New Interfaces for Textual Expression

Adam Parrish
NYU ITP
455 Macdonough St. Brooklyn, NY 11233
aparrish@nyu.edu

ABSTRACT
New Interfaces for Textual Expression (NITE) is a series of devices intended to create and manipulate text. Analogous to contemporary interfaces for musical composition and performance, New Interfaces for Textual Expression are intuitive but not literal: they map gestures not to characters (as with conventional writing devices, such as the keyboard and the pen), but to broader manipulations of language and layout. The devices suggest new syntaxes for composing, reading, and performing text.

INTRODUCTION

Digital media has made it easy to manipulate text in sophisticated ways, but the primary interface between the writer and the text is still relatively simple: a keyboard, which maps a gesture (pressing a key) to a character (which appears on the screen). The interfaces described in this paper seek to expand the vocabulary of interfaces for generating text, using methodologies inspired by contemporary practices in digital musical instrument design. These interfaces demonstrate a new way of conceptualizing the relationship between the writer, the interface for writing, and the text