Third Joint Eurohaptics Conference and Symposium on Haptic Interfaces
for Virtual Environment and Teleoperator Systems
Salt Lake City, UT, USA, March 18-20, 2009

A 2D Haptic Glyph Method for Tactile Arrays : Design and Evaluation

Christoph W. Borst

University of Louisiana, Lafayette

Vijay B. Baiyya

University of Louisiana, Lafayette

ABSTRACT

We present a new framework for information cue rendering on 2D vibrotactile arrays, and we describe an experiment that investigated the feasibility of our approach. The methods are broadly applicable, but our work is motivated by the potential for a tactile array to provide an additional useful channel for information such as location cues related to dataset features or remote user behaviors in visualization systems. Our experiment measured the accuracy with which three basic haptic glyph parameters (position, direction, and an intensity profile) communicated information to users. Results show that multiple parameters can be communicated simultaneously by a glyph, although with reduced accuracy in some cases. In these cases, we give insight into relevant effects to guide the design of improved glyphs. Besides these results, novel contributions of our work are the general information cue approach for 2D arrays (2D haptic glyphs) and the extension of graphical visualization techniques to haptics (glyphs, spatial anti-aliasing, gamma correction).

KEYWORDS: Haptics, Haptic Glyphs, Haptic Rendering, Vibrotactile Array.

INDEX TERMS: H.5.2 [Information Interfaces And Presentation (e.g., HCI)]: User Interfaces—Haptic I/O

1 INTRODUCTION

We present methods for information cue rendering on 2D vibrotactile arrays. To date, most methods have been ad-hoc, with minimal generality. In contrast, the haptic glyph method presented here, combined with our previous rendering work [1], provides a framework that is applicable to a range of 2D array configurations and encompasses and generalizes many of the cues that have been rendered on 2D arrays in past research. Futhermore, it extends previous approaches by considering information cues that encode multiple channels of (potentially) continually-changing information. As an example, we discuss the use of a low-cost palm-sized vibrotactile array to provide information about location and behavior of remote users in a collaborative VR system, i.e., a tactile version of a heads-up map.

We briefly introduced the 2D haptic glyph concept in [2], but extend it in this paper and present the first experimental results. The new experiment investigated the feasibility of haptic glyphs by measuring the accuracy with which three parameters could be communicated. Results are promising, especially considering the limitations of the low-cost experiment apparatus and the use of novice subjects. More importantly, results provide specific insight into possible problems and corresponding remedies.

2 MOTIVATION: EXAMPLE APPLICATION

The use of low-cost vibrotactile elements (tactors) is increasingly common in haptics research and applications. Small vibrating DC motors and Piezo speakers have been used to construct haptic displays with low power requirements, low cost, low weight, portability, and simple implementation. Commonly-explored configurations include elements placed in clothing or used to build arrays on the backs of seats to communicate shape, orientation, direction, or attention cues for both real and virtual environments, e.g., [3-6]. Experimental studies have verified that such cues can be communicated.

Figure 1 shows a 2D vibrotactile array prototype used to test concepts described in this paper, as well as an illustrative example of the array being used with a "fishtank" VR system. In this example application, we considered use of the array in a collaborative geosciences visualization system [7] to communicate information about remote users or datasets features. Besides the vibrotactile array, the pictured hardware includes a mirror-based fishtank-style VR display with a Phantom force-feedback stylus as the interaction tool.

We believe that the introduction of a tactile array into this type of VR display may provide an additional useful channel for information, especially location cues, with minimal added cost or cumber. While the Phantom device can provide contact cues for stylus interactions and other force cues, earlier-cited research suggests a vibrotactile array is suited to location cues, which do not map so directly to the Phantom. Besides our VR setting, tactile arrays can be used with conventional 2D desktops, can be built in various form factors, and might one day be used in other applications such as cues for vehicle operators. Smaller tactors that are already available allow arrays to be made smaller or denser and adapted to mount on a mouse or hand-held controller.

Using an array as a tactile map of remote users' positions and actions addresses a problem that is particularly notable in fishtank VR displays due to their small field of regard: avatars or view proxies conveying similar information can be difficult to find or keep track of when not immediately in front of the user. Even if visual indicators (such as a graphical heads-up map) are kept in view, they reduce space available for dataset views and interactive objects, and may distract visual attention from the task being performed.

