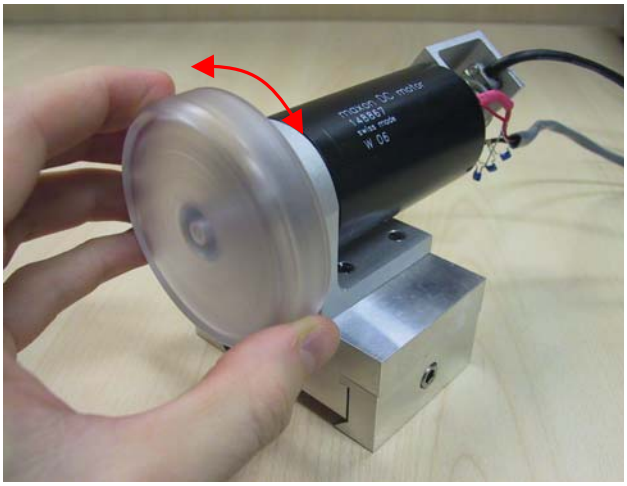

**Figure 3: Example haptic knob used with our icon prototyper**

#### *5.1.1. Position Icons*

Let's assume we first want to create a force ramp clockwise (CW) from the *Hi* position. Figure 5 shows an interaction sequence for creating a torque ramp. The user starts with a new waveform editor pane (step 1), and drags a few points up (steps 2-3) to define a ramp. The click and drag motion of a point is quite fluid because the underlying representation is a cubic spline interpolation. Typing exact numbers in a text box can specify precise angle and torque parameters associated with the current position on the waveform editor. Now the user selects the finished ramp waveform with a click & drag mouse motion (step 4). Then the user drags the selection into the icon palette (refer to Figure 1) and releases it to create a new ramp icon.

As illustrated in Figure 6, we then create the detents (i.e., 'clicks') for the *Off*, *Low Med*, and *Hi* settings. Instead of creating the points from scratch, we first open a collection of icon templates in the icon palette, then drag an icon representing a long series of detents from the icon palette to the waveform editor (step 1). We then select a section containing 4 clicks, and create a new icon by selecting and dragging a portion of the waveform back into the icon palette (step 2). The sine wave selection will create the feeling of 4 clicks because the waveform selection crosses the position axis 4 times on the torque vs. position plot. After creating a new icon, we could create a new container for this icon and subsequent icons. Or, we could add the icon to an existing icon hierarchy that had meaning to us. The tabs in our icon palette enable users to

