SenseText: Gesture Based Control of Text Visualization

Jason Lewis, Frank Tsonis

OBX Labs, Concordia University
1455 de Maisonneuve Blvd W, Montreal, Canada
jason.lewis@concordia.ca, frank.tsonis@sympatico.ca
http://www.obxlabs.net

Abstract. This paper describes *SenseText*, a wearable device that provides performers with real-time control of text visualization. *SenseText* measures the performer's hand gestures and uses those measurements to determine what visual behaviors get applied to a projected stream of text. The goal is to enhance, expand and augment the performance of audio and visual poetry as well as other works that incorporate visual text.

1 Introduction

Spoken word artists use some degree of hand gesture when they perform. These gestures are just as expressive as the words spoken. Based on this observation, we have created *SenseText* a wearable device that provides performers with real-time control of text visualization.

Adam Kendon refers to gestures accompanying speech as "gesticulation". He states that gestures seldom occur without speech, and that these gestures contain additional information about a speaker's thoughts [1]. He defines gesticulation as "idiosyncratic spontaneous movements of the hands and arms during speech" [2]. *SenseText* builds on Kendon's ideas, by monitoring a performer's speech qualities and hand gestures. This paper will describe the *SenseText* system and how we use it to enable a performer to use hand gestures to control the visualization of text.

2 Related Work

Several interactive media works have focused on monitoring body gestures to control audio and visuals. David Rokeby's *Very Nervous System* [3] vision system tracks body movements including minute hand gestures to control various audio parameters. His system evolves in real-time, providing a performer with a new tool for music composition/performance. Sha Xin Wei's *TGarden* is a multi-user environment where

users improvise body gestures in collaboration with others and by themselves. Costumes equipped with wireless sensing devices track body movements, translating these actions into sound and video [4]. *Body-Brush* is a vision tracking system for creating 3 dimensional graphics using body movement and gestures. Choreographers and dancers can use *Body-Brush* to create real-time visual controlled performances [5]. Where *TGarden* concentrates on creating an interactive environment, *Very Nervous System* and *Body-Brush* focus more on creating and supporting full-body performances. Our motivation for *SenseText* is to design a performance-oriented system that focuses and refines the use of hand gestures rather than full body interactions.

3 System Design

The *SenseText* system consists of four components (see Fig 1): 1) wearable cuffs containing wireless transmitters and accelerometers; 2) a wireless receiver for capturing the movement data; 3) a MAX/MSP [6] patch that analyzes the data; and 4) *TextEngine* [7], a software application for visualizing text in real-time environments.

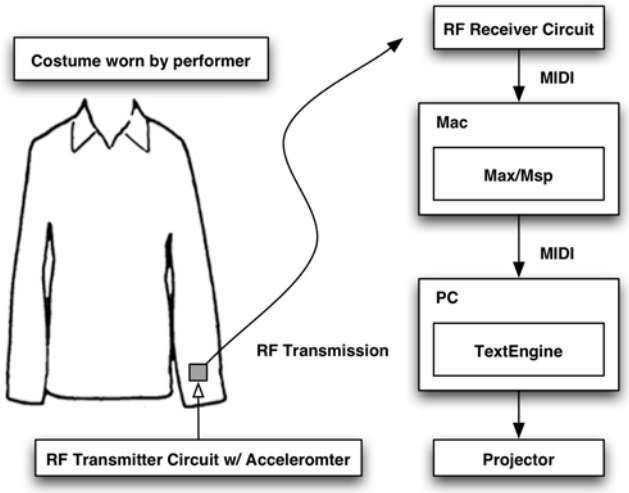


Fig. 1. SenseText System Diagram.

3.1 Wearable Cuffs and Wireless Transmitter Circuit

The transmitter circuit's small size allows for it to fit inside a shirt cuff, making it unobtrusive to the performer (see Fig 2). Since this circuit is wireless, it differs from the

traditional *glove-based devices* used in the past to monitor hand gestures. These devices were seen be some as unnatural to a user, as the cables that connected them to a computer would interfere with a user's performance [8]. By eliminating these cables, our wearable device provides a performer with complete freedom of movement.