3 ARRAY PROTOTYPE; CAPABILITIES AND LIMITATIONS

Before describing the 2D haptic glyphs, we briefly review our array design, previous work on rendering, and array limitations. Note that to apply our work to other 2D arrays, they merely need to be accessible as a 2D matrix of intensity values. Only the low-level driver is specific to the vibrotactile display being used.

Our vibrotactile array prototype consists of 30 vibrating motors on a controller box with a serial connection to a PC. The tactors (Sanko Electric IE120) form a 5×6 grid with grid spacing of about 18mm and are mounted on foam pads to help isolate them from each other and to help the array conform to hand shape. We measured the fundamental vibration frequency of tactors to range from 27 Hz to 100 Hz, depending on contact pressure and input.

At the lowest level, the controller treats the array as a monochrome raster (30 bits). However, we use pulsewidth modulation (PWM) with a switching rate of roughly 300 Hz to provide 23 tactor intensity levels, allowing applications to treat the array as a grayscale raster. Furthermore, driver software

cwborst@gmail.com, vijay.609@gmail.com
P.O. Box 44330, Lafayette, LA, 70504, USA

provides a function for rendering point primitives expressed with respect to an array coordinate system. Line segments or curves are rendered as moving point traces, because static raster images spanning several motors are more difficult to perceive meaningfully (e.g., see discussion in [3]). A trace is generated by evaluating a parametric equation describing a point on a curve as a function of time. A conversion of the point to actual tactor intensities, described next, allows an application designer to focus on generating point or curve primitives rather than specific tactor activation sequences.

A distinguishing feature of our array rendering approach is our use of spatial anti-aliasing and gamma correction. We use unweighted area sampling to compute tactor intensities for a point primitive, i.e., the intensity for each tactor is the area of overlap between its unit-area grid cell and a unit square centered on the rendered point coordinate. As an alternative to this, we also include other rendering methods, including a bi-level "closest-tactor" method and an anti-aliased "interpolated midpoint" method, as detailed in previous work [1]. Any nonzero intensity I computed by this rendering step is furthermore adjusted by the function:

$$I_{adjusted} = (1-\mu)(I)^{\frac{1}{\gamma}} + \mu$$

where μ is a threshold parameter that sets minimum nonzero tactor intensity (e.g., minimum level resulting in vibration) and γ is a gamma parameter that can adjust for nonlinearities in tactor or perceptual responses (or can be tuned to preference). Our driver selects the PWM pattern having duty cycle closest to $I_{adjusted}$.

We previously investigated these techniques using human factors and psychophysics methods. We showed that anti-aliasing, by either area sampling or an interpolated midpoint technique, improves perceived quality. And, we showed that the combination of area sampling and proper selection of μ and γ allows users to detect the direction of short line segments rendered on the array, even for line segments with length below one grid spacing. This result and results from earlier-cited research suggest that low-cost tactor arrays are usable for location or direction cues rendered as points or line traces.

On the other hand, limitations of this array type should be noted. Resolution is low and the low-cost DC tactors do not allow precise or independent control of vibration frequency and amplitude. Shape discrimination is difficult when it hinges on detection of corners in traced shapes (e.g., square vs. circle [1]). Beats (low-frequency pulses) can occur when multiple tactors vibrate simultaneously at different frequencies, but our previous experiments nonetheless showed quality improvements for anti-aliased approaches that use multiple tactors. Perceivable beats can be reduced by avoiding a static or slow moving stimulus, so any remaining beats are transient. Finally, motors can have a low-pass filtering effect on intensity profiles of our glyph mechanism (presented in Section 4.1.2), limiting the range of possible effects. Note that the severity of these limitations depends on the particular tactor and control technology used, but a main purpose of this paper is to present concepts that generalize to other arrays with different specific capabilities and limitations.

4 Supporting Haptic Visualization

We built on our earlier work by developing higher-level mechanisms to support haptic visualization (sometimes called haptization or haptification) of information for applications such as our geosciences application. The main extension to our framework is the development of a haptic glyph mechanism that matches well to 2D array capabilities.