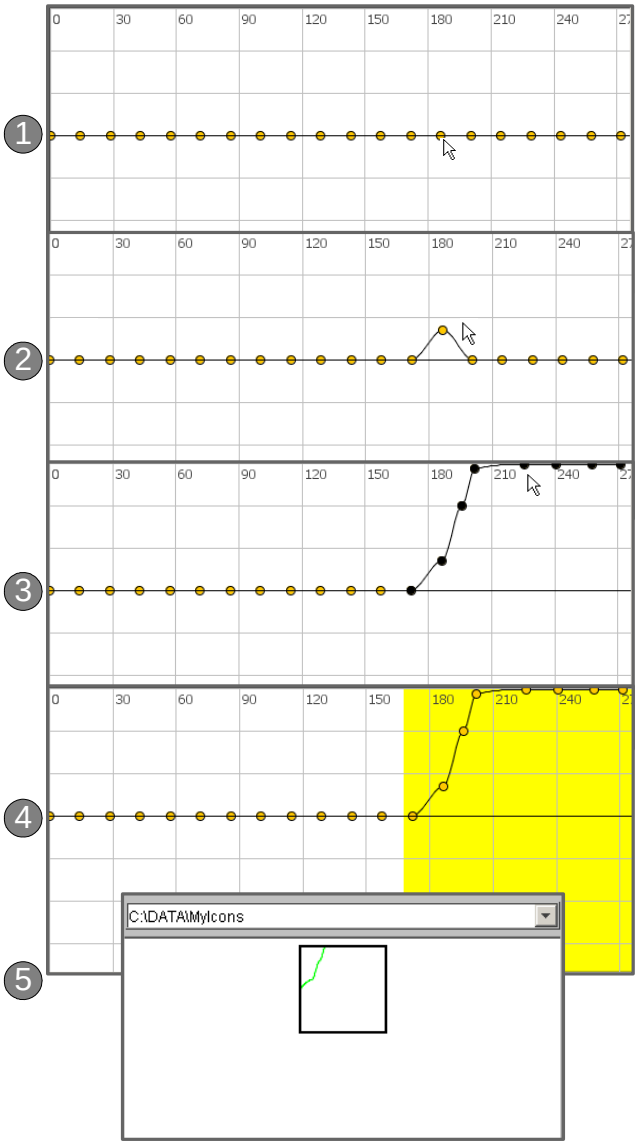


**Figure 4: Screen captures showing an interaction sequence to create a torque ramp haptic icon**

create and organize collections of icons according to their own mental models. Furthermore, the icons are stored in a text file format that is easily read and written by other software packages such as Matlab [4]. Thus, more complex functionality that is not currently integrated into the icon prototyper, such as bandpass filtering, on a particular icon can be rapidly performed in another package and then reintegrated into the haptic icon prototyper.

### *5.1.2. Velocity Icons.*

A standard damping effect across the knob movement will be defined. Also, increased damping clockwise (CW) from the *Hi* fan position, and counterclockwise (CCW)

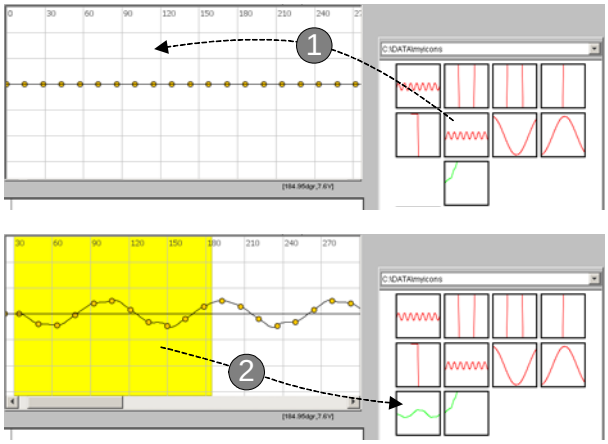


**Figure 7: Screen captures showing an interaction sequence to create one version of the basic detent module that will be used for the *Off*, *Lo*, *Med*, and *Hi***

from the *Off* fan position is illustrated. Thus, we use an icon of a constant line for $f_v(\theta)$ in Equation 1 (see Figure 6). More complex velocity-dependent effects such as Stribeck friction properties (see Figure 7) could be achieved with a curvilinear $f_v(\theta)$ and appropriate event mechanisms. The relationship between higher-order icons (i.e., velocity & acceleration) and the knob position is described further in the section *5.2 Organizing Haptic Behavior Primitives* .

### *5.1.3. Acceleration Icons.*

One acceleration icon similar to Figure 6 can be used to create the effect of a constant mass for the knob. In other words, a constant line is created for $f_a(\theta)$ in Equation 1.