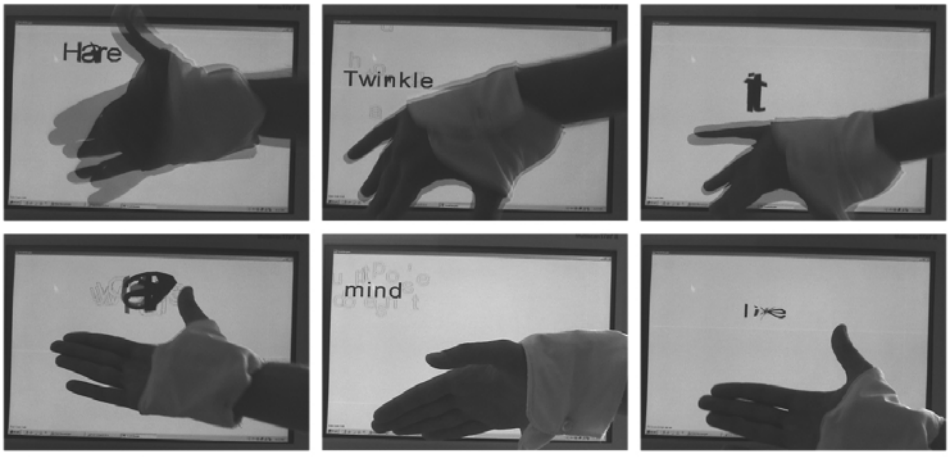


Fig. 2. Shirt Cuff with Embedded Wireless Transmitter Circuit.

The circuit consists of three components: an accelerometer, a PIC chip and a Radio Frequency (RF) transmitter (see Fig 3). The accelerometer measures the wrist movement of a performer, sending these measurements to the receiver circuit using RF. The cuff's shape is similar to that of traditional dress shirts, only the compartment between the two layers of fabric holds the transmitter circuit. Two snap buttons have been sewn onto the shirt and are used to open/close the power supply (9 volt battery) to the circuit.

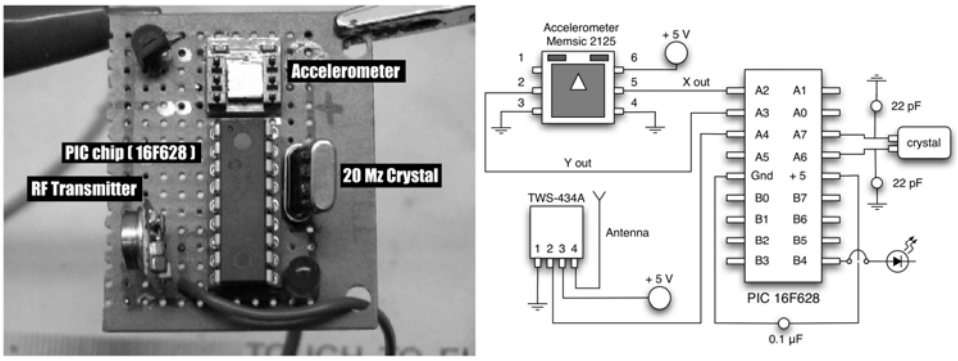


Fig. 3. Wireless Transmitter Circuit.

3.2 Receiver Circuit

The receiver circuit consists of three components: an RF receiver, a Basic Stamp 2 and a MIDI connector (see Fig 4). The circuit receives the accelerometer data and converts this to MIDI data**.** The MIDI connector allows for this circuit to be connected to any MIDI device.