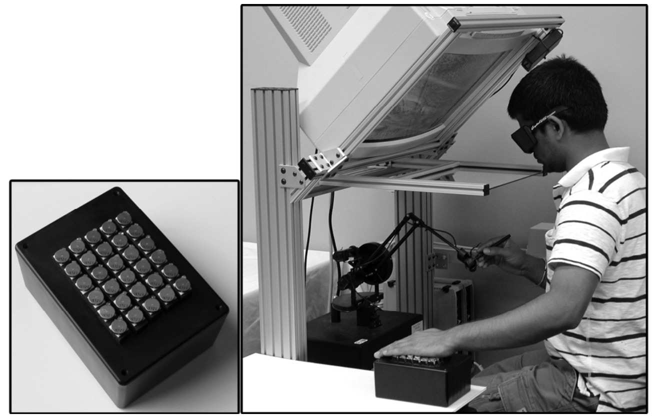

Figure 1. Left: An example of a 2D vibrotactile array (the palm-size array used for experiments presented in Section 6). Right: User of a visualization system with the free hand on the array.

4.1 Haptic Glyphs

4.1.1 Glyph Concept and Related Work

A glyph is an object that is modified by input data to communicate information. Glyphs are well-known in graphical visualization, where they are considered types of icons (e.g., Demarcelle and Hesselink [8]), but Roberts and Franklin [9], who presented haptic glyphs, distinguished them from haptic icons by explaining that glyphs actively encode information mapped to multiple parameters, while icons have constant form and unique association. Based on this, work on haptic glyphs is minimal but is related to techniques that have been called haptic icons and tactons. These techniques have been applied primarily to single-transducer systems. Roberts and Franklin briefly proposed force glyphs for the Phantom using grooves and caverns [9], while Osawa investigated sequence and strength patterns for Cybertouch glove-mounted tactors to represent abstract information (calling them tactile glyphs) [10]. Enriquez and MacLean introduced editable force profiles called haptic icons for a force feedback knob [11], which led to later investigations of haptic icons for other transducers (e.g., [12]). Brewster and Brown describe tactons and potentially useful parameters such as frequency, amplitude, and duration, which could represent different properties [13]. Their later work used up to three transducers, spaced apart on an arm. However, none of these works provided a general rendering framework for arrays such as ours.

Our glyph parameters and glyph software architecture support a wide range of effects and, together with the methods summarized in Section 3, provide a more general framework for rendering a broad range of information cues on 2D tactor arrays.

4.1.2 Basic Haptic Glyph Parameters

Our glyph mechanism allows an application to generate a glyph by specifying any subset of the following parameters:

Shape: The shape to render, specified by a list of curve segments. For example, a linear segment is specified by two 2D endpoints P_0 and P_1, and the system traces this segment by varying t from 0 to 1 in $(P_0 + (P_1 - P_0)t)$. This approach extends readily to standard curve types such as 2D Bézier segments. If no shape is specified, the system defaults to using a point (the origin).

Position: Specifies a 2D translation applied to the glyph, or zero by default.

Orientation: Specifies a rotation (one angle) applied to the glyph, or zero by default. The rotation is applied in the glyph's local coordinate frame, which may be translated relative to the array frame.

Scale (2D): Specifies a 2D scale factor, or no scaling by default. This transformation acts in the glyph's local frame, which may be translated and rotated relative to the array frame.

Count: Specifies how many more times the shape should be traced, with a default of the maximum representable value.

Durations: Specifies timing for shape tracing and is used to compute *t*-increments needed to trace segments. Two different forms are supported for this specification: A pair of values can be given to specify total shape trace time and a delay following the trace, or a list of pairs can be given to specify this on a per-segment basis. Default value is one second per-shape trace time and no delay.

Intensity Profile: When specified, this is a profile used to modulate tactor intensity. It can be specified on a per-shape, per-segment, or real-time basis. An intensity profile consists of a list of time-intensity pairs (linear interpolation generates intermediate values). For per-shape or per-segment specification, time values are multiplied by duration so they are specified in normalized form. For real-time specification, values are in seconds with no normalization, and additional offset and repeat values are available to control repeating patterns. See Figure 2 for three simple examples of profiles we have used.

Type/Priority: This integer value affects the priority and behavior of the glyph. The default behavior is for glyphs to be rendered repeatedly, subject to the count parameter, as changes to parameters are applied between traces. Multiple glyphs can be rendered serially (simultaneous display is likely to make glyphs uninterpretable). The system additionally supports two special glyph types called event and superimposed glyphs. An event glyph interrupts other glyphs and can therefore be used to immediately signal an event. A superimposed glyph, on the other hand, replaces just a subset of the current glyph parameters. For example, an intensity profile can be superimposed onto whatever glyph is already being rendered in order to signal an event.