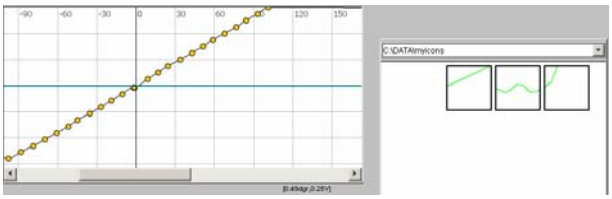

**Figure 6: Example waveform for a constant damping effect**

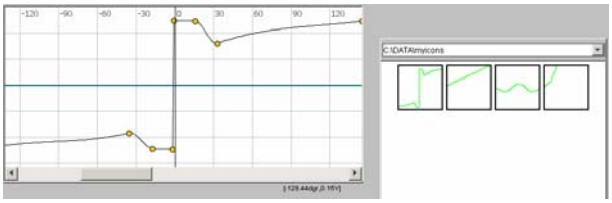

**Figure 7: Example waveform for the moving component of a Stribeck friction effect**

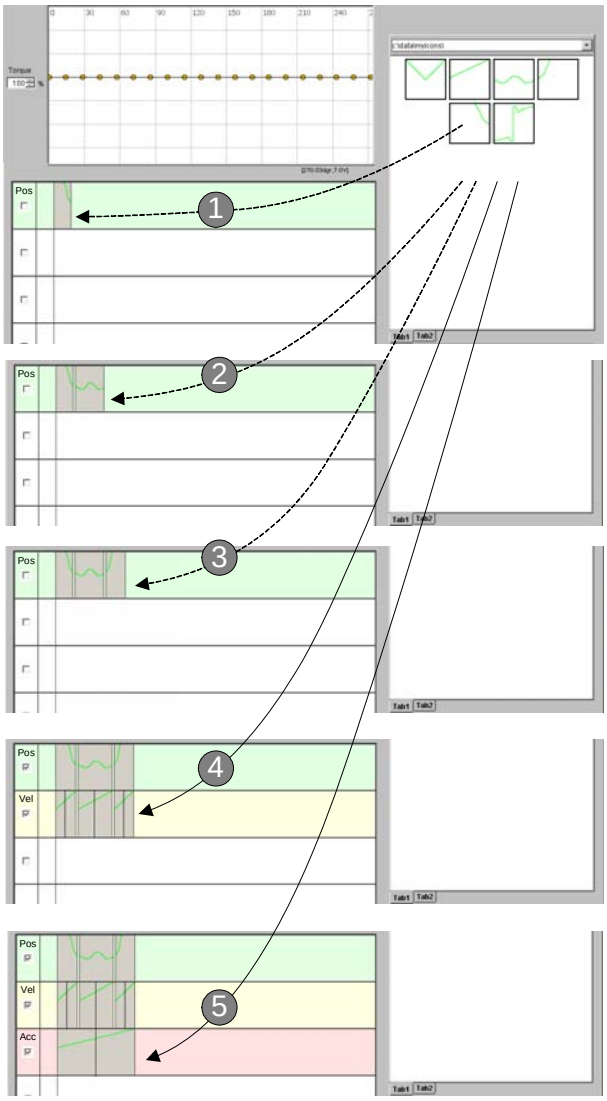


**Figure 8: Screen captures showing an interaction sequence with the tile pane to organize position, velocity, and acceleration haptic icons to build the static & dynamic haptic behaviors for an example fan knob**

### 5.2. Organizing Haptic Behavior Primitives

Organizing and editing collections of haptic icons into more sophisticated haptic behaviors is important because prototyping is typically an iterative learning process. The organization of position, velocity, and acceleration icons with our fan knob example is continued in this subsection. We introduce haptic tiles as a way to organize primitive icons and facilitate iterative design towards more complex icons.

Figure 8 illustrates use of the haptic tile pane. To aid portability and re-use, the haptic icons in the icon palette do not have associated position, velocity, or acceleration properties. Such a property is inherited from the tile pane region on which the tile is placed. In our implementation, we have chosen to color and label different layers for position, velocity, and acceleration regions of the haptic pane. Generally, haptic tiles can be dragged and dropped into the tile pane and then easily re-arranged or removed.

Steps 1-5 of Figure 8 illustrate the icon placement for:

1 A wall behavior for CCW rotation beyond the *Off* position.

2 Detents for the *Off*, *Lo*, *Med*, and *Hi* knob positions.

3 A wall behavior for CW rotation beyonds the *Hi* position.

4 Subtle knob damping behavior over the range of *Off*, *Lo*, *Med*, and *Hi* positions; and, stronger damping beyond the *Off* and *Hi* positions.