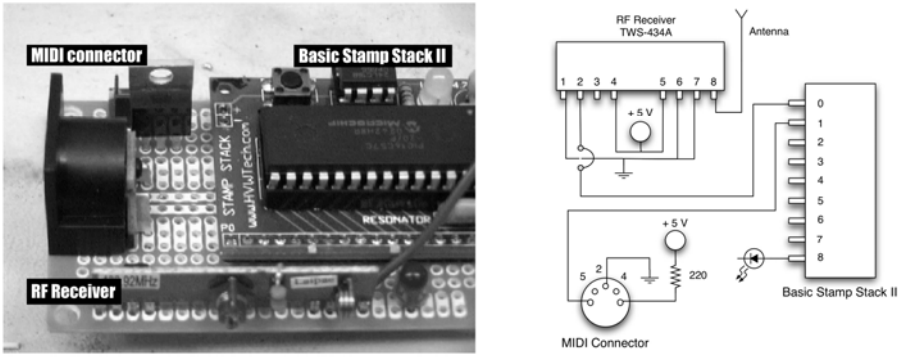


Fig. 4. Receiver Circuit.

3.3 Max/Msp Patch

Max/Msp is a graphical programming language that combines objects together to create structures, know as patches [6]. Our Max/MSP patch analyzes the incoming MIDI data from the receiver circuit, deciphering the extent of the performer's wrist movement (see

Fig. 5). A MIDI value is assigned to each gesture and sent from the MAX patch to the PC. *TextEngine*, running on the PC, assigns these values to the appropriate text behavior.

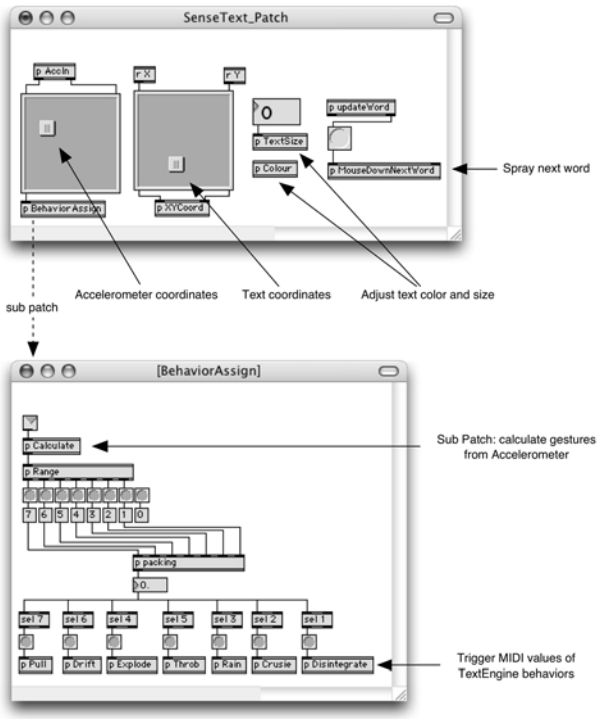


Fig. 5. Max Patch.

3.4 TextEngine

TextEngine provides performers with support for real-time acquisition, recognition, analysis and display of the spoken word. It is based on the *ActiveText*/*NextText* [9,10] text visualization software library. In the *SenseText* scenario, the performer's speech is parsed into text by *TextEngine's* speech recognizer. Simultaneously, the gestures of the performer are fed into *TextEngine* and mapped to specific behaviors. These behaviors are then applied to the text and the combined output is sent to a video device. When *TextEngine* is used with *SenseText*, the location of the user's body determines where the text initially appears onscreen.

4 Mapping Text Behaviors to Hand Gestures

TextEngine supports a number of visual behaviors. These include simple behaviors that change font, size, color, and position. More complex behaviors, such as those that deform the letterforms or interact with the user, are built on the simple behaviors. In *SenseText*, we map vocal qualities to simple behaviors and hand gestures to complex behaviors. This mapping arises out initial tests with a rough prototype, which suggested that vocal qualities were adjusted in an almost unconscious manner, and thus the related behaviors should be engaged automatically. For example, the color of each word is chosen to reflect the pitch of a performer's voice when that word is spoken. Similarly, font size corresponds to the volume intensity for each word spoken.

In contrast, the more complex visual behaviors require concentrated attention. Hand gestures, which provide six control axes (roll, pitch, yaw, as well as horizontal, vertical and depth translation), provide a rich means for controlling such behaviors.