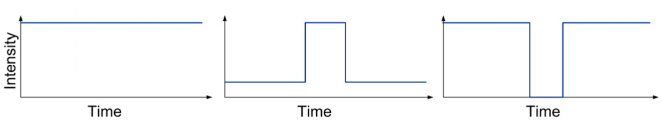

Figure 2. Example Intensity Profiles (before gamma correction). Left: Constant high intensity. Middle: Mid-level intensity with high-intensity pulse, at center of trace, for 20% of the trace duration. Right: Intensity dip at center of trace, for 13% of trace duration.

4.1.3 Glyph Mechanism Software Architecture

Figure 3 overviews the software architecture for our tactile glyph mechanism. The system provides an application programmer with a set of methods for managing glyphs and their rendering. Any requested changes to glyphs (e.g., changing a parameter value) are immediately applied to a representation of the glyph stored in a container, Container 1, that maintains a copy of all created glyphs and a table of intensity profiles that can be accessed by the application through an API (application programming interface).

To avoid artifacts that would result from glyph updates being applied in the middle of a trace, another representation of glyphs and profiles is stored in a second container, Container 2, and the two containers are connected by a thread-safe update queue to communicate requested changes to a glyph rendering thread without ever suspending that thread. The rendering thread also manages Render and Superimpose queues, which are small queues containing only IDs of glyphs to be rendered according to the behavior described earlier for the type parameter. For simple applications, these queues may never need to contain more than one glyph ID, but their inclusion supports a broader range of behaviors than maintaining only a single active glyph. The Render queue can store multiple glyphs in order of priority. When a high-priority glyph arrives, the rendering of a lower-priority glyph can be interrupted and the higher-priority glyph rendered. Once the higher-priority glyph is finished rendering, the lower-priority glyph can either be resumed or restarted. The superimpose queue can store multiple superimposed glyphs to handle the case of multiple superimpose requests by the application.

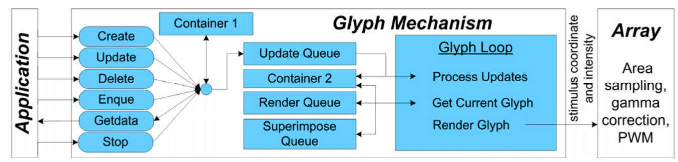

Figure 3. Glyph Mechanism Software Architecture

4.2 Dataset Feature Extraction

In the example visualization application, methods are needed for mapping information about the environment to glyph parameters. In some cases, such as a collaborative map discussed in Section 5, information is already in a form that readily maps to glyph parameters. In other cases, such as haptic representation of terrain data features, information must be converted to a suitable form. For example, in our geosciences application, extrema and crevices in topographic data (e.g., Shuttle Radar Topography Mission data and laser ranging meshes) are studied and compared to features in associated geophysical datasets (e.g., gravity magnitude). A haptic display may be useful for conveying properties such as position, shape, magnitude, or direction for such features when not directly visible. For example, a feature of interest may be associated with a secondary dataset, may not be illustrated by current visual parameters, or may be out of view due to scaling or viewpoint.

For example, an extrema detection mechanism can consider a dataset subregion inside a region selector that follows the user's interaction tool during exploration, i.e., a region near the Phantom tool tip. The data may be from a secondary dataset that differs from visually displayed data, and the size of the region selector can be adjusted and may correspond to a much different scale than that seen in the visual display. The mechanism can find minima or maxima in the selector range and extract coordinates and values for mapping to glyph parameters.

5 Glyph Example: Tactile Map for Collaboration

As motivated in Section 2, we believe one suitable application of glyphs is to haptically convey information about remote users during collaboration. The main approach is to map position of a remote user to haptic glyph position as a location cue. Here, position may refer to head or body position, interaction tool position, or a view target, depending on intent. To create the haptic version of a heads-up map (forward-up map, as opposed to fixed-orientation map), the local user's pose (position and orientation) is also used, to convert remote user pose to relative form. In addition to the location cue, this relative pose is used to provide orientation of the remote user by controlling glyph orientation, with glyph shape being a short line segment as an orientation vector. Identity of the remote user can be mapped to intensity profile of the glyph.