5 A constant simulated mass feeling for the knob.

During design, physically feeling and changing the incremental effects of particular haptic icons, and particular collections of haptic icons, is often more important than exploration of the final haptic behavior. To support such exploration, a haptic rendering of each line of icons in the tile pane can be felt by simply selecting the icons and pressing the *play* button (see Figure 1). Consequently, the feeling of the walls, detents, damping, and inertia behaviors can all be felt individually – or in any combination. This functionality greatly helps the haptic designer build up appropriate mental mappings between the haptic icon waveforms and their physically rendered behaviors.

## 6. Supporting Iterative Design

When prototyping, users must be able to rapidly realize their haptic behavior intentions. Reducing a 2 second operation to take 1 second can be the difference between a usable and an unusable haptic prototyper. The following lists some of our design decisions to improve user interaction fluidity in the user interface:

- ***Single interaction window.*** Most interaction is performed in the waveform editor, icon palette, and tile pane regions. Instead of having one or more of these regions available via a separate window, they are grouped close to one another such that click & drag motions between all 3 regions can be performed easily and rapidly. Additional space is obtained with use of tabs (e.g., several tabs can be selected in the icon palette).
- ***Focus on notification over correctness.*** A message bar provides the user with most status and error messages. Dialogs and pop-up messages are usually avoided.
- ***Speed & fluidity over precision.*** The general form of haptic behaviors are quickly obtained using the waveform editor and tile regions. The idea is that the user would use our haptic prototyper to explore the kinds of waveforms and haptic behaviors that are most appropriate for the current task. Precise fine tuning may be better done afterwards in another tool such as Matlab

and/or C++. Nevertheless, the icon prototyper is still an important piece of the haptic design process. It is analogous to paper prototyping or storyboarding a graphic user interface design before starting to code. Paper prototyping – like our haptic icon prototyper – is more flexible to brainstorm different options; but eventually, a less flexible, more time consuming is used to develop the final product because such tools typically offer higher fidelity or better contextualization support.

Some specific examples of interaction sequences designed to speed up haptic icon exploration are:

- ***Changing amplitude.*** Figure 9 shows the selection of a waveform segment (step 1); and, the resulting amplitude increase after dragging the selection (step 2). Various distortions are possible with such amplitude adjustments. For example, should the waveform points be evenly redistributed between the maximum value and the axis? Should points be simply translated? Or, should a more complex redistribution function (not supported in our current interface) be used? Additionally, a 'mirroring' line placed along the position axis such that points above the line are stretched upwards and points below the line are stretched downwards. Our interface supports horizontal and vertical mirrors.

