악보를 사용한 진동 패턴 저작 도구

2010

이재봉

석 사 학 위 논 문

악보를 사용한 진동 패턴 저작 도구

이 재 봉(李 在 俸)

전자컴퓨터공학부(컴퓨터공학 전공)

포항공과대학교 대학원

2010

악보를 사용한 진동 패턴 저작 도구

Vibrotactile Score for Designing Vibrotactile Patterns

Vibrotactile Score for Designing Vibrotactile Patterns

by

Jaebong Lee

Division of Electrical and Computer Engineering
(Computer Science and Engineering)
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

A thesis submitted to the faculty of Pohang University of Science and Technology in partial fulfillment of the requirements for the degree of Master of Science in the Division of Electrical and Computer Enginneering (Computer Science and Engineering)

Pohang, Korea

December 8, 2009

Approved by

Seungmoon Choi, Major Advisor

악보를 사용한 진동 패턴 저작 도구

이 재 봉

위 논문은 포항공대 대학원 석사 학위 논문으로 학위 논문 심사 위원회를 통과하였음을 인정합니다.

2009 년 12 월 8 일

학위논문 심사위원회 위원장 최 승 문 (인)

위 원 이 승 용 (인)

위 원 박 성 우 (인)

MECE 20080313 이 재 봉, Jaebong Lee, Vibrotactile Score for Designing Vibrotactile Patterns. 악보를 사용한 진동 패턴 저작 도구, Division of Electrical and Computer Engineering (Computer Science and Engineering), 2010, 53 P, Advisor: Seungmoon Choi. Text in English

Abstract

Despite the plethora of available vibrotactile applications that have already begun to impact our everyday life, how to design vibrotactile patterns easily and efficiently continues to be a challenge. As an intuitive and effective approach for vibrotactile pattern design, this paper proposes a *vibrotactile score*. The term comes from a metaphor of a musical score and its design is adapted from two common musical scores (piano score and guitar tablature). Another metaphorical feature analogous to the musical clef, named a *vibrotactile clef*, is also introduced to decouple the processes of low-level vibrotactile signal design and high-level pattern composition. The conceptual design of the vibrotactile score and clef are fully realized in a graphical authoring tool named the Vibrotactile Score Editor (VibScoreEditor). We demonstrate the expressiveness of the vibrotactile score with several examples. In addition, the usability of the vibrotactile score was evaluated, focusing on its learnability, efficiency, and user preference. Experiment 1 was to compare the vibrotactile score and the current dominant practices of vibrotactile pattern implementation including programming and scripting. The results gained from programming experts validated the substantially superior performance of the vibrotactile score. Experiment 2 was to compare the vibrotactile score with the waveform-based design already implemented in a few recent graphical authoring tools for vibrotactile patterns. Ordinary users without programming backgrounds participated in this experiment, and the results substantiated the excellent performance of the vibrotactile score.

Contents

List of Figures

List of Tables

Chapter 1

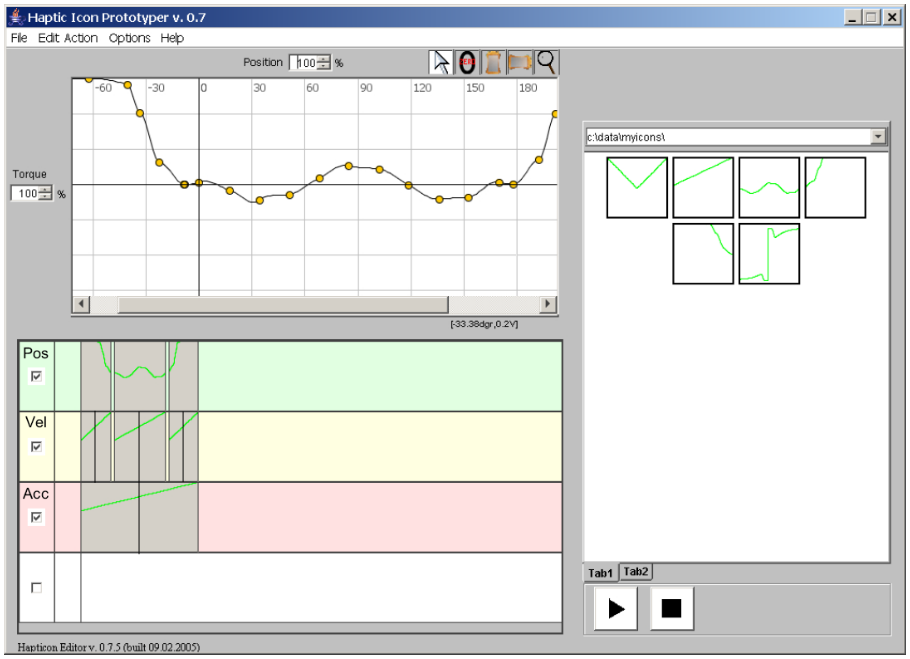
Introduction

1.1 Motivation and Related Works

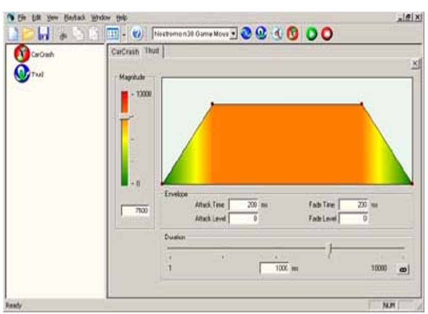
Vibrotactile feedback can enhance interactivity between the user and a device by conveying useful information to the user, especially when visual and audio information is limited. A large number of studies have investigated the usefulness of vibrotaction for various applications in the areas of HCI (Human-Computer Interaction), VR (Virtual Reality), and gaming. A detailed literature review of the subject can be found in [10, 17] centered on information communication and [28] focused on in-vehicle applications. Furthermore, vibrotactile rendering has recently been adopted for use in commodity products including full touch-screen mobile phones and active safety systems for automobiles.

In parallel with technological advances, there has been an increasing demand for software tools that can facilitate the design and evaluation of vibrotactile patterns. Notable developments include the Hapticon Editor [9], Haptic Icon Prototyper [29], VibeTonz studio [12], and posVibEditor [24]. All of these editors offer an intuitive and easy-to-use GUI (Graphical User Interface) along with some unique features.

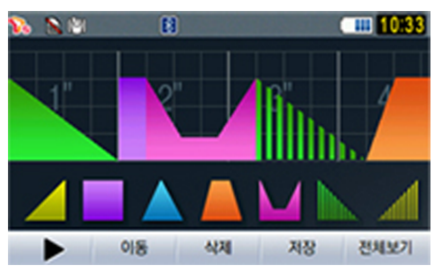
The Hapticon Editor and its upgraded version, the Haptic Icon Prototyper (Fig. 1.1a), were developed for haptic icons to be played with a one degree-of-freedom force-feedback device (e.g., a haptic knob). Even though the target attribute was a force profile, the two editors can be easily adapted to allow for vibrotactile pattern design. The VibeTonz studio



(a)Haptic icon prototyper [29].



(b)VibeTonz studio [12].



(c)"My own haptic" menu of Haptic Phone.

Fig. 1.1 Previous vibrotactile pattern authoring tools.

(Fig. 1.1b), a commercial editor from Immersion Corp., is for mobile devices, and offers a template of simple pattern elements and a timeline interface in which the elements are combined for complex vibrotactile patterns. In addition, the VibeTonz studio can automatically generate vibrotactile patterns from music files in MIDI (Musical Instrument Digital Interface) format, which is a convenient feature for application development. A simplified version was included in a Haptic Phone (Samsung Electronics; model SCH-W420), so that the users could create their own vibrotactile patterns (Fig. 1.1c). Recently, our research group released the posVibEditor that had several advanced functions. For instance, it supports pattern design and tests for multiple vibration actuators, which are common in HCI and VR research, using a multi-channel timeline interface. Another unique feature is a design mode for perceptually transparent rendering that can minimize distortion in the user's percept from the intended vibrotactile effects using a psychophysical magnitude function [26, 23].

1.2 Research Goal

Previous vibrotactile pattern authoring tools are analogous to a sound composition program that allows for the direct manipulation of sound waveforms. Although this low-level access provides the greatest flexibility in shaping a waveform, composing music or audio icons in this way is far from being intuitive or efficient. For this reason, we usually rely on a musical score. Based on the same metaphor, we propose, to compose vibrotactile effects or tactile icons, the use of a *vibrotactile score* that represents vibrotactile patterns using symbols adapted from musical scores. Previously, a similar but simpler representation was anecdotally used to describe tactons [1]. In this paper, we formalize the definitions of symbols for a vibrotactile score with an explicit link to their signal-level counterparts.

Our vibrotactile score is based on a piano score with a few features borrowed from a guitar tablature. Using the score, we can represent the desired pitch, strength, and duration of a vibrotactile note, and compose "vibrotactile music" in the same way as we do with musical scores. This process is intuitive and easy to learn, allowing for the efficient design of vibrotactile patterns even for non-experts.

We also implemented the conceptual design of the vibrotactile score in a graphical editor, named the Vibrotactile Score Editor (VibScoreEditor), which has the following features:

- An intuitive GUI: Vibrotactile patterns can be graphically designed and edited simply by adding and deleting the notes and rests on a score.

- Decoupled management of vibrotactile notes and corresponding vibrotactile signals: The VibScoreEditor provides a “vibrotactile clef” to define the physical characteristics of a vibrotactile signal for each staff line in a vibrotactile score in a way analogous to a musical clef.

- Data files in the XML (eXtensible Markup Language) format: By using XML documents to store data, the VibScoreEditor ensures reusability and extensibility.

- On-the-fly tests of designed patterns: A vibrotactile pattern player is embedded in the editor to facilitate iterative designs and testing.

- Support of different vibration actuators: A driver module for a vibration actuator can be independently developed and linked to the VibScoreEditor.

In addition, we experimentally evaluated the usability of the vibrotactile score in two experiments. The experiments differed in target user populations and the design methods used for comparison. In Experiment 1, conventional design methods, programming in C language and scripting in XML, were compared with editing with the VibScoreEditor by expert users. In Experiment 2, two graphical vibrotactile pattern authoring methods, waveform editing and vibrotactile score editing, were compared by ordinary users. The results of both experiments demonstrated the greatly improved performance of vibrotactile score in terms of learnability, efficiency, and user preference.

Chapter 2

Vibrotactile Score

Our vibrotactile score has been designed by combining the features of two musical scores (piano score and guitar tablature). This section presents its conceptual design.

2.1 Musical Scores

It goes without saying that the piano score is the most widely used musical score. In this score, the pitch of each note is represented by its position with respect to the five staff lines with a higher position indicating a higher tone, and its duration is denoted by its shape (Fig. 2.1). The symbols for rests represent their durations only. This is quite an effective notation for the piano, since the position of a note is mapped on a one-to-one basis to the position of a key on the keyboard.

However, using a piano score to play a guitar is not straightforward. The position of a note is not intuitively mapped to a finger position on the fingerboard of a guitar, and can even be mapped to multiple finger positions. A more popular way is to use the guitar tablature that has six horizontal lines mapped on a one-to-one basis to the six strings of a guitar (Fig. 2.2). In the guitar tablature, the head of each note includes a number that specifies the fret number on which a finger should be placed on the fingerboard. Another difference is that in the guitar tablature, the half note (the fourth symbol in Fig. 2.2) is distinguished from the quarter note (the second symbol) by the lengths of their stems (a vertical line connecting

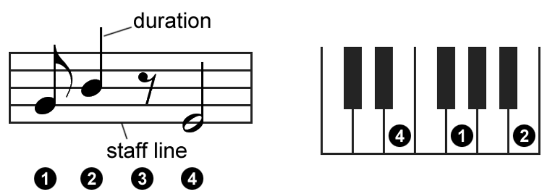


Fig. 2.1 Each note in a piano score (left) corresponds to a key on the keyboard (right). See the circled numbers.