Discrete actions of the remote user may be communicated using event or superimposed glyphs that signal the event using an intensity profile. Examples of events include a user joining or leaving the session, placing an interpretive mark on a dataset, or switching views between multiple co-located datasets. Repeating pulses or patterns could be used to continually remind a user which of the co-located datasets the collaborator is exploring, potentially reducing confusion about differences in view. As suggested by existing work on haptic icons, intensity profiles may also be effective for mediating turn-taking for collaboration without intrusive visual indicators [12]. For example, one-shot profiles could indicate gain or loss of control, while periodically-displayed profiles could remind a user of pending requests for control. To contrast these examples, a much simpler use of intensity profiles is to simply map information to an intensity level that is constant per glyph trace, e.g., to represent virtual distance between users.

6 EVALUATION

6.1 Experiment Design Overview

We conducted an experiment to evaluate the feasibility of 2D vibrotactile glyphs and to gain insight into their effective use. Experiment subjects were required to identify position, direction, and intensity profile of glyphs that consisted of line segments (except in certain simplified conditions described shortly).

Trace time (duration) of glyphs was 1 second, with a glyph traced 10 times by initializing its *count* parameter to 10.

We used 84 trials (glyphs) per subject, in addition to practice runs. These were divided into 7 groups of 12 trials each, with each group corresponding to a different condition. The conditions were:

FULL - Full glyph: the subject felt a line segment glyph having position, orientation, and an intensity profile, and the subject estimated these values. Position was the center of the trace. Profile was one of DIP, RISE or CONSTANT, as seen in Figure 2.

POS - Position only: the subject felt only a static rendered point and estimated only its position.

DIR - Direction only: the subject felt a line segment glyph with constant intensity profile and estimated only its direction.

PROF - Profile only: the subject felt an intensity profile rendered at a single point and identified only profile type, selecting from DIP, RISE, or CONSTANT.

Hint conditions (3): in three additional conditions, subjects received hints about some parameters while estimating the values of others. These conditions are omitted from much of the following discussion for clarity, but are discussed in Section 6.5.4.

Thus, the experiment had a within-subjects design. We generated a randomized sequence of position, orientation, and profile values and then used the same sequence and glyph order for all subjects. However, order of condition groups above was randomized in a balanced manner across subjects, so specific glyphs were not tied to specific conditions, and order effects were addressed. Additionally, each group of 12 trials was further split into 2 groups of 6 to make it unlikely that subjects could track the number of times an intensity profile had already occurred in a condition.

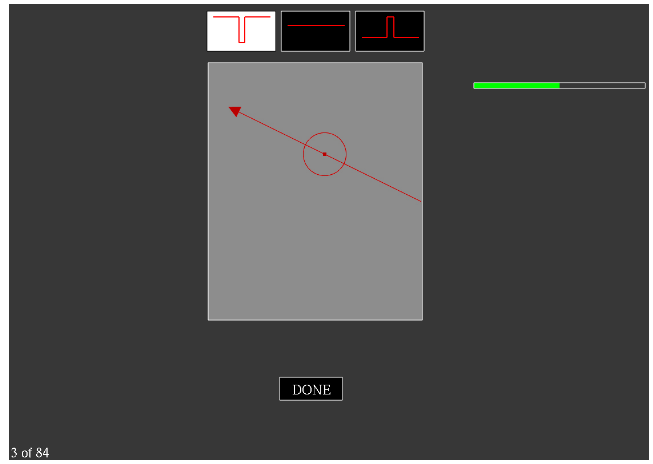

Figure 4. Screenshot of the graphical interface from the experiment. A subject has marked position and orientation of a haptic glyph by positioning the circle and arrow in the main box and has selected an intensity profile from the three icons above. A timer bar on the top right counts down glyph traces.

6.2 Materials

We used the 30-tactor palm-sized array described in Section 3. Other equipment included a standard PC, a mouse as an input device, and liquid-filled headphones rated for 29db attenuation. The PC provided instructions, gathered responses via mouse input, and controlled the array. Subjects wore the headphones for the duration of the experiment.

6.3 Participants

35 subjects participated in the experiment. 34 of them were university students, and one was a recent graduate. 32 were male, and median subject age was 24. Subjects reported no previous experience with 2D vibrotactile arrays, but 17 subjects reported experience with gaming devices such as vibrating gamepads. Subjects were not compensated.