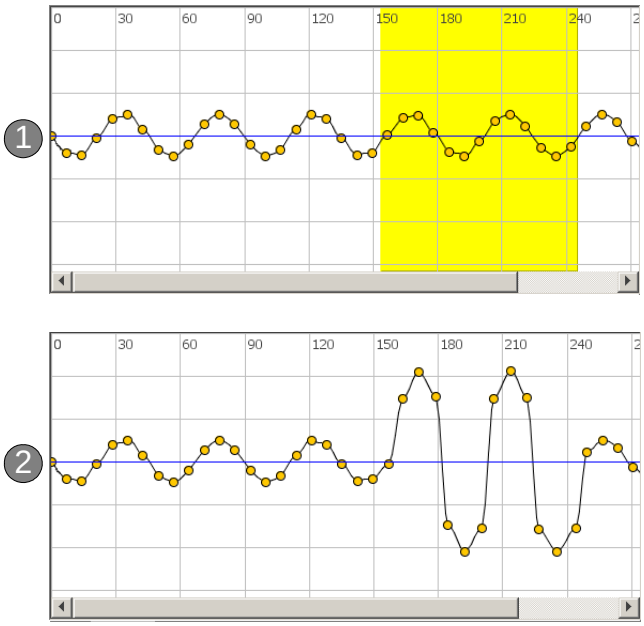


**Figure 9: Example amplitude adjustment using a 'mirror' line along the position axis in the waveform editor**

- ***Changing frequency.*** Figure 10 shows a way to decrease the frequency by directly interacting with a tile. A user selects the right boundary of the tile (step 1), and then drags the boundary leftward to yield a shorter icon of higher frequency (step 2). Similar to the amplitude modification as described above, there are different desirable actions for clicking and dragging tiles. For example, should the waveform in the tile be squished linearly, or according to some function? Should the waveform be cut?

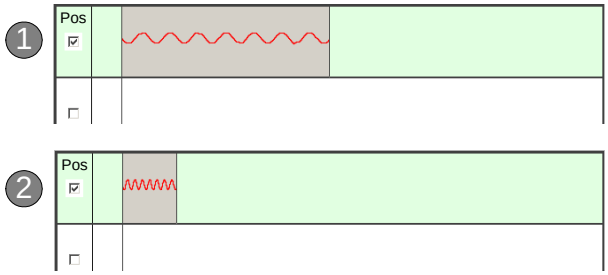


**Figure 10: Example frequency adjustment by dragging the right boundary of a haptic icon leftward in the tile pane**

- ***Combining icons.*** Once a collection of two or more haptic icons have been designed to collectively represent a common haptic behavior, combining these icons into a single entity is often desirable. For example, suppose we wish to add a subtle, high frequency clicking feel to our knob motion, we could add a high frequency sine wave to the haptic behavior illustrated in Figure 6. Such an added haptic behavior could be constructed by simply adding another line of haptic tiles as shown by step 1 in Figure 11. We could additionally superimpose the icon for this new subtle clicking behavior to the icon(s) for the existing behavior for the *Off*, *Lo*, *Med*, *Hi* feeling. Step 2 in Figure 11 illustrates such a superimposition procedure performed in our haptic tile pane. However, for iterative exploration, the user may wish to keep the two active lines illustrated in step 1 such that incremental modifications of either layer will be superimposed into the currently rendered haptic behavior 'on the fly'.

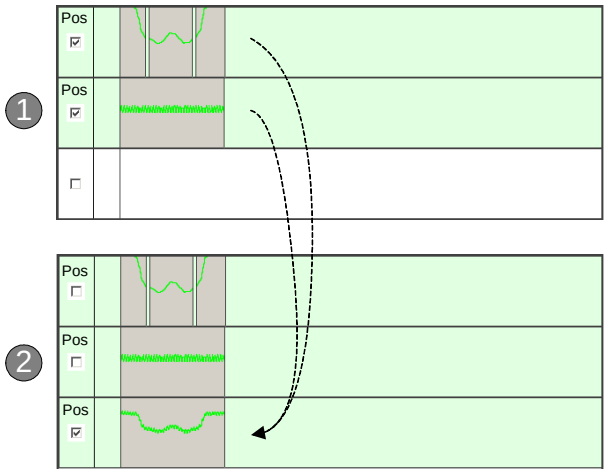


**Figure 11: Example superimposition of two icons to create a single haptic icon**

- ***Changing previous designs.*** The criteria to design an optimal haptic behavior for a given task is not known *a priori*. Thus, the ability to cycle between previous states is needed in a rapid prototyper. Figure 11 shows one example addition – a subtle clicking feel. Example features in our haptic icon prototyper that support rapid exploration include the ability to select layers to play in the tile pane, drag and drop support of icons between interface regions, additional tabs for 'scratch' collections of icons in the icon palette, and undo functionality.