We have grouped a subset of *TextEngine* behaviors into two categories, soft and hard. The behaviors are scaled according to the intensity and of their movement. Soft behaviors are categorized as those having fluid movements and are triggered by minimal hand gestures (see Fig 6).

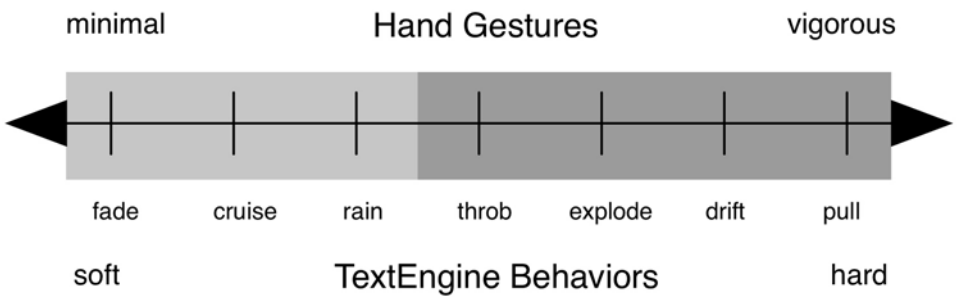


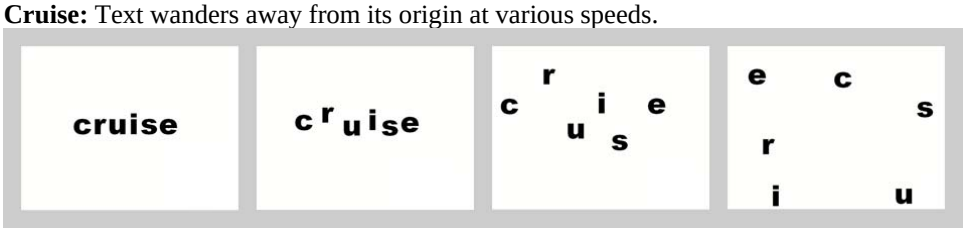
Fig. 6. Mapping Hand Gestures to TextEngine Behaviors.

Examples include *fade*, *cruise* and *rain*. Hard behaviors are those that are more rigid and abstract and are triggered by vigorous hand gestures. Examples include *throb*, *explode*, *drift* and *pull*. Table 1 provides descriptions for these *soft* behaviors and Table 2 provides descriptions for these *hard* behaviors.

Table 1. Descriptions and images of *soft* behaviors.

Fade: Text fades to background color.

Cruise: Text wanders away from its origin at various speeds.



Rain: Text falls from top of screen, as do droplets of rain.

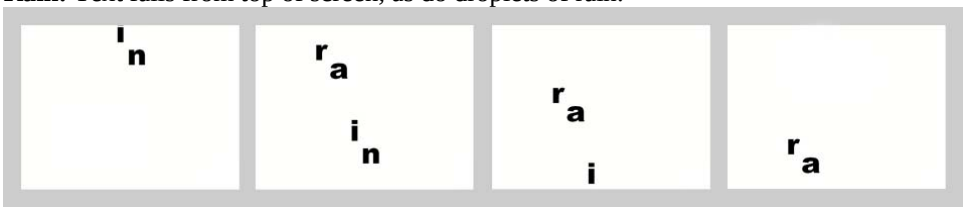
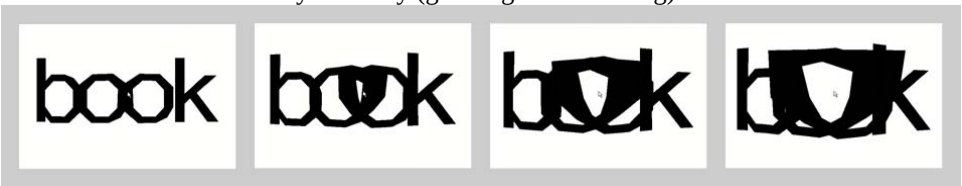
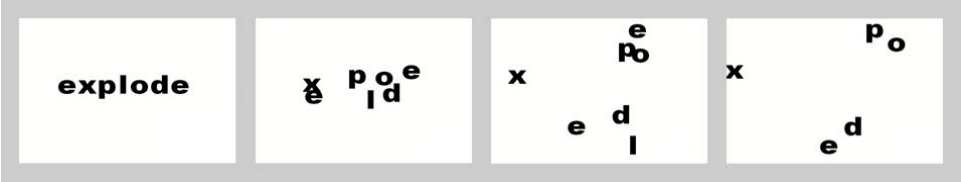