6.4 Procedure

Each subject first reviewed and signed a consent document, completed a background survey, and received a short introduction to the array. Then, the subject donned the noise-reducing headphones and placed the left hand on the array. The subject then received a demonstration of each condition, followed by practice runs that included feedback about accuracy (in total, 10 demonstrations and 20 practice runs).

After a short rest following the practice runs, the subject performed the experiment trials. During these, the subject felt the haptic glyphs and responded by marking parameters on the graphical interface illustrated in Figure 4. The subject specified position by clicking in a rectangle representing the array area, specified orientation by orienting an arrow around the marked position, and selected an intensity profile by clicking one of three buttons illustrating a profile shape. The subject could continue to edit the marked information until satisfied, although the glyph rendering ceased after 10 tracing intervals (10 seconds). In the simplified conditions (POS, DIR, and PROF), the interface was simplified so that the subject only marked the relevant parameters. After each trial was complete, feedback was given to illustrate the correct values graphically. Each subject rested briefly at the experiment's halfway point.

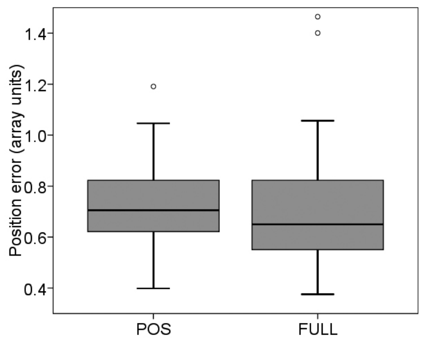

Figure 5. Box-and-whiskers plot showing position error distributions for POS and FULL conditions. Boxes and whiskers show quartile boundaries, excluding outliers shown by points (outliers are NOT omitted in statistical comparisons).

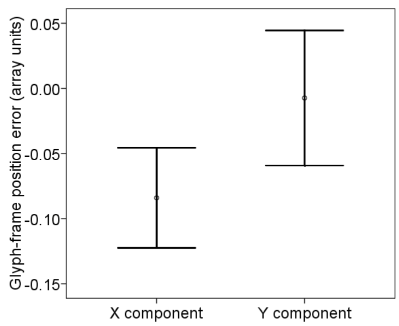

Figure 6. Distribution of Glyph-referenced error components for FULL condition (Mean +/- 1 SE).

6.5 Results and Discussion

We measured the accuracy with which glyph parameters were determined in both the simplified and FULL conditions.

6.5.1 Position Error

We computed a position error for each subject as the mean distance, over 12 trials, between actual glyph position and subject-marked position. The distribution of position error for the 35 subjects is summarized in Figure 5.

Median position error was 0.71 array units for the position-only condition and 0.65 array units for the full glyph condition (one array unit is 18mm). No statistically significant difference was detected between these by Pairwise Wilcoxon Signed Ranked Test for related samples ($Z = -0.393$). Furthermore, visual inspection of the plotted distribution suggests no meaningful difference. Considering the previously-mentioned array limitations and other studies, position accuracy below one array unit is a promising result. Importantly, accuracy appears sustained when users must simultaneously interpret multiple glyph parameters.

It may be possible to reduce position error by accounting for perceptual distortions or related effects. This would involve measuring the relationship between actual and perceived position to derive a mapping that adjusts rendered position accordingly. Although further work is needed for this, we investigated one related effect in our current data. Specifically, for each glyph in our experiment, we computed error components along its principal axes, i.e., an X component along the glyph direction and a Y component perpendicular to it. We computed the mean per-glyph X and Y errors considering only data from the FULL condition (i.e., five data points per glyph, on average, due to the randomized ordering of seven conditions for 35 subjects). Figure 6 summarizes the distribution of these means for the set of all glyphs. The plot shows a mean X error component significantly below 0, showing that subjects tend to judge a glyph trace center to occur earlier in the trace than the true center. This suggests that some error can be removed by shifting the glyph rendering along its X axis. Thus, we demonstrate an effect that should be considered for intensity profile design, in case the timing features in profiles such as DIP and RISE influence perception of position.

6.5.2 Direction Error

We computed direction error for each subject as the mean, over 12 trials, of the absolute value of the angle between actual glyph direction and subject-marked direction. The distribution of direction error for the 35 subjects is summarized in Figure 7.