## 7. General Design Principles

Our primary objective with our icon prototyper was to explore user interface widgets that enable an experienced designer to rapidly iterate between various static & dynamic behaviors to be rendered on a 1 DOF haptic actuator. Essentially, we are interested in tool design choices that facilitate rapid *exploration* of a variety of haptic parameters. A designer rarely has a clear idea of the perfect, finished haptic behavior before starting the design. S/he needs to be able to create several possible behaviors and be able to rapidly compare these behaviors. Furthermore, once one, or a small number, of promising behaviors have been developed, subtle refinements need to be made. During such a refinement process, the designer performs frequent comparisons of many haptic behaviors, and often returns to previously designed behaviors after rejecting a particular design change.

In our prototyping tool example, we focus on renderings of kinaesthetic force-based waveforms, but our interface design supports mappings to other waveforms such as temperature or moisture. We also focus on the design of a representation for a one DOF actuator. State changes and sequencing of haptic icons are currently performed by designing separate haptic icons for each state and DOF, then referencing these icons from programmed code.

Now that we've described an example haptic icon prototyper and application, we list some general design principles for an optimal haptic behavior design tool. Although, our current haptic icon prototyper is not able to completely meet this list of demanding criteria, the concepts employed in our example icon prototyper are a step towards such a goal.

### 7.1. Necessary Attributes of Tools to Support Haptic Behavior Design

#### *7.1.1. Scope and General Capabilities*

The ideal haptic icon prototyper would be able to:

- Completely represent the psychophysical capabilities of the user via a standard set of mathematical relations.
- Link up to haptic hardware in a way that provides consistent, high-quality renderings of the haptic behavior representations.
- Have easy-to-understand mental mappings between the underlying mathematical representations, the interaction widgets, and the final haptic renderings.
- Provide usable interaction widgets for designers to effectively create and modify haptic renderings.
- Integrate seamlessly with other haptic development tools, and development tools for other sensory modalities (i.e., vision, hearing, smell, and taste).

#### *7.1.2. Usability*

Resulting haptic icon renderings should effectively enable the user to complete his/her desired task. Thus, the designed haptic icon must:

- Function technically (i.e., be physiologically perceivable, and relate to the user's preconceived mental models of the task.)
- Function socially (i.e., fit into the task's social & cultural milieu.)

#### *7.1.3. Representations*

The basic haptic functional requirements of a complete icon prototyper would be to represent and convey haptic:

- ***Type.*** Kinaesthetic, tactile, temperature, moisture, or pressure sensations – including the body site. For example, control of a pressure device on a person's back.
- ***Interactions.*** Detailed 1 DOF attributes, and higher-level interactions between several DOF. For example, how does the force profile of a haptic knob change in response to the position of the knob, the time it is moved, and the current state of the system? How are a series of vibrotactile stimuli excited to on person's arm to create the feeling of a continuous motion (i.e., sensory saltation)?
- ***Psychophysics.*** Biological properties of the user's body site(s), and how these properties relate to technical attributes. For example, is a particular change in pressure level perceivable at rest? Under duress? How will a typical elderly person perceive the same surface texture compared to a teenager?
- ***Meaning.*** What the icon means to the user in terms of the target task. Will the smooth, heavy feel of a radio tuning knob induce a sense of quality and admiration for the radio? What vibrotactile behavior of a cell phone best conveys the term "call completed" or "friend calling"? What pressure should be applied to a car driver via a haptic seat to suggest s/he turn left to avoid an obstacle?

### 7.2. Accommodating Different Types of Users

We categorize and label three main types of users of haptic icon prototypers:

- ***"Programmers".*** Require the richest amount of flexibility and control. Are willing to learn and use more complicated user interfaces and program code to create sophisticated and/or novel haptic effects. Example tools include Matlab Simulink [4] and the C++ programming language.
- ***"Designers".*** Are technically savvy, but may not have the programming skills – or desire to use them – compared to programmers. The desired technical components of haptic effects are usually less specialized and novel compared to programmers. They require tools to rapidly iterate and carefully tune haptic effects. They will usually pay more attention to higher-level concepts than programmers. Our example haptic icon prototyper fits into this category.
- ***"End users".*** Operators of systems containing haptic components such as cell phones, automobiles, or test controls. They wish to focus on relatively pre-defined, stock customizations that are specialized to their particular device. For example, a cell phone GUI to select a vibrotactile ring tone fits into this category.