Table 2. Descriptions and images of *hard* behaviors.

Throb: Letterforms beat rhythmically (growing and shrinking).



Explode: Letterforms fly apart in all directions from their origin.



Drift: The outlines of letterforms are deformed based on interaction with mouse.

Pull: The pixels of letterforms are pulled towards the mouse.

5 Conclusion and Future Developments

The current version of *SenseText* provides us with a rich platform for exploring the relationship between gesture and the spoken word. Our informal tests to date have already exposed several interesting characteristics of this relationship, most notably in relationship to conscious/subconscious attention and control. We anticipate implementing a number of technical improvements to the prototype, including:

1. Eliminating the Max/Msp patch by reprogramming the transmitter circuit so that this circuit itself processes the intensity of the wrist movement. This will reduce latency and increase flexibility by reducing the amount of equipment and software required to mount a viable system.
2. Increasing the accuracy of the speech recognition component to provide a better match between the spoken and visualized words. We have recently begun collaborating with ScanSoft's SpeechWorks division to assist us in this effort.
3. Expanding the number and types of behaviors.

The improved system will allow us to field-test *SenseText* with spoken-word performers. We will use those performances to calibrate the mapping of gestures to behaviors, to fine-tune the behaviors themselves, and to design an interface that will give performers the ability to adjust the system themselves. In the process we hope to learn much more about the connection between gesture and the spoken word.

Acknowledgements

The authors would like to thank Alexander Taler, David Bouchard and Bruno Nadeau for their work on the *NextText* library and *TextEngine*. The work described herein was conducted at Obx Laboratories, which is supported with funding from Hexagram: Centre inter-universitaire des arts médiatiques, and the Fonds québécois de la recherche sur la société et la culture.

References

1. “Communicative Gesture”. 2003. McNeill Lab: Center for Gesture and Speech Research. Jan 5, 2004. http://mcneilllab.uchicago.edu/topics/comm.html.

2. Mulder A. 1996. “Hand Gestures for HCI”. Jan 18, 2005. http://www.cs.sfu.ca/people/ResearchStaff/amulder/personal/vmi/HCI-gestures.htm

3. David Rokeby: Very Nervous System, http://homepage.mac.com/davidrokeby/vns.html

4. Sha X.W. “Resistance is Fertile: Gesture and Agency in the Field of Responsive Media”, *Configurations*, Vol 10, Number 3, Baltimore: Johns Hopkins University Press, 2003, pp. 439-472.

5. Ip, H., Young H., Tang A. "Body Brush: A Body-driven Interface for Visual Aesthetics", *ACM Multimedia* 2002.

6. Max/Msp software, http://www.cycling74.com/products/maxmsp.html

7. TextEngine: dynamic text software, http://obx.hybrid.concordia.ca/research/nexttext/textengine/research_textengine.htm

8. Pavlovic, V., Sharma R., Huang T. “Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review", *IEEE Transactions on Pattern Analysis and Machine Intelligence 19(7)*: 677-695.

9. Lewis, J. and Alex Weyers. “ActiveText: A method for creating interactive and dynamic texts”, *Proceedings of the 12th annual ACM Symposium on User Interface Software and Technology, Asheville, North Carolina. New York: ACM Press, 1999.*

10. NextText: text visualization software library, http://obx.hybrid.concordia.ca/research/nexttext/research_nextext.htm