Median direction error was 12.8° for the direction-only condition and 21.1° for the full glyph condition. The difference was found to be statistically significant by Pairwise Wilcoxon Signed Ranked Test for related samples ($Z = -3.145$, $p = 0.002$).

Data revealed that large errors were close to 180°, with essentially no errors near 90°. Thus, subjects understood slope but sometimes flipped direction. Most subjects made no flips for CONSTANT and RISE profiles (median flip count of 0). Flips occurred most frequently with the DIP profile, for which some subjects made multiple flips (median flip count 1). We believe the error occurs in part because the DIP profile can make the line appear as two points, one being perceived as the glyph beginning and the other as the glyph end, thus requiring a user to track beginning and end. This proved too difficult for some users in the full glyph condition, possibly because they were focusing on other parameters in addition to direction. Possible remedies include using asymmetric profiles, pauses between traces, or other changes in timing.

For an analysis neglecting direction flips, we computed an adjusted direction error as the smaller angle between the marked and actual direction lines (in effect subtracting 180° from errors above 90°). Resulting adjusted direction error was 12.8° for the direction-only condition and 14.0° for the full glyph condition, with no statistically significant difference remaining ($Z = -1.736$). Results suggest that understanding of direction is good except for the direction flips, which may be remedied as suggested and may not be relevant for some applications.

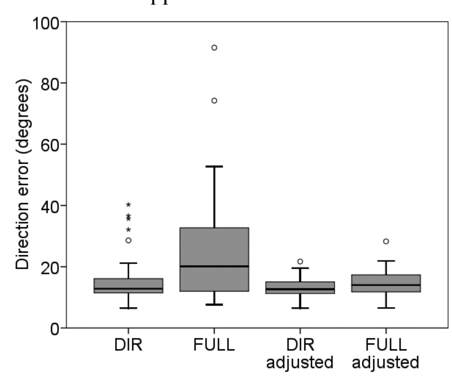

Figure 7. Box-and-whiskers plot showing direction error distributions for DIR and FULL conditions both before and after adjusting for direction flips.

6.5.3 Profile Error

We computed profile error for each subject as the total number of incorrect profile identifications during 12 trials. The distribution of profile error for the 35 subjects is summarized in Figure 8.

Median profile error was 0 for the profile-only condition and was 3 (25%) for the full glyph condition. We found this difference to be statistically significant ($Z = -4.394$, $p = 0.000$). In the full glyph condition, the profile was rendered on a line glyph rather than a point. To render a line glyph, multiple motors are active simultaneously as the trace moves across the array. Considering the limitations discussed in Section 3, we expect that this results in perceivable intensity changes during glyph rendering even in the case of a constant profile. More generally, this is a type of noise that complicates profile identifications. This is the most likely reason for the increase in profile errors in the full glyph condition, besides the increased complexity of the task that required users to also focus on other glyph parameters.

This reasoning suggests that strong differences between profiles are needed to help users easily identify them on arrays such as ours. Some improvement may be possible with different tracing speeds or careful tuning of μ and γ to minimize perceivable changes for constant profiles. However, we expect the best results will come from future tactor technology that supports precise frequency and amplitude control.

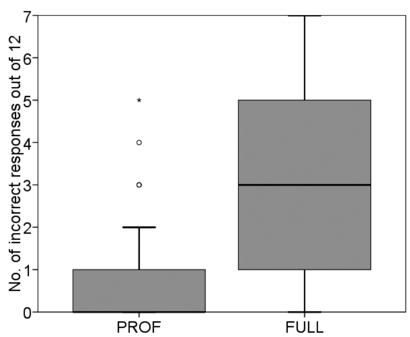

Figure 8. Box-and-whiskers plot showing profile error distributions.

6.5.4 Hint Conditions

In each of three hint conditions mentioned in Section 6.1, full glyphs were rendered as in the FULL condition, but the experiment interface illustrated correct answers (hints) for two of the three parameters. For example, in one of these conditions, correct position and direction were already displayed graphically, but the subject nonetheless had to specify all three parameters as in the FULL condition.

These conditions were omitted from the previous discussion for clarity and because they do not impact any reported conclusions – results were simply consistent with the previous analysis.

In the case of position error, considering hint conditions led to no findings of significance.

Regarding profile error, a hint condition with position and direction hints produced significantly higher profile error than PROF. This is consistent with the discussion about increased difficulty of detecting a profile on a moving point stimulus.