## 8. Conclusions and Future Work

We presented an example haptic prototyper tool, containing several novel user interaction techniques, as a contribution towards the larger goal of developing the aforementioned optimal prototyping resource. Specifically, our haptic icon prototyper is primarily targeted towards rapid, iterative development of single degree-of-freedom haptic icon designs. Using the previous prototyping tool examples, we then summarized high-level design attributes of tools to support haptic behavior design. An optimal set of haptic icon prototyping tools should support design based on all haptic types (e.g., kinaesthetic & tactile), many levels-of-detail, interactions between all degrees-of-freedom, psychophysical capabilities of the user(s), and mental models related to the user(s) task(s).

Future planned enhancements to our haptic icon prototyper include integration of visualizations representing the psychophysical limitations and physical device rendering limitations related to the haptic behavior being designed. Additional research into psychophysical limitations and mechatronic modeling would aid such visualizations. Functionality to incorporate state changes and multi-degree-of-freedom interactions would also be beneficial. Finally, tighter integration with a more generic tool that supports advanced filtering capabilities, such as Matlab, would be helpful (e.g., have our haptic icon prototyper and a Matlab session share common data structures). Thus, users could start a haptic design session with the haptic icon prototyper, and then make more time-consuming, detailed refinements to promising haptic behaviors using Matlab console commands and then reload the icon into the prototyper.

## Acknowledgements

We thank Mario Enriquez, Melanie Tory, Jerome Pasquero, Tina Li, and James Riecken for their constructive design suggestions. We also thank George Pava for his technical work of the back-end haptic system. We also thank Jukka Linjama from Nokia, and Angela Chang & Conor O'Sullivan from Motorola for discussing their haptic design strategies with us.

## References

[1] _____. Adobe Premiere http://www.adobe.com
[2] _____. Apple Soundtrack Pro http://www.apple.com
[3] _____. Microsoft Visio http://www.microsoft.com.
[4] _____. The MathWorks Matlab and Simulink http://www.mathworks.com
[5] Buxton, B., Gaver, B., & Bly, S. The Use of Non-Speech Audio at the Interface. *Tutorial Notes: ACM Conference on Human Factors in Computing Systems*, 1990.
[6] Chang, A. & O'Sullivan, C. Audio-Haptic Feedback in Mobile Phones. *Extended Abstracts of ACM Conference on Human Factors in Computing Systems*, 2005.
[7] Enriquez, M.J. & MacLean, K.E. The Hapticon Editor: A Tool in Support of Haptic Communication Research. *IEEE Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (HAPTICS)*, 2003.
[8] Gaver, W. W. Auditory icons: Using sound in computer interfaces. *Human-Computer Interaction*, 2, 1986.
[9] Immersion Corporation. *TouchSense Programmable Rotary Modules*, 2004. Available at: http://www.immersion.com/industrial/docs/TSRotary_Oct04_v3_LR.pdf
[10] Lee, G. & Hannaford, B. Anisotropies of Touch in Haptic Icon Exploration. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2003.
[11] MacLean, K. & Enriquez, M. Perceptual Design of Haptic Icons. *Eurohaptics*, 2003.
[12] Snibbe, S.S., MacLean, K.E., Shaw, R., Roderick, J.B., Verplank, W., & Scheeff, M. Haptic Metaphors for Digital Media. *ACM Symposium. on User Interface Software & Technology (UIST)*, 2001.
[13] Weir, D.W., Peshkin, M., Colgate, J.E., Buttolo, P., Rankin, J., & Johnson, M. The Haptic Profile: Capturing the Feel of Switches. *IEEE Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (HAPTICS)*, 2004.