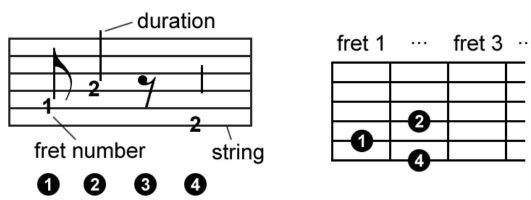


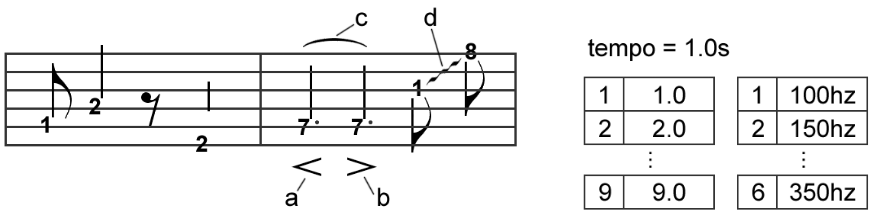
Fig. 2.2 Guitar tablature (left) and corresponding fingering positions on the guitar fingerboard (right). See the circled numbers.

the head and flag of a note). The shapes of other note and rest symbols are the same as those of the piano score. These features in the guitar tablature (use of strings, fret numbers, and duration) allow for easy and intuitive score reading for guitar players.

2.2 Vibrotactile Score

When designing the vibrotactile score, we had two major considerations to take into account. The first was that to minimize the learning and design times, the vibrotactile score needs to be as similar as possible to one (or two) widely recognized musical scores. The second was that the representations of vibrotactile patterns must be comprehensive enough to cover vibrotactile signals that can be generated by currently available vibrotactile actuators.

For the first design goal, we speculated on which requirements are unique to a score for vibrotaction. It immediately became clear that whereas a musical score is played by a human, a vibrotactile score is played by a computer. The absolute strength of a note is not specified in a musical score since it is determined by a human performer. On the



(a)Vibrotactile score.

(b)Vibrotactile clef.

$$y(t) = \begin{cases} 1.0 \cdot \sin(2\pi \cdot 150 \cdot t), & 0.0 \le t < 0.5 \\ 2.0 \cdot \sin(2\pi \cdot 200 \cdot t), & 0.5 \le t < 1.5 \\ 0.0, & 1.5 \le t < 2.0 \\ 2.0 \cdot \sin(2\pi \cdot 100 \cdot t), & 2.0 \le t < 4.0 \end{cases}$$

(c)Corresponding vibration equations of the first measure in (a).

Fig. 2.3 Conceptual design of the vibrotactile score.

contrary, a vibrotactile score must have a way to represent the strength of a vibrotactile note to be produced by a vibration actuator. Even though computer-playable music protocols exist that contain intensity information such as MIDI, they are excessively complicated for vibrotactile patterns and recognizable only by music experts. Our strategy, therefore, has been to combine the features of two common musical scores, the piano score and guitar tablature, while preserving their familiarity to ordinary users.

In our design (Fig. 2.3a), the pitch of a note is represented by its vertical location on the staff lines (as in the piano score), and its duration is determined by its shape (as in the piano score and guitar tablature). We denote the strength of the note by an integer inside its head. It is important to point out that this notation is the same as the fret number in the guitar tablature, but its meaning is different. We use six staff lines as in the guitar tablature simply to express one more pitch. More staff lines can be added if more pitches are desired, but this may weaken the score metaphor. Overall, the appearance of the vibrotactile score is more similar to the guitar tablature, but its functionality is closer to the piano score.

Four more symbols, among the many used in musical scores, are employed in the vibrotactile score (see the second measure in Fig. 2.3a). The first two are dynamics symbols:

the crescendo and decrescendo (a and b in the figure), which express gradual changes of vibration strength. As in music, the former indicates that the intensity of a note is to be increased over time, and the latter suggests that it should be decreased. These two patterns are very effective and frequently used in tacton design [4]. The other two symbols, legato and portamento (c and d in Fig. 2.3a), describe how to play adjacent notes. When playing a vibrotactile score, a short silence interval is inserted between two adjacent notes by default to make them distinguished. When the notes are tied with a legato, they are played continuously without a silence interval. If a portamento connects two neighboring notes, the pitch and strength of their vibration are interpolated continuously from those of the preceding note to those of the following note without a silence interval. Diverse and expressive transition effects can be designed using the four symbols.

For the second design criterion, we introduced another metaphor named the *vibrotactile clef*. The "vibrotactile clef", which is analogous to the musical clef[1], defines a mapping from the position and intensity level of a vibrotactile note to the respective physical parameters associated with the pitch and strength of a vibrotactile signal. For example, assume that we use simple sinusoidal vibrations expressed by $y(t) = A\sin(2\pi Ft)$ where A is amplitude, F is frequency, and t is time. Then, in its vibrotactile clef, the position and intensity level of a note are mapped to the frequency and amplitude of a sinusoid, as shown in Fig. 2.3b. A tempo variable stores the duration of the quarter note in seconds, and the durations of other notes are scaled accordingly. With this vibrotactile clef, the first four notes in the vibrotactile score shown in Fig. 2.3a will generate the vibrotactile pattern shown in Fig. 2.3c.

Note that the current design of the vibrotactile score places notes only on the staff lines (but not between them), which provides only six available pitches for pattern design. This decision was due to the limited frequency discriminability of our vibrotactile perception. The JND (Just Noticeable Difference) of signal frequency is about 20% for sinusoidal vibrations perceived at the fingertip [18]. Moreover, the number of vibrations that can be

[1]The musical clef indicates the actual pitches of staff lines. For example, the treble clef makes the five staff lines represent E-G-B-D-F from bottom to top. If other clef symbols are used, the pitches of the lines are changed accordingly.

reliably distinguished by frequency only is also fairly small; three in 100 – 300 Hz [27] or three in 80 – 250 Hz [25]. From our experience, including diverse features to express more pitches, such as placing note heads between the staff lines or using sharp and flat symbols, tends to make the design and editing process unnecessarily complicated. If more than six pitches are indeed necessary, the user can use multiple vibrotactile clefs as shown in the first design example in Chapter 5.

Our design of a vibrotactile score with a vibrotactile clef has several important advantages over other methods. First, since the vibrotactile score shares exactly the same notation as piano scores and guitar tablature, it is intuitive and easily to learn, even for non-experts. Second, the decoupled structure of the vibrotactile score and clef allows the user to focus on pattern composition using a sequence of score symbols while not paying much attention to their low-level physical parameters. Third, given a vibration actuator, the expert user can define a vibrotactile clef using a set of vibrotactile stimuli that are effective with the actuator. A well known fact is that each type of vibrotactile actuators has different actuation characteristics [21]. Lastly, given a vibration actuator and a vibrotactile score, changing a vibrotactile clef from one setting to another may produce perceptually distinct vibrotactile patterns, greatly increasing the expressiveness of pattern composition. The only requirement is that to preserve the metaphor, a vibrotactile tone represented by a higher string should have a higher pitch than one represented by a lower string.

Chapter 3

XML Schema for Vibrotactile Score

We have embodied the conceptual design of the vibrotactile score in an XML format, which has strong structurability, extensibility, and interoperability [11]. It is therefore a suitable format for representing the structured data in a vibrotactile score. It has been also used for other types of haptic data [32, 24]. We define XML schemas for the vibrotactile score and clef in two respective documents.

3.1 XML Document for Vibrotactile Score

The document structure for the vibrotactile score is shown in Fig. 3.1. This document stores information about notes and rests separated by bars in a score. The definitions of elements and associated attributes are summarized in Table 3.1. The vibrotactile clefs to be used in a vibrotactile score are declared in element $Clef$. Each Bar element designates a vibrotactile clef to use via attribute $clef$, in a similar way to a clef change in a musical score. The attribute $length$ in the $Note$ and $Rest$ elements can have one of ten discrete values: $whole$, $half\_dot$, $half$, $quarter\_dot$, $quarter$, $8th\_dot$, $8th$, $16th\_dot$, $16th$ and $32th$, where the postfix $\_dot$ indicates a dotted symbol (e.g., a Rest element with $length$ = $quarter\_dot$ represents the dotted quarter rest). A $dynamics$ attribute is used to specify a special pattern

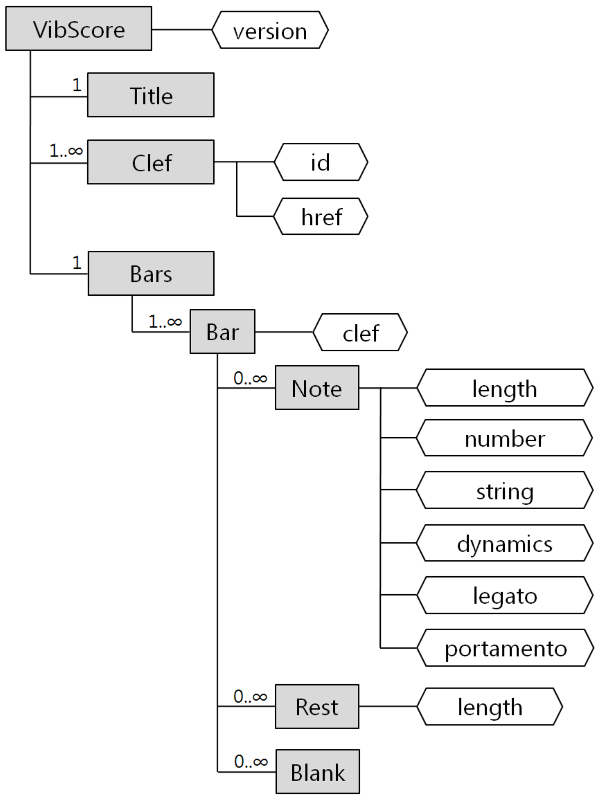


Fig. 3.1 XML schema for a vibrotactile score document.

of intensity change, and can be one of *crescendo*, *decrescendo*, or *none*. Notes tied with legato are represented by the same non-zero serial integer in the *legato* attribute. The same applies to the *portamento* attribute. The roles of the other elements should be self-evident in Table 3.1.

3.2 XML Document for Vibrotactile Clef

The XML document for the vibrotactile clef has the hierarchy shown in Fig. 3.2, with the elements and attributes defined in Table 3.2. The clef document supports three types of vibrotactile waveforms: sinusoidal, curve, and line waveforms, which are denoted in attribute *type* by *sine*, *curve*, and *line*, respectively. If $type = sine$, element *String* has

Table 3.1 Roles of elements and attributes in the score document.

VibScore	Root element of a score document	
	version	Current version number
Title	Title of a vibrotactile score	
Clef	Declaration of vibrotactile clefs that will be used in the score	
	id	Identifier of a vibrotactile clef
	href	Relative path to the XML document of a vibrotactile clef
Bar	Definition of a bar	
	clef	Identifier of the vibrotactile clef that will be used in the bar
Note	Definition of a note	
	length	Duration of the note
	number	Number inside the note
	string	String of the note
	dynamics	Dynamics of the note
	legato	Legato serial number of the note
	portamento	Portamento serial number of the note
Rest	Definition of a rest	
	length	Duration of the rest
Blank	A white space, needed for editing only	

frequency values in its definition. For example,

```
<Strings>
  <String>50</String>
             ...
  <String>300</String>
</Strings>
```

If $type = curve$, $String$ stores a set of control points that are smoothly interpolated to form a curve waveform using the Catmull-Rom spline [6]. If $type = line$, the control points in $String$ constitute a piecewise linear waveform. In both cases, a set of $Point$ elements are used in the definition, such as

```
<Strings>
  <String>
    <Point time="0.0000" level="0.3000"/>
                    ...
    <Point time="1.0000" level="0.0000"/>
  </String>
     ...
  <String>
    <Point time="0.0000" level="0.2000"/>
                    ...
    <Point time="1.0000" level="1.0000"/>
  </String>
</Strings>
```

The $time$ and $level$ attributes of each $Point$ element represent the normalized time and amplitude of a control point, respectively. The element $Number$ is a scaling factor for the normalized amplitude of waveforms stored in $String$ elements, and the element $Tempo$ does the equivalent to scale time. The pitch of a vibrotactile signal that is perceived by the user is determined by the waveforms and the associated parameters of normalized amplitude and tempo [20].