Regarding direction error, a hint condition with position and profile hints produced significantly more direction flips than DIR. The condition showed significantly increased direction error before, but not after, direction flip adjustment. Again, the result matches the previous analysis.

Unsurprisingly, errors tended to be close to 0 for hinted parameters. However, three subjects made multiple profile errors in conditions where correct profile was already highlighted. Furthermore, two of the same subjects had notable, although not very large, direction errors in conditions with direction hints. This illustrates that even presenting the information graphically does not eliminate error for all subjects. On the other hand, position errors were small for all subjects when position was hinted.

7 Conclusion and Future Work

Overall, the experiment shows that the approach is feasible: users can interpret multiple parameters of a 2D tactile glyph, although some precision can be lost when parameters are presented in combination rather than individually (specifically, profile in the case of our array, due in part to device characteristics). However, and more importantly, investigating errors produced insight into their causes that will be useful for the design of haptic cues in future systems. Given the limitations of the low-cost experiment apparatus and the use of novice subjects, results are promising.

We developed a framework for haptic visualization on vibrotactile arrays, along with several specific methods, focusing here on the first general haptic glyph mechanism for 2D vibrotactile arrays. We discussed a potential application to a haptic collaborative map for providing awareness of a remote user's pose and actions in a collaborative system, and also briefly discussed extraction and mapping of dataset features to an array. Future work will focus on improved array technology and on further evaluation of glyph parameters and their communicative power. Further experiments are necessary to uncover guidelines for optimal use of glyph parameters and to evaluate the concepts in a more application-oriented setting.

References

[1] C. W. Borst and A. V. Asutay, "Bi-level and Anti-aliased Rendering Methods for a Low-Resolution 2D Vibrotactile Display," in *WorldHaptics*, 2005, pp. 329-335.

[2] C. W. Borst and V. B. Baiyya, "Enhancing VR-based Visualization with a 2D Vibrotactile Array," in *ACM Virtual Reality Software and Technology (VRST)*, 2007, pp. 183-186.

[3] Y. Yanagida, M. Kakita, R. W. Lindeman, Y. Kume, and N. Tetsutani, "Vibrotactile Letter Reading Using a Low-Resolution Tactor Array," in *12th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 2004, pp. 400-406.

[4] H. Z. Tan, R. Gray, J. J. Young, and R. Traylor, "A Haptic Back Display for Attentional and Directional Cueing," *Haptics-e http://www.haptics-e.org,* vol. 3, June 11, 2003.

[5] H. van Veen and J. van Erp, "Tactile Information Presentation in the Cockpit," in *Haptic Human-Computer Interaction*, Glasgow, UK, 2000, pp. 174-181.

[6] R. W. Lindeman, J. L. Sibert, E. Mendez-Mendez, S. Patil, and D. Phifer, "Effectiveness of Directional Vibrotactile Cuing on a Building-Clearing Task," in *CHI*, 2005, pp. 271-280.

[7] C. W. Borst and G. L. Kinsland, "Visualization and Interpretation of 3-D Geological and Geophysical Data in Heterogeneous Virtual Reality Displays: Examples from the Chicxulub Impact Crater," in *Transactions: Gulf Coast Association of Geological Societies, v. 55*, New Orleans, 2005, pp. 23-34.

[8] T. Delmarcelle and L. Hesselink, "A Unified Framework for Flow Visualization," in *Computer Visualization*, R. S. Gallagher, Ed. Boca Raton: CRC Press, 1994, pp. 129-170.

[9] J. C. Roberts and K. Franklin, "Haptic Glyphs (Hlyphs) - Structured Haptic Objects for Haptic Visualization," in *WorldHaptics*, 2005, pp. 369-374.

[10] N. Osawa, "Tactile Glyphs for Palpation of Relationships," in *Information Visualization*, 2006, pp. 575-584.

[11] M. J. Enriquez and K. E. MacLean, "The Hapticon Editor: A Tool in Support of Haptic Communication Research," in *11th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 2003, pp. 356-362.

[12] A. Chan, K. MacLean, and J. McGrenere, "Learning and Identifying Haptic Icons under Workload," in *WorldHaptics*, 2005, pp. 432-439.

[13] S. Brewster and L. M. Brown, "Tactons: Structured Tactile Messages for Non-visual Information Display," in *Fifth Conference on Australasian User Interface*, 2004, pp. 15-23.