The waveform for each string is specified for a quarter note. A note that has a shorter duration than a quarter note plays the quarter note waveform only for the duration of the note. If a note has a longer duration than the quarter note, the quarter note waveform is repeated during the duration of the note. For example, an eighth note plays only the first half of a quarter note waveform, and a half note plays a quarter note waveform twice.

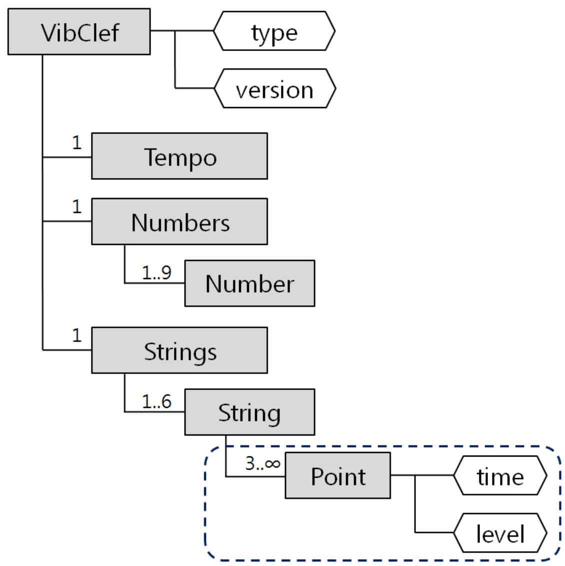


Fig. 3.2 XML schema for a vibrotactile clef document.

Table 3.2 Roles of elements and attributes in the clef document.

<table>
<tr><td rowspan="3">VibClef</td><td colspan="2">Root element of a clef document</td></tr>
<tr><td>type</td><td>Type of a clef document</td></tr>
<tr><td>version</td><td>Current version number</td></tr>
<tr><td>Tempo</td><td colspan="2">Duration of a quarter note</td></tr>
<tr><td>Number</td><td colspan="2">Definition of a number inside a note head</td></tr>
<tr><td>String</td><td colspan="2">Definition of a string</td></tr>
<tr><td rowspan="3">Point</td><td colspan="2">Definition of a control point</td></tr>
<tr><td>time</td><td>Normalized time of a control point</td></tr>
<tr><td>level</td><td>Normalized amplitude of a control point</td></tr>
</table>

Chapter 4

Vibrotactile Score Editor

All of the conceptual designs and the XML documents related to the vibrotactile score have been implemented in the VibScoreEditor on Microsoft Windows using Microsoft Visual C++ 2003. The editor consists of four modules for XML loading, data management, user interface, and vibration playback, respectively.

4.1 XML Loading and Data Management

The XML loader was implemented using the MSXML (Microsoft Core XML Service) 6.0 library. It loads and stores the XML documents for vibrotactile score and clef definitions. The loaded XML data are managed in the data management module using a C++ class hierarchy. This class instance is also referenced by other modules for user interface and vibration playback.

4.2 User Interface

To provide an easy-to-learn and easy-to-use design environment, the VibScoreEditor employs the user interface shown in Fig. 4.1, which is similar to those of guitar tablature authoring tools such as Power Tab Editor [15]. To begin a pattern design, the user first defines a set of vibrotactile clefs using the two small windows shown on the right side in the figure.

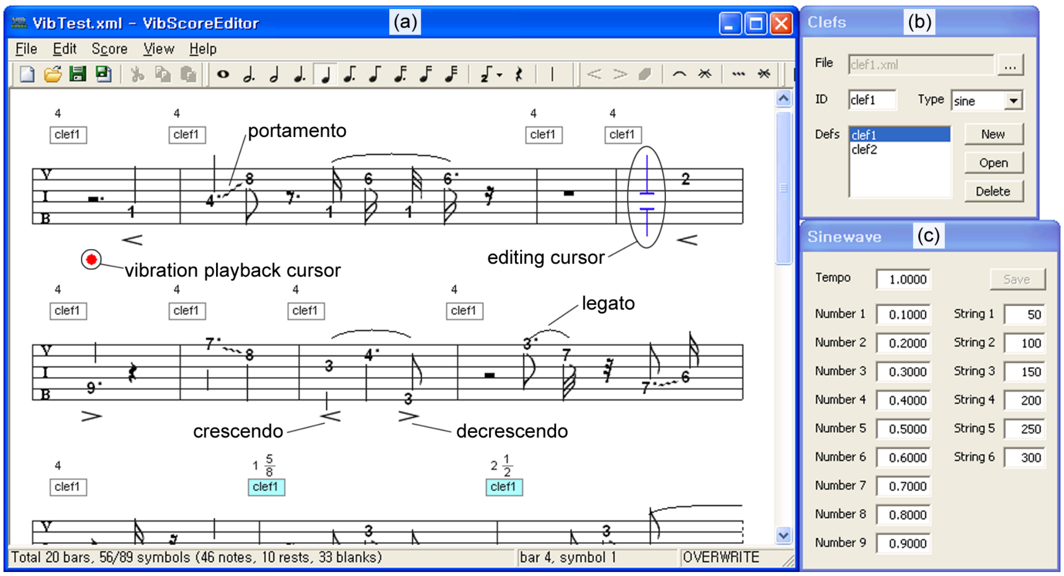
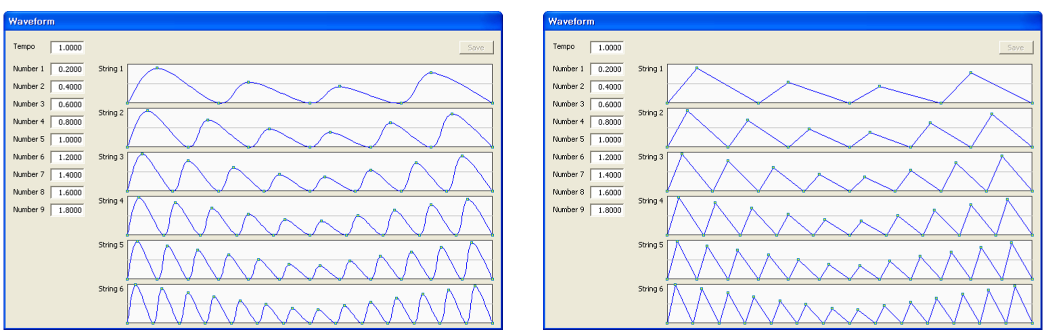


Fig. 4.1: User interface of the VibScoreEditor.



(a)Curve waveform window. (b)Line waveform window.

Fig. 4.2: Curve and line waveform windows.

Fig. 4.1(b) shows a clef management window in which the user can add a vibrotactile clef by making a new clef or loading an existing one. Fig. 4.1(c), called the waveform window, provides an interface to view and edit the actual properties of the vibrotactile clef currently selected in the clef management window. Depending on the type of the selected clef, one of three waveform windows (sinusoidal, curve, or line waveform) appears. For the sinusoidal option, a user can assign each string to a desired frequency and set each in-head number to a desired amplitude using the edit boxes. For the curve or line waveform option, a user can manipulate a waveform by adding, dragging, and deleting control points using a mouse (see Fig. 4.2).

In the main window (Fig. 4.1(a)), the blue vertical bar on the staff lines represents an editing cursor. The current active string is the one between the upper and lower vertical bars of the cursor. The user can move the cursor to a desired position by using the arrow keys or the mouse. A score symbol can then be inserted via the corresponding key, menu item, or tool bar icon, as summarized in Fig. 4.3. The duration of a note or rest can be selected from the menu or toolbar. Convenient editing functions such as block selection along with copy and paste functions are fully supported in the standard way of Microsoft Windows. Strings and the in-head numbers of notes can be changed simultaneously after selecting them in a block. This is done by pressing the up or down key in conjunction with the shift key to increase or decrease the strings of notes, and by pressing the plus or minus key to increase or decrease the in-head numbers of notes.

A small box above each bar line displays the ID of the vibrotactile clef to be used for the bar. The ID can be set by using a popup menu, or by selecting a desired clef in the clef management window and then double-clicking the clef ID box above the bar. In addition, a number above a clef box denotes the total number of beats in the bar. We set one beat to be the same as the duration of a quarter note. The VibScoreEditor computes the number of beats in each bar, displays it, and denotes whether the number is less than, equal to, or larger than a default value by coloring the corresponding clef box with cyan, white, or magenta, respectively. The default number of beats is set to four in each bar.

4.3 Vibration Playback

The VibScoreEditor includes a playback module for vibrotactile patterns. To enable this, a vibration playback cursor is shown in a red-filled circle under the bars in the main window (see Fig. 4.1(a)). The user can move the playback cursor to the position at which playback should begin, and start or stop playback using the corresponding controls in the toolbar. During playback, the playback cursor automatically progresses to a position under the note

Function	Key	Icon
move editing cursor	arrow key	
insert note	number key (**1~9**)	
insert rest	zero key (**0**)	
split bar	grave accent key (`)	|
delete symbol or block	**delete** key	
set crescendo	**c** key	<
set decrescendo	**d** key	>
remove dynamics	**e** key	
set legato	**l** key	
remove legato	**g** key	
set portamento	**p** key	
remove portamento	**m** key	
set current length	Shift + number key (**Shift + 0~9**)	~

Fig. 4.3 Editing functions and corresponding keys and tool bar icons in the VibScoreEditor.

(or rest) that is being played, and meanwhile all the editing functions are disabled.

Another important implementation issue is how to support the various hardware configurations of vibrotactile actuators and communication devices. Since no such standards exist for vibrotactile rendering, we encapsulate the vibration playback module in a DLL (Dynamic Link Library) for this purpose. Each DLL controls a specific vibrotactile rendering device. This allows the user to extend the VibScoreEditor for use with their hardware simply by implementing a DLL as long as the DLL shares the header file interface. By default, the VibScoreEditor loads 'VibModule.dll' in the same directory where the executable file of the VibScoreEditor is located, and then the user can change the vibration playback module at runtime by selecting the 'Load DLL' menu and opening another DLL file.

The current VibScoreEditor provides three DLLs for a mini-shaker (Brüel & Kjær; model 4810), a voice-coil actuator (Samsung Electro-Mechanics; also called a linear resonance actuator; used in the Samsung Haptic phone), and a vibration motor. A data acquisition board (National Instruments; model PCI-6229) is used for communication. A null DLL is also included to support vibrotactile pattern design without playback.

4.4 Design Issues of Vibrotactile Clef

The vibrotactile clef serves as a gateway to send patterns in a vibrotactile score to the physical world. Therefore, vibrotactile clefs must be designed with an adequate understanding of the dynamic performance of a vibration actuator. Otherwise, the actual vibrotactile signals produced by the actuator could significantly deviate from those intended using a vibrotactile clef (e.g., see [2]).

Actuators that have a wide bandwidth, such as the mini-shaker and a piezoelectric actuator, do not put a severe limit on the design of vibrotactile clefs, except that even such high-performance actuators may not be able to produce waveforms with very sharp transitions (e.g., a square wave). A voice-coil actuator has a frequency band of large gains centered at its resonance frequency. The principal frequencies selected in the vibrotactile clef should reflect this fact. For a vibration motor, a vibrotactile clef should use either strings or in-head numbers to represent the driving voltage levels, since the voltage applied to the motor determines both the frequency and amplitude of vibration [13]. For the successful design of vibrotactile patterns, it is important to confirm whether or not the vibrotactile clef works as intended by measuring actually produced vibrations.

Chapter 5

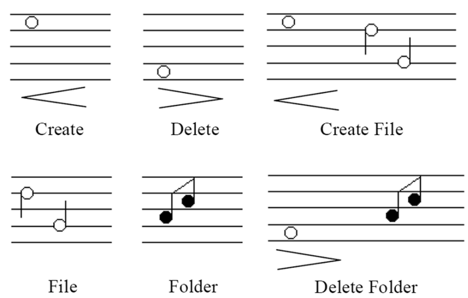
Design Examples

In this section, we demonstrate the expressiveness of the vibrotactile score by using three examples. The first example as shown in Fig. 5.1a is to make a vibrotactile score for tactons taken from [1]. In these simple tactons, a high-frequency vibration with an increasing intensity represents 'create,' a low-frequency vibration with a decreasing intensity represents 'delete,' two consecutive notes with a decreasing frequency represent 'file,' and two consecutive notes with an increasing frequency represent 'folder.' They can also make compound messages.

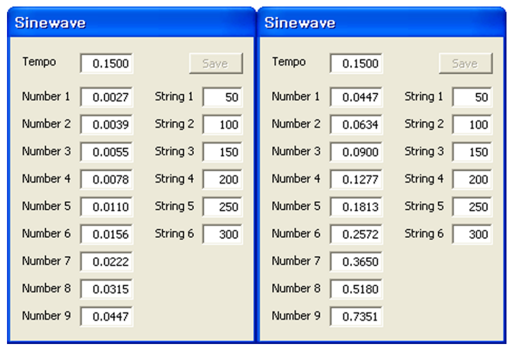
Fig. 5.1c shows a score example for a tacton 'create file' where 'create' is designed using a crescendo. The vibrotactile clef is made for a mini-shaker and is similar to one shown in Fig. 4.1(c) with a linearly increasing intensity. The data in the right panel is an actual vibration measured from the mini-shaker with an accelerometer when the tacton was played back. In contrast, a tacton for 'delete folder' in Fig. 5.1d uses two clefs as defined in Fig. 5.1b. They represent exponentially decaying sinusoids that are frequently used to model high-frequency vibrotactile transients that occur during collisions between rigid objects [22]. The two sinusoidal clefs have the same frequency set but different intensity sets, one for a low amplitude range (0.0027 – 0.0447) and the other for a high amplitude range (0.0447 – 0.7351), with an exponentially growing intensity. In this example, the part for 'delete' is implemented using many notes of the same duration, with a sufficiently small tempo (0.15 seconds for a quarter note) to produce continuously varying waveforms.

The second example uses the vibration motor included in the vast majority of cellular phones. Since the frequency and amplitude of vibration induced by a vibration motor are correlated, rhythm and roughness are used to encode information in this example, based on the approach taken by [5]. As illustrated in Fig. 5.2a, the rhythm represents a message type. Then, the roughness delivers the message priority with a rougher tacton indicating a higher priority. One effective way to modulate the roughness is to use a vibrotactile clef in Fig. 5.2b where each string represents a square wave with a period in the range of 20 ms (string 6) – 120 ms (string 1). Due to the large actuation delay of a vibration motor (typically in 100 – 300 ms), using such short duration pulses cannot fully accelerate the motor to a steady state. Instead, it results in a waveform that includes high-frequency sinusoids (determined by the level of applied voltage) with their amplitude spiking irregularly, as seen in the right panel of Fig. 5.2c. The perceived roughness increases along with the signal duration to a certain degree. For example, string 4 feels much rougher than string 6, but not string 1 (also see [5]). Using this effect, a tacton for a 'high priority voice call' can be designed as seen in the left panel of Fig. 5.2c, which then produces an acceleration profile in the right panel.

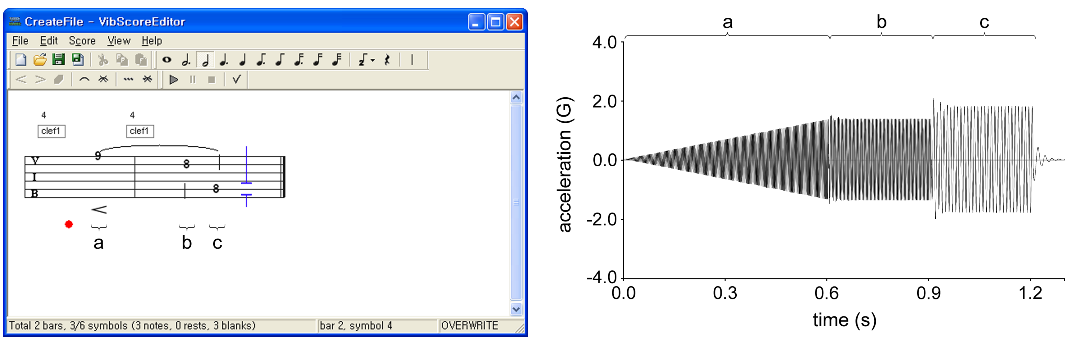
The last example demonstrates how easily a musical score can be translated to a vibrotactile score. In Fig. 5.3a, the first part of the main theme of movie "Mission: Impossible" is presented, with its vibrotactile version in Fig. 5.3b. The two scores are very similar and have identical rhythms. Slight differences in the staff positions of corresponding notes are due to the simpler structure of the vibrotactile score which places notes only on the staff lines. The note intensities were manually tuned by the designer. This vibrotactile music example was demonstrated to the public at the World Haptics Conference 2009 via the mini-shaker using a clef similar to the one in Fig. 5.1c, and received very favorable responses.



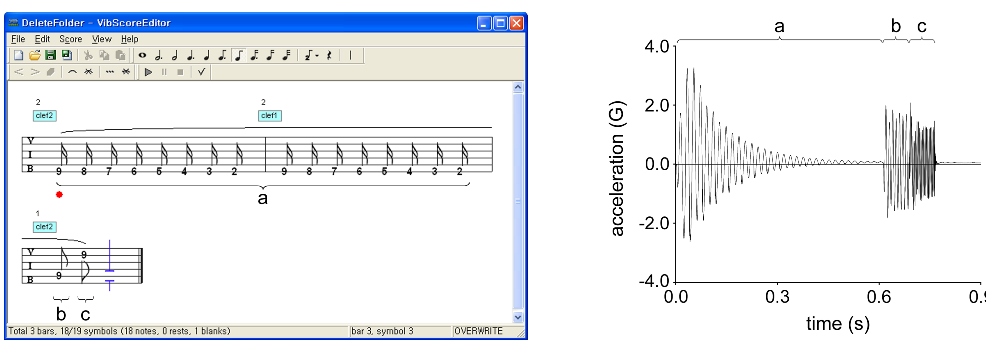
(a)Compound tactons taken from [1].



(b)Vibrotactile clefs used for a ‘delete folder’ tacton.

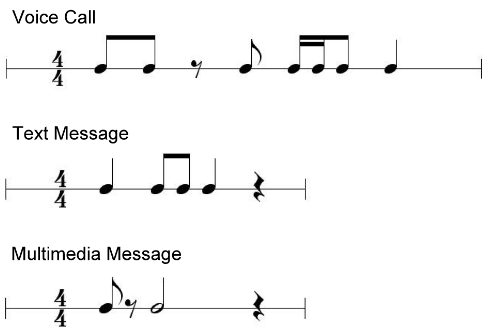


(c)Design example of the ‘create file’ tacton and a measured vibrotactile pattern from the mini-shaker.

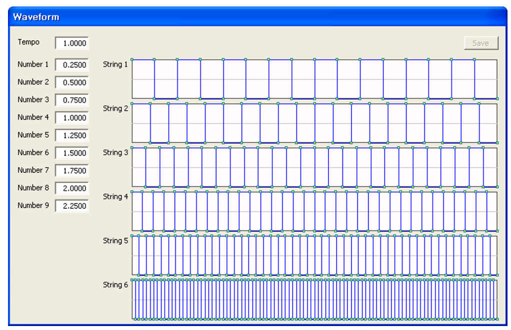


(d)Design example of the ‘delete folder’ tacton and a measured vibrotactile pattern from the mini-shaker.

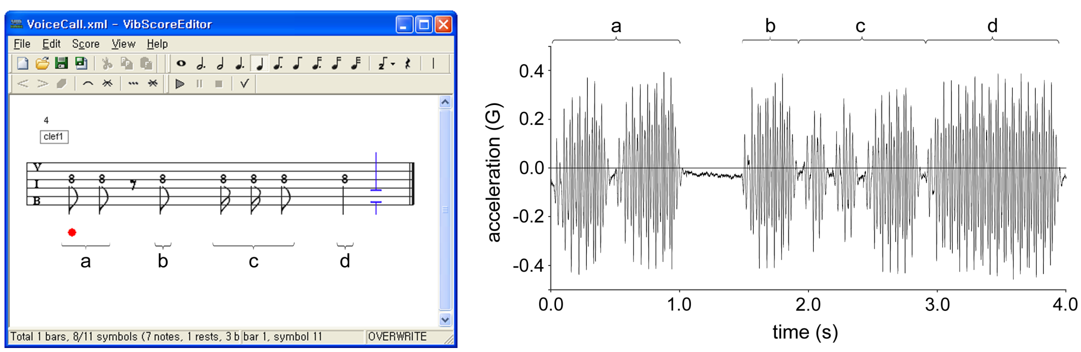
Fig. 5.1: Design example 1: Compound tactons.



(a)Tactons for a mobile phone [5].



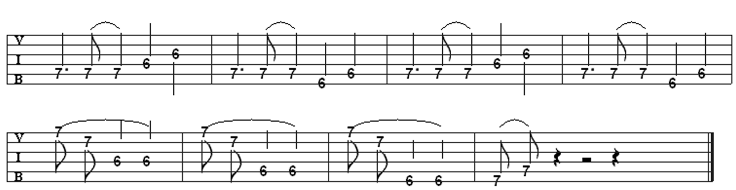
(b)Vibrotactile clef.



(c)Design example of a tacton for ‘high priority voice call’ (left) and resulting acceleration profile from the cellular phone containing a vibration motor (right).

Fig. 5.2: Design example 2: Tactons for a mobile phone.

(a)First part of the “Mission: Impossible” main theme.



(b)Vibrotactile score composed from Fig. 5.3a.

Fig. 5.3: Design example 3: Vibrotactile music.

Chapter 6

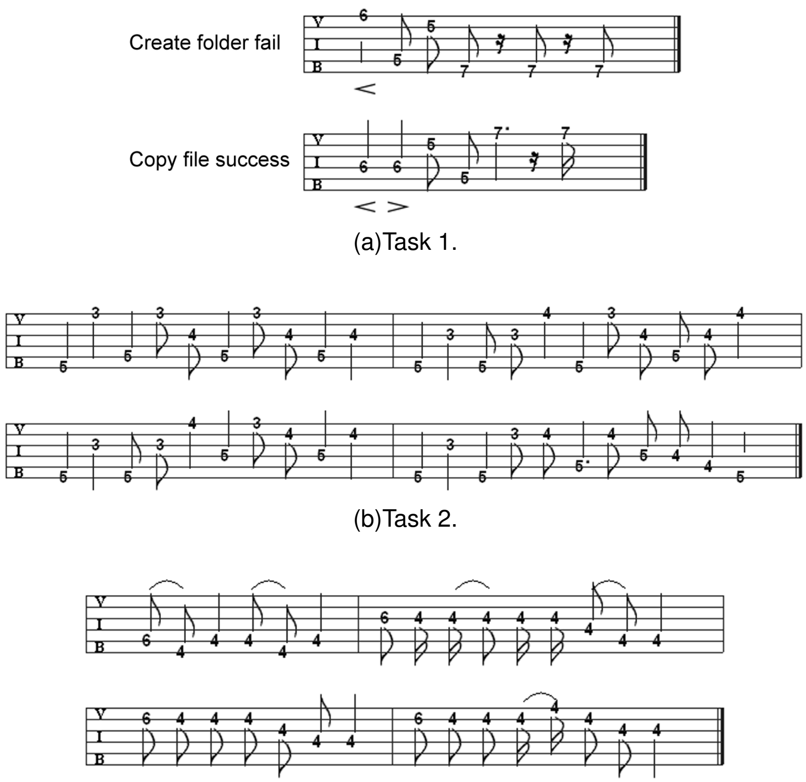
Usability Evaluation 1

In order to evaluate the usability of the vibrotactile score, we have conducted two quantitative user experiments with an emphasis on learnability, efficiency, and subjective preference. The two experiments differed in terms of the representative user group and the design method used for the comparison. In Experiment 1 (reported in this section), the sample user group were experts in vibrotactile pattern design who could provide authoritative and informed assessments. The design method used for the comparison was programming, since it is currently the dominant and most powerful method for vibrotactile pattern design. However, it was infeasible to recruit a large number of participants with expertise in both fields because vibrotactile pattern design experts are still scarce. As an alternative, we used programming experts for this experiment. In Experiment 2 (reported in the next section), the participant group consisted of common users without any prior experience in programming. The design method used for the comparison was GUI-based authoring with a waveform-based editor.

6.1 Methods

6.1.1 Participants

Twelve participants (all male; 21-28 years old) took part in the experiment and were paid for their efforts. All participants were seniors or graduate students with a computer science



(a)Task 1.

(b)Task 2.

(c)Task 3.

Fig. 6.1 Example solutions in vibrotactile scores.

major attending the authors' institution. They had considerable knowledge and experience in programming with the C language and in coding with HTML or XML, but no prior exposure to vibrotactile pattern design. They also passed a simple screening test to confirm their familiarity with basic musical scores and symbols.

6.1.2 Experimental Conditions

The participants implemented vibrotactile patterns using three methods: programming with C, scripting with XML, and graphical editing with the VibScoreEditor. For the C programming, the participants were provided with a complete sample program that included all working functions, e.g., those for device initialization, sinusoidal wave generation, and communication. They were allowed to modify, copy, and paste codes as necessary. For the XML scripting, templates of the XML documents used in the VibScoreEditor (e.g., those presented in Chapter 3) were given along with example files. A separate playback program for XML documents was also provided for testing designed vibrotactile patterns. The participants could select their favorite text editor for editing. For graphical editing with the VibScoreEditor, the participants began with an empty document. The preparation ensured that only the time necessary for the implementation and test of vibrotactile patterns was measured. The time required to make the initial working codes was excluded for the C programming and XML scripting methods.

For each design method, the participants completed three tasks. Task 1 was implementing simple 5-second long compound tactons that consisted of three elements for action, object, and result, respectively, taken from [1]. The participants were given the graphical instructions shown in Fig. 6.2. No musical notations were used since that could have been advantageous to the vibrotactile score. The participants implemented two compound tactons (e.g., 'move file success'). No elementary tactons were used twice for the same participant. One elementary tacton for action was selected as 'create' or 'delete,' and the other; was selected as 'copy' or 'move.' In task 2, the participants were asked to make a long vibrotactile pattern (13.6 seconds) by combining 42 short sinusoidal vibrations. A graphical illustration similar to those in Fig. 6.2 was used to define the pattern. Task 3 was to compose vibrotactile patterns based on music. The participants were presented with the musical score of a very popular Korean pop song ('Tell me' by Wonder Girls), and were asked to compose a vibrotactile music piece that matched the first four measures of the song. To help with the task, a few guidelines were also given; a high-pitch note should have higher frequency than a low-pitch note, a quarter note should be played for 0.8 seconds, and the first note of each measure should have a larger amplitude than the following notes.

In order to illustrate the complexity of the tasks, example solutions for the VibScoreEditor are presented in Fig. 6.1.

To objectively declare that each task was complete, all of the tasks involved implementing specified vibrotactile patterns rather than designing new creative patterns. The latter,

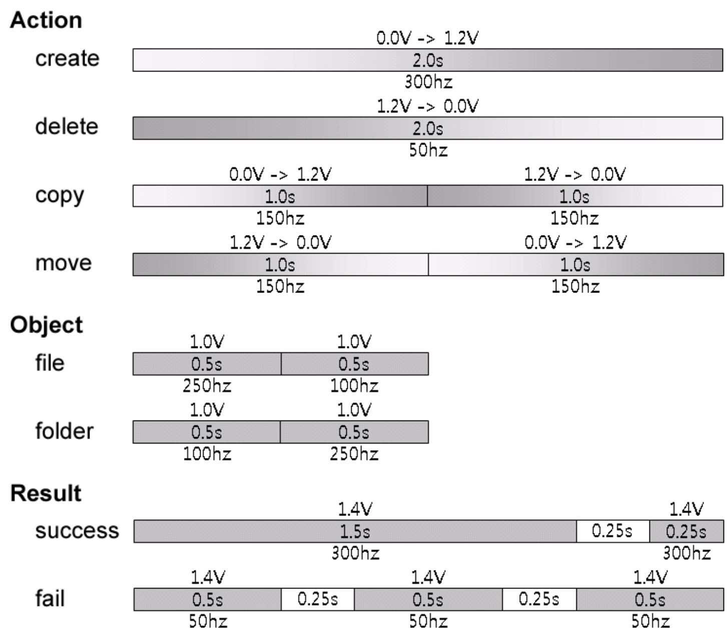


Fig. 6.2 Graphical instructions used in task 1. The length and darkness of a bar denotes the length and intensity of vibration, respectively.

which corresponds to the actual design process, would require much more repetitions of implementation and test. Thus, the differences in some usability metrics, such as the task completion time, can be significantly amplified in practice.

6.1.3 Procedures

The experiment took place during three consecutive days per participant. The experiment of each day consisted of a training session followed by the three main sessions for the three tasks, all done using one design method per day. During the training, the experimenter explained the design method based on a script, which was written and memorized prior to the experiment, to regulate the amount of knowledge provided to the participants. Then, as exercise, the participants implemented three simple vibrotactile patterns that contained patterns useful for the main tasks. After the training, the participants performed tasks 1, 2, and 3 in the increasing order of expected difficulty. It usually took 60 – 90 minutes to complete the one-day sessions. The order of the design methods used on the three days were balanced across the participants using the Latin square to avoid any order effects [31].

The participants were closely monitored in all sessions, and were given a hint if they did not make any progress for more than one minute during the exercise and for more than ten minutes during the main tasks. After the participant declared that the task was completed, the implementation results were inspected. If errors were identified, the participant was asked to fix them. The task completion time was recorded after the participant fixed all of the errors.

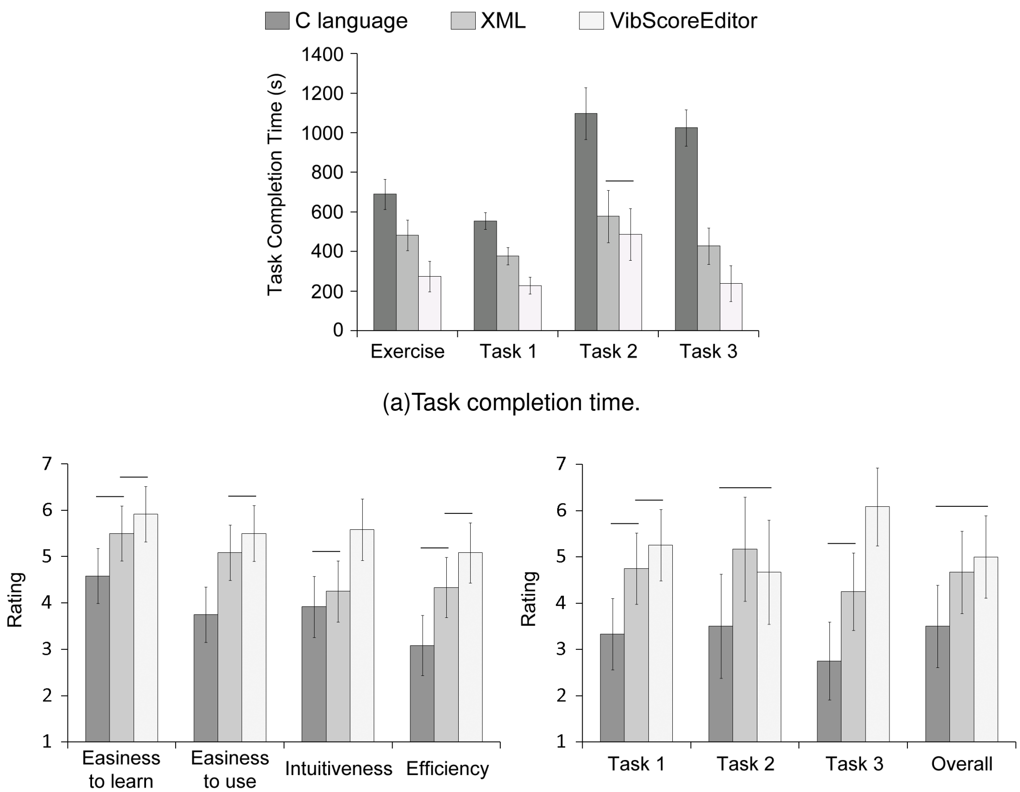
After finishing the experiment sessions each day, the participants filled out a questionnaire to assess the easiness to learn and use, intuitiveness, and efficiency of the design method used in the day. After completing the entire experiment, they were asked for their subjective preference for each design method and each task. All of the questions were rated on a seven-level Likert scale.

6.2 Results and Discussion

The experimental results are summarized in Fig. 6.3, and the results of the one-way ANOVA with the design method as an independent variable are shown in Table 6.1.

Fig. 6.3 shows that the average task completion times of the C language condition were 555 – 1097 seconds (about 9 – 18 minutes), those of the XML condition were 377 – 578 seconds (about 6 – 10 minutes), and those of the VibScoreEditor condition were 228 – 487 seconds (about 4 – 8 minutes). The VibScoreEditor resulted in the smallest average task completion time for all tasks. The design method was a statistically significant factor for the task completion time in all tasks (Table 6.1). Tukey's HSD test indicated that the task completion times were all statistically different, except for task 2 in a comparison between XML and the VibScoreEditor (Fig. 6.3a).

The task completion times had large individual variances, especially in the C language condition, due to the nature of programming. In particular, they were distributed between 545 – 1878 seconds for task 2. In spite of the large variances, the vibrotactile score was more efficient than C programming for all tasks with a statistical significance. The C pro-



(a)Task completion time.

(b)Subjective metrics.

(c)Preference.

Fig. 6.3: The results of usability evaluation 1. Error bars represent Tukey multiple comparison intervals. Line above the bar indicates that there is no clear difference between the design methods (Tukey grouping).

gramming required a 2.4 times longer completion period than the vibrotactile score editing in task 1 which was the simplest, and took 4.3 times longer in task 3 where a vibrotactile pattern was designed from the musical score. The efficiency gain of the VibScoreEditor is expected to be even higher in actual use when vibrotactile patterns are designed and tested repeatedly. It also should be reiterated that all the pre-made functions were given for the C programming activity, which would not be always the case in practice.

The XML condition shared exactly the same data structure as the VibScoreEditor. The major difference between them was whether a text editor or the GUI was used to input data. The VibScoreEditor showed smaller average task completion times than the XML condition

Table 6.1 ANOVA results at significance level $\alpha = 0.05$.

Task completion time	$F_{2,22}$	p
Exercise	23.39	< 0.0001*
Task 1	45.52	< 0.0001*
Task 2	19.89	< 0.0001*
Task 3	63.78	< 0.0001*
Subjective metric	$F_{2,22}$	p
Easiness to learn	4.16	0.0293*
Easiness to use	7.37	0.0036*
Intuitiveness	5.60	0.0108*
Efficiency	7.59	0.0031*
Preference	$F_{2,22}$	p
Task 1	5.26	0.0136*
Task 2	1.83	0.1847
Task 3	12.45	0.0002*
Overall	2.49	0.1060

- Statistically significant cases are marked by *.

with a statistical significance, except in task 2. At the individual data level, XML scripting had slightly shorter task completion times in only 3 cases out of the 48 trials (12 participants $\times$ (exercise + 3 tasks)). This suggests that the use of the score GUI improves efficiency in vibrotactile pattern design.

It should be noted that even in the exercise, the VibScoreEditor resulted in a significantly lower task completion time, indicating its excellent learnability. Before the experiment, the participants were already familiar with C programming and XML scripting, but not with the VibScoreEditor.

In the subjective evaluation (Fig. 6.3b), the VibScoreEditor exhibited the best ratings in all of the subjective metrics: easiness to learn, easiness to use, intuitiveness, and efficiency. The design method had a statistically significant effect in all the subjective metrics (Table 6.1). For user preferences (Fig. 6.3c), the participants preferred the VibScoreEditor for all tasks except task 2 and also in the overall rating. The effect of the design method was statistically significant in tasks 1 and 3 (Table 6.1).

The raw data of the subjective evaluation showed very large individual variances. This

Table 6.2 Summary of verbal comments.

Advantages of the C language
○ Provides the maximum flexibility in pattern design.
○ Easy to learn and use since it is a very familiar language.
Disadvantages of the C language
○ The code tends to grow very long.
○ Memories and all kinds of detailed must be managed by a programmer.
○ It is difficult to debug.
Advantages of the XML
○ Familiar text editors (e.g. Vim) can be utilized.
○ Errors can be easily found and fixed.
Disadvantages of the XML
○ Many new keywords must be memorized.
○ Less convenient than the VibScoreEditor, and less flexible than the C language.
Advantages of the VibScoreEditor
○ Easy to learn and use.
○ Vibrotactile patterns can be managed intuitively and errors can be found easily.
○ No need to concern about memory management and programming details.
○ Allows consistent parameter management via a vibrotactile clef.
○ Has many convenient features (e.g. legato).
Disadvantages of the VibScoreEditor
○ A vibrotactile clef must be referred to frequently.
○ Takes time to learn.

seems correlated with the preference and skills of each participant. The participants were programming experts who usually have strong habits and preferences that have developed over the years. For example, one participant who was an expert at HTML coding using Vim (a popular VI clone) exhibited an extraordinary preference for the XML condition. After finishing task 2, the participant reported that he felt using the VibScoreEditor took longer than using XML since he was new to the VibScoreEditor. In the measured task completion times, however, he spent 64 more seconds for XML scripting. A few other participants showed similar responses; they thought that they spent more time in the VibScoreEditor condition than in the C programming or XML scripting condition, but actually they did not. It is likely that these tendencies led to the individual differences in the subjective ratings.

Verbal comments collected from the participants are summarized in Table 6.2. Most of the comments agreed with our expectations. Noticeably, it was reported that in tasks 1 and 2 the VibScoreEditor was inconvenient since they needed to refer to the vibrotactile clef frequently to convert the properties of vibration to a corresponding note. This was because the vibration properties were specified in terms of numbers for frequency, amplitude, and duration. It seems easier for experienced programmers to copy and paste the necessary text codes and then change several numerical parameters in them, which can the most effective strategy for coding with C or XML. We however note that such an implementation is not common in actual vibrotactile pattern designs. Instead, several predefined properties are combined for "design," with little need to remember their physical definitions. For instance, to design tactile icons, [2] used three roughnesses and three rhythms, [16] used three waveform shapes, four frequencies, and three amplitudes, and [25] used three frequencies and two amplitudes. The vibrotactile score and clef can be much more useful for making a sequence of predefined properties, as demonstrated in Chapter 5.

Chapter 7

Usability Evaluation 2

In this experiment, two GUI-based vibrotactile authoring methods, waveform editing and score editing, were comparatively assessed by common users.

7.1 Methods

7.1.1 Participants

Twelve participants (ten males and two females; 17-25 years old) took part in the experiment, and were paid for their efforts. All the participants were undergraduate students enrolled in the authors' university. They had no prior knowledge or experiences of vibrotactile pattern design. They had moderate skills in computer use (e.g. Microsoft Office), but no programming experience. The participants passed a simple screening test to confirm their familiarity with basic musical scores and symbols. No participants had special music skills. Besides regular school education, the participants did not receive extra music training for more than three years.

7.1.2 Experimental Conditions

The participants implemented vibrotactile patterns using two graphical authoring tools: the posVibEditor for waveform editing [24] and the VibScoreEditor for score editing. The posVibEditor was the only waveform-based authoring tool that we had access to at the

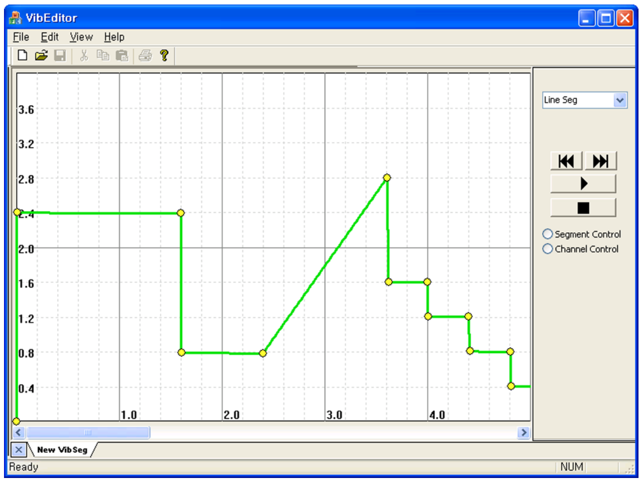


Fig. 7.1 Adapted posVibEditor for the usability evaluation 2.

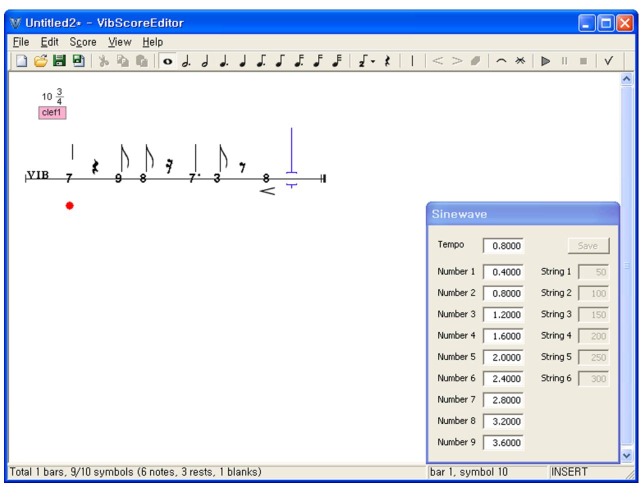


Fig. 7.2 Adapted VibScoreEditor for the usability evaluation 2.

source-code level. Owing to the different functionalities of the two editors, we slightly modified both editors to offer similar capabilities. In the posVibEditor, the multichannel timeline interface for multiple actuators was removed since the VibScoreEditor supports only one actuator (compare Fig. 7.1 to Fig. 3 in [24]). In the VibScoreEditor, only one staff line was used since the posVibEditor was developed for a vibration motor that has a

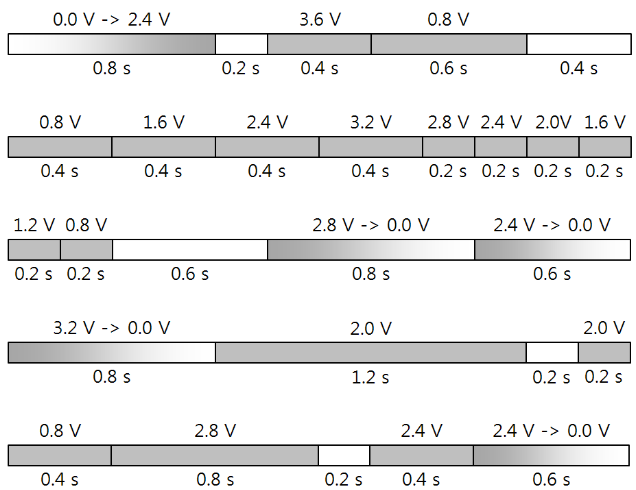


Fig. 7.3 Instruction used in task 2.

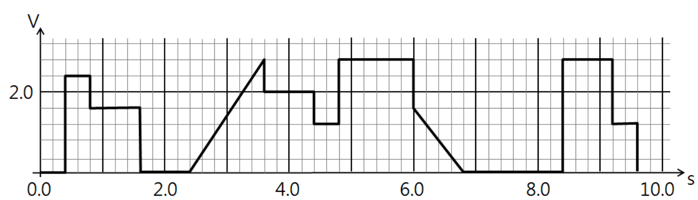


Fig. 7.4 Instruction given as a waveform.

Fig. 7.5 Instruction given as a musical score.

correlated output frequency and amplitude (compare Fig. 7.2 with Fig. 4.1).

The participants completed four tasks with each design method. Task 1 was essentially the same as task 1 of Experiment 1. The graphical instructions in Fig. 6.2 were slightly modified. The frequencies were removed and the voltages were slightly adjusted for the vibration motor used in this experiment. Task 2 was also similar to task 2 of Experiment 1. The participants made a long vibrotactile pattern (12 seconds) as a combination of 27 short sinusoidal vibrations (Fig. 7.3). The next two tasks were to examine the effect of instructions given in different forms. In the literature, several notations were used to describe vibrotactile patterns in the design step, e.g., detailed description [7], waveform [8, 19], and musical score [2, 3]. Both tasks consisted of combining 12 short sinusoidal vibrations to make a 9.6-second long pattern (Fig. 7.4 and 7.5). In the task given as a musical score, we also instructed the participants to play a quarter note for 0.8 seconds and to make the voltages of four notes in each measure 3.6, 1.2, 2.4, and 1.2 V, respectively. When the participants used waveform editing, they solved the task given as a waveform first and then solved the task given as a musical score. When the participants used vibrotactile score editing, the order was reversed.

7.1.3 Procedures

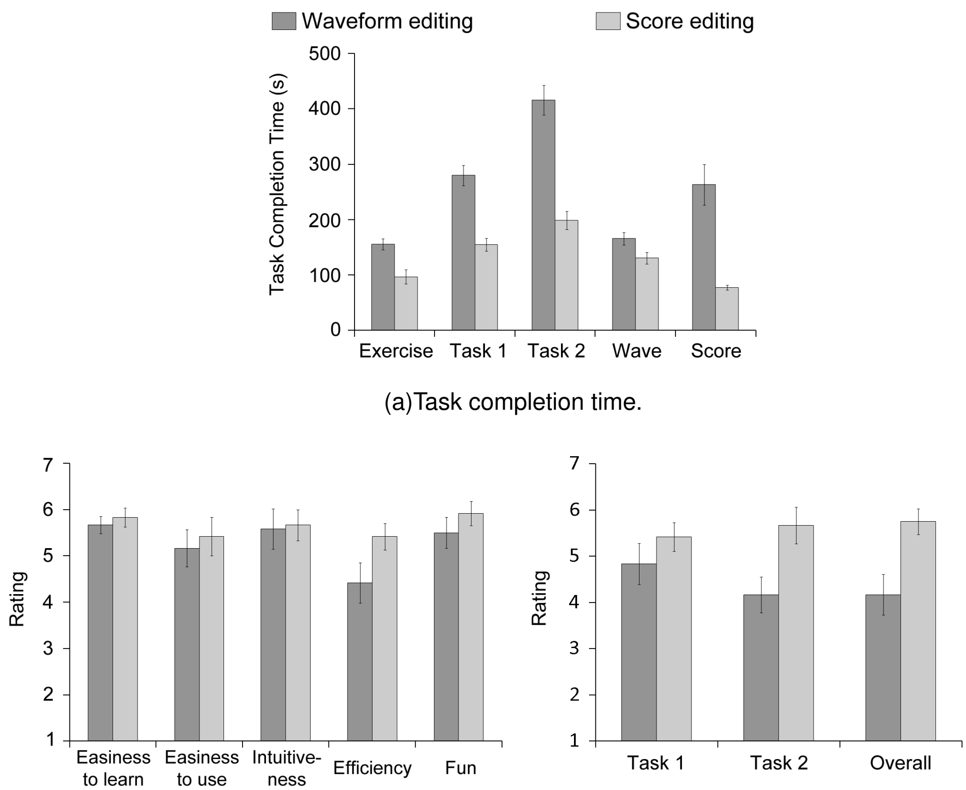
Each participant carried out the experiment on two consecutive days. Six participants used waveform editing on the first day and score editing on the next day. The other six participants followed in the opposite order. Each day, the participants finished the main sessions consisting of the four tasks in addition to a training session. A vibration motor was used as an actuator as the posVibEditor only supports vibration motors. The other procedures were the same as Experiment 1.

The questionnaire was also very similar. In particular, the participants were asked to answer the questions based on the editing methodology itself and not on the functions of the two editors. As the target population was common users, the question for overall subjective preference was changed to be more practical: “How much do you like waveform editing (or vibrotactile score editing) as a part of the functions in your mobile phone?”.

7.2 Results and Discussion

The experimental results are shown in Fig. 7.6. The results of a one-way ANOVA performed with the design method as an independent variable are summarized in Table 7.1.

Editing with the vibrotactile score resulted in lower average task completion times than editing with the waveform for all tasks (Fig. 7.6a). The average task completion time of waveform editing and score editing were in the range of 155 – 416 seconds (about 3 – 7



(b)Subjective metrics. (c)Preference.

Fig. 7.6: The results of usability evaluation 2. Error bars represent standard errors.

minutes) and 77 – 199 seconds (about 1 – 3 minutes), respectively. The differences were statistically significant for all tasks (Table 7.1). Waveform editing took 1.8 times longer than vibrotactile score editing for the simplest task (task 1), and 2.1 times longer for the complex and long task (task 2). This suggests that as a task becomes more difficult, the difference in task completion time may also increase further. Even when the task was given as a waveform, score editing showed a significantly lower task completion time than waveform editing. When the task was given as a musical score, the performance gain was much more evident.

In the subjective evaluation, score editing received better ratings in all the subjective metrics, easiness to learn, easiness to use, intuitiveness, efficiency, and fun (Fig. 7.6b).

Table 7.1 ANOVA results at significance level $\alpha = 0.05$.

Task completion time	$F_{1,11}$	p
Exercise	34.12	0.0001*
Task 1	47.39	< 0.0001*
Task 2	131.02	< 0.0001*
Wave	23.43	0.0005*
Score	28.11	0.0003*
Subjective metric	$F_{1,11}$	p
Easiness to learn	0.65	0.4382
Easiness to use	0.19	0.6742
Intuitiveness	0.03	0.8742
Efficiency	3.88	0.0745
Fun	1.35	0.2691
Preference	$F_{1,11}$	p
Task 1	1.05	0.3283
Task 2	5.03	0.0464*
Overall	7.37	0.0201*

- Statistically significant cases are marked by *.

Although the ratings of score editing were consistently higher, the differences were not noticeable except for efficiency. The differences due to the design method, however, were not statistically significant in any metric (Table 7.1). In addition, the subjective ratings of this experiment were higher than those of C programming and XML scripting of Experiment 1 (compare Fig. 7.6b and 6.3b). This indicates that both methods are easy, intuitive, efficient, and fun, but just have different characteristics. For example, some participants commented that waveform editing was more intuitive since it showed changes in vibration strength using a graph. On the other hand, other participants commented that score editing was more intuitive since it effectively expressed vibration rhythm using familiar musical notations (Table 7.2).

In terms of user preference, the participants preferred vibrotactile score editing more than waveform editing (Fig. 7.6c). Its statistical significance was confirmed by ANOVA except in task 1 (Table 7.1). It appears that the preference of score editing is more apparent in complex and long tasks. In terms of overall preference, the participants gave much higher

Table 7.2 Summary of verbal comments.

Advantages of the waveform editing
○ Easy to learn and use.
○ Intuitive and easy to read since it shows changes in vibration strength over time us-ing a graph.
Disadvantages of the waveform editing
○ Frequent use of mouse is annoying.
○ Takes time to design pattern.
Advantages of the vibrotactile score editing
○ Easy to learn and use.
○ Expresses vibration rhythm effectively and intuitively using musical notations.
○ Efficient since it expresses strength and duration concisely.
○ Interesting and fresh design method.
Disadvantages of the vibrotactile score editing
○ Confusing to novice who is unaccustomed to musical score.
○ Difficult to read strength and duration simultaneously.

ratings for the score editing task. In the individual data, 10 out of the 12 participants preferred vibrotactile score editing. Most participants reported that vibrotactile score editing was a fresh and new idea, and that it was more efficient than waveform editing (Table 7.2). Verbal comments collected from the participants are summarized in Table 7.2.

Chapter 8

General Discussion

In the two usability experiments, the vibrotactile score demonstrated quantitative and qualitative performances that were superior to the other existing vibrotactile pattern design methods of programming with C, scripting with XML, and waveform editing with the posVibEditor. This was unanimous in both the user groups consisting of programming experts and common users, even when the tasks were unfavorable to the vibrotactile score. We believe that the strength of the vibrotactile score stems from its concise and abstract representation of various vibration attributes adapted from the musical symbols that have been refined for centuries. The vibrotactile score may prove more useful in actual authoring where pattern designs and tests are repeated.

Another unique strength is that the vibrotactile score is adequate by its nature for designing vibrotactile patterns from musical sources, as demonstrated in the usability experiments. This feature can be very useful for several important applications, such as tactile icons for information delivery using rhythm variations [17], tactile melodies transformed from music [30], and tactile stimuli for the hearing impaired to feel music [14].

In addition, the vibrotactile score can be valuable in small electronic devices such as mobile devices. Programming or scripting is not an option in mobile devices for obvious reasons. Waveform editing is essentially a continuous process, and is not suitable for a mobile device with a limited number of buttons. It is likely that a touch-screen interface with a stylus would be necessary for waveform editing. In contrast, a vibrotactile score consists of discrete symbols and therefore it can be easily managed with a small number of buttons and/or with a GUI interface using a stylus.

An obvious drawback of the vibrotactile score is the need to learn symbols for users unfamiliar with musical scores. For this reason, we used only basic and simple musical symbols in the design of the vibrotactile score that are usually included in regular elementary education. To assess the learnability in a formal experiment, we attempted to find many participants who could use the computer but not read basic musical symbols, but this search failed as such people were not common. Instead, we asked two participants who did not pass the screening test about the basic musical notations to solve the tasks of Experiment 2. They memorized the basic musical symbols for only five minutes before the experiment. Interestingly, the task completion times were similar to those reported in Experiment 2. The errors found in the results, however, were about five times higher than those of the ordinary users who passed the screening test. Most mistakes were made on the durations of a note and a rest. Although informal, these results also corroborate the excellent learnability of the vibrotactile score, which originated from the use of musical symbols that are already familiar to most people.

Chapter 9

Conclusions

In this paper, we have presented the *vibrotactile score*, an easy and effective method for authoring vibrotactile patterns, by adapting the common musical scores to vibrotaction. The high-level composition process using the vibrotactile score is made independent from the hardware and the low-level physical characteristics of vibration by another metaphoric concept, *vibrotactile clef*. The vibrotactile clef determines the physical roles of score symbols in pattern playback. We also developed *VibScoreEditor*, a graphical authoring tool that implemented the musical metaphor with the data structure defined in XML. Several examples that showed the advantages of the vibrotactile score were given. Finally, the usability of vibrotactile score editing with the VibScoreEditor was evaluated with two user groups and various metrics, which proved the efficacy and adequacy of the vibrotactile score for vibrotactile pattern design.

The source code of the VibScoreEditor is available for downloading at `http://hvr.postech.ac.kr/wiki/wiki.php/VibrotactileScore`.

요 약 문

악보를 사용한 진동 패턴 저작 도구

최근 진동 피드백이 다양한 분야에서 활용되고 있지만, 쉽고 효율적으로 진동 패턴을 디자인하는 방법에 관한 연구는 아직 많이 이루어지지 않았다. 본 논문은 음과 진동의 유사성에 착안하여 피아노 악보와 기타 타브 악보의 표현 방법을 사용해 진동을 디자인하는 방법을 제안한다. 진동 악보와 진동 음자리표라는 두 가지 음악적 메타포를 사용해 높은 수준의 진동 패턴과 낮은 수준의 진동 신호를 분리해서 정의하여 높은 편의성과 확장성, 다양성을 보장할 수 있게 하였다. 또한, 진동 악보와 음자리표의 개념을 실제 구현한 진동 패턴 저작 도구를 제작하였으며, 이를 VibScoreEditor라고 하였다. 몇 가지 실질적인 진동 패턴 저작 예를 통해 진동 악보의 표현력을 확인하였다. 진동 악보의 장점을 검증하기 위해, 학습 및 사용 용이성, 직관성, 효율성, 선호도 중심으로 사용자 평가 실험을 수행하였다. 첫번째 실험은 프로그래밍에 능숙한 전문가 그룹을 대상으로 하여 이뤄졌으며, VibScoreEditor를 통해 진동 악보를 편집하는 방식이 프로그래밍이나 스크립팅 같은 기존의 방법보다 훨씬 더 나은 것을 확인하였다. 두번째 실험에서는 일반 사용자들을 대상으로 파형 조작 방식과 악보 편집 방식의 두 가지 진동 저작 방식을 비교하게 하여, 악보 편집 방식이 파형 조작 방식보다 더 우수한 것을 확인하였다.

Bibliography

[1] S. Brewster and L. M. Brown. Tactons: Structured tactile messages for non-visual information display. In *Proceedings of the 5th Australasian User Interface Conference*, pages 15–23, 2004.

[2] L. M. Brown, S. A. Brewster, and H. C. Purchase. A first investigation into the effectiveness of tactons. In *Proceedings of the First Joint Eurohaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (World Haptics 2005)*, pages 167–176, 2005.

[3] L. M. Brown, S. A. Brewster, and H. C. Purchase. Multidimensional tactons for non-visual information presentation in mobile devices. In *Proceedings of the 8th conference on Human-computer interaction with mobile devices and services*, pages 231–238, 2006.

[4] L. M. Brown, S. A. Brewster, and H. C. Purchase. Tactile crescendos and sforzandos: Applying musical techniques to tactile icon design. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, pages 610–615, 2006.

[5] L. M. Brown and T. Kaaresoja. Feel who's talking: Using tactons for mobile phone alerts. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, pages 604–609, 2006.

[6] E. Catmull and R. Rom. A class of local interpolating splines. In R. E. Barnhill and R. F. Riesenfeld, editors, *Computer Aided Geometric Design*, pages 317–326. Academic Press, Orlando, FL, USA, 1974.

[7] A. Chan, K. MacLean, and J. McGrenere. Learning and identifying haptic icons under workload. In *Proceedings of the First Joint Eurohaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (World Haptics 2005)*, pages 432–439, 2005.

[8] M. Enriquez, K. MacLean, and C. Chita. Haptic phonemes: Basic building blocks of haptic communication. In *Proceedings of the 8th international conference on Multimodal interfaces*, pages 302–309, 2006.

[9] M. J. Enriquez and K. E. MacLean. The Hapticon Editor: A tool in support of haptic communication research. In *Proceedings of the International Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pages 356–362, 2003.

[10] A. Gallace, H. Z. Tan, and C. Spence. The body surface as a communication system: The state of the art after 50 years. *Presence*, 16(6):655–676, 2007.

[11] C. F. Goldfarb and P. Prescod. *The XML handbook*. Prentice Hall, Upper Saddle River, NJ, USA, 1998.

[12] Immersion Corp. Mobility overview, 2007. `http://www.immersion.com/mobility`.

[13] L. A. Jones, B. Lockyer, and E. Piateski. Tactile display and vibrotactile pattern recognition on the torso. *Advanced Robotics*, 20(12):1359–1374, 2006.

[14] M. Karam, F. Russo, C. Branje, E. Price, and D. I. Fels. Towards a model human cochlea: Sensory substitution for crossmodal audio-tactile displays. In *Proceedings of Graphics Interface Conference*, pages 267–274, 2008.

[15] B. Larsen. *Power Tab Editor*, 2000. `http://www.power-tab.net/`.

[16] K. MacLean and M. Enriquez. Perceptual design of haptic icons. In *Proceedings of Eurohaptics*, pages 351–363, 2003.

[17] K. E. MacLean. Foundations of transparency in tactile information design. *IEEE Transactions on Haptics*, 1(2):84–95, 2008.

[18] D. A. Mahns, N. M. Perkins, V. Sahai, L. Robinson, and M. J. Rowe. Vibrotactile frequency discrimination in human hairy skin. *Journal of Neurophysiology*, 95:1442–1450, 2006.

[19] T. McDaniel, S. Krishna, D. Colbry, and S. Panchanathan. Using tactile rhythm to convey interpersonal distances to individuals who are blind. In *Proceedings of the 27th international conference extended abstracts on Human factors in computing systems*, pages 4669–4674, 2009.

[20] J. W. Morley and M. J. Rowe. Perceived pitch of vibrotactile stimuli: Effects of vibration amplitude, and implications for vibration frequency coding. *The Journal of Physiology*, 431:403–416, 1990.

[21] B. J. P. Mortimer, G. A. Zets, and R. W. Cholewiak. Vibrotactile transduction and transducers. *Journal of the Acoustical Society of America*, 121(5):2970–2977, 2007.

[22] A. M. Okamura, M. R. Cutkosky, and J. T. Dennerlein. Reality-based models for vibration feedback in virtual environments. *IEEE/ASME Transactions on Mechatronics*, 6(3):245–252, 2001.

[23] J. Ryu and S. Choi. Benefits of perceptually transparent vibration rendering in mobile device. *Lecture Notes in Computer Science (EuroHaptics 2008)*, 5024:706–711, 2008.

[24] J. Ryu and S. Choi. posVibEditor: Graphical authoring tool of vibrotactile patterns. In *Proceedings of the IEEE International Workshop on Haptic Audio Visual Environments and Games*, pages 120–125, 2008.

[25] J. Ryu, J. Chun, G. Park, S. Choi, and S. H. Han. Vibrotactile feedback for information delivery in the vehicle. Accepted for publication in IEEE Transactions on Haptics, 2009.

[26] J. Ryu, J. Jung, S. Kim, and S. Choi. Perceptually transparent vibration rendering using a vibration motor for haptic interaction. In *Proceedings of the IEEE International Symposium on Robot & Human Interactive Communication*, pages 310–315, 2007.

[27] C. Sherrick. A scale for rate of tactual vibration. *Journal of the Acoustical Society of America*, 78(1):78–83, 1985.

[28] C. Spence and C. Ho. Tactile and multisensory spatial warning signals for drivers. *IEEE Transactions on Haptics*, 1(2):121–129, 2008.

[29] C. Swindells, E. Maksakov, K. E. Maclean, and V. Chung. The role of prototyping tools for haptic behavior design. In *Proceedings of the Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pages 161–168, 2006.

[30] J. B. van Erp and M. M. Spape. Distilling the underlying dimensions of tactile melodies. In *Proceedings of Eurohaptics*, pages 111–120, 2003.

[31] B. J. Winer, D. R. Brown, and K. M. Michels. *Statistical Principles in Experimental Design*. McGraw-Hill, 3rd edition, 1991.

[32] J. Zhou, X. Shen, I. Shakra, A. El Saddik, and N. Georganas. XML-based representation of haptic information. In *Proceedings of the IEEE International Workshop on Haptic Audio Visual Environments and Their Applications*, pages 116–120, 2005.

감 사 의 글

낯선 포항에 온지 9년이라는 긴 시간이 흘러 드디어 졸업을 하게 되었습니다. 고등학교 때 최고의 프로그램 제작자가 되겠다는 꿈을 꾸며 포항공대에 온 이후 이곳은 이제까지 제 삶에서 가장 소중한 곳이 되었습니다. 이 곳에서 있었던 모든 일이 부족하지만 그래도 지금의 저를 길러준 소중한 밑거름이 되었습니다.

아는 것도 별로 없고 부족하기만 한 저를 잘 지도해 주셔서 좋은 연구 결과를 내고 졸업할 수 있게 도와주신 최승문 교수님께 먼저 큰 감사를 드립니다. 제 석사 연구와 논문을 심사해주시고 좋은 지적을 해주셨던 이승용 교수님과 박성우 교수님께도 감사를 드립니다.

제 초기 도자기 랜더링 확장 연구에 많은 도움을 주셨던 성길이 형, 석사 논문 연구의 핵심적인 아이디어를 주신 종현이 형, 언제나 열심히 하시고 NRL 연구 중에 좋은 조언자가 되어준 석희 형, 물어보면 이런 저런 자료 잘 보내주셨던 성훈이 형, 항상 꾸준히 성실하면서 언제나 편안하게 도움을 청할 수 있었던 인욱이, 일 처리가 꼼꼼하고 섬세한 인이, 척척박사 모르는 게 없는 갑종이, 하드웨어를 도맡아 늘 고생하면서도 사소한 일로 이것저것 많이 부탁할 수 밖에 없었던 건혁이, 잡다한 부탁 들어주느라 고생하면서도 항상 열심히 해준 종만이, 2년 간의 짧지 않은 석사 과정을 함께 했던 연구실 선후배들에게도 제가 많은 도움을 주지는 못하고 늘 좋은 것을 받기만 한 것 같아 미안하고 감사한 마음뿐입니다.

대학 생활뿐만 아니라 내 인생에 가장 소중한 친구들인 분반 친구들에게도 정말 감사합니다. 항상 함께 재미있게 지냈고 이제 같은 회사 입사 동기가 될 으뇽이, 포항에 남아 자주 못 보겠지만 고맙고 소중한 친구 강씨, 늘 재미있는 성수, 얄밉지만 좋은 친구 종민이, 든든한 창수, 미래의 의사 선생님 승백이와 민호, 박사가 되면 지구를 구할 노루, 감칠맛 나는 욕의 씹탱 등등. 대학원 오면서 거의 보지 못했지만 학부 생

활 동안 좋은 친구와 동료가 되어준 수학과 선후배들, 현욱이 형, 재현이 형, 영준이 형, 진명이 형, 창우 형, 재우, 써킹, 성현이. . . 탁구 동아리 후배 명선이, 종우. . . 배울 점 많은 믿음의 분반 후배 인모, 대학원에서 많은 도움을 주었던 오조교, 진하형. . . 감사하고 고맙고 아쉬운 마음뿐입니다.

항상 내 마음과 영혼의 편안한 안식처와 내 인생의 기쁨과 소망이 되었던 효자 교회 공동체 많은 분들에게 감사합니다. 인생과 신앙에 늘 많은 조언을 해준 현철이 형, 항상 유쾌한 일준이, 가장 가까운 선배였던 신효 형, 성실하고 믿음직한 헌호, 많이 못 도와 드려 죄송한 김규민 도사님, 함께 임원 했던 신애 누나, 민희 누나, 해룡 형, 경주 누나, 본받을 만한 리더장 은경 누나, 멋진 헌신의 주일학교 선생님들, 좋은 말씀으로 가르침 주신 이하준 목사님, 박성민 전도사님, 김태호 전도사님 등, 많은 분들 한 분 한 분 감사 드리지 못하여 죄송한 마음입니다. 제게 가장 소중한 아이들, 포항 생활에 큰 기쁨과 보람이 되어주었던 중등부 소년부 학생들 고맙습니다.

마지막으로 언제나 저를 믿고 지원을 아끼지 않으셨던 어머니 아버지, 부족한 저를 아껴주시고 기다려주신 장인어른, 장모님께 감사 드립니다. 못난 남편 믿고 따라 주며 많이 고생했던 인생의 동반자, 동역자 여보 그리고 시온이 사랑하고 미안하고 고맙습니다. 이제까지 지켜주시고 힘과 도움이 되어주신 내 삶의 반석이신 하나님 감사합니다.

Curriculum Vitae

Name	:	Jaebong Lee
Date of Birth	:	1982. 5. 13
Present Address	:	경북 포항시 남구 효자동 포항공대 대학원아파트 3-105

Education

2001–2008	:	B.S. in Mathematics, POSTECH
2008–2010	:	M.S. in Computer Science and Engineering, POSTECH Thesis Title : 악보를 사용한 진동 패턴 저작 도구**(Vibrotactile Score for Designing Vibrotactile Patterns)** Advisor: Prof. Seungmoon Choi

Publications

1. Jaebong Lee, Jonghyun Ryu, and Seungmoon Choi "Vibrotactile Score: A Score Metaphor for Designing Vibrotactile Patterns" *in Proceedings of World Haptics Conference*, pp. 302–307, 2009.

2. Jaebong Lee, Jonghyun Ryu, and Seungmoon Choi "Graphical Authoring Tools for Vibrotactile Patterns" *in DVD Proceedings of World Haptics Conference*, pp. 388–389, 2009 (Demonstration).

3. Jaebong Lee and Seungmoon Choi "Effects of Haptic Guidance and Disturbance on Motor Learning: Potential Advantage of Haptic Disturbance" *To be presented in the Haptics Symposium (HS)*, 2010.

4. Jaebong Lee, Gabjong Han, and Seungmoon Choi "Haptic Pottery Modeling Using Circular Sector Element Method" *Lecture Notes on Computer Science (EuroHaptics 2008)*, pp. 668–674, 2008.

5. Jaebong Lee and Seungmoon Choi "Improvement of the Haptic Pottery Modeling System Using Circular Sector Element Method" *in Proceedings of the International Conference on Mechatronics and Information Technology (ICMIT)*, pp. 7-9, 2009.

6. Gabjong Han, Jaebong Lee, In Lee, Seokhee Jeon, and Seungmoon Choi "Effects of Kinesthetic Information on Memory Chunking in 2D Sequential Selection Task" *To be presented in the Haptics Symposium (HS)*, 2010.

7. 이재봉, 이인, 한갑종, 전석희, 최승문 “이차원 순차 선택에서의 햅틱 정보 제공을 통한 단기기억능력 향상” 한국 지능로봇 종합 학술대회 논문집, pp. 455–457, 2009.

8. 이재봉, 한갑종, 최승문 “부채꼴 요소법을 이용한 3차원 도자기 모델링” 한국 HCI 학술대회 논문집, 1권, pp. 78–84, 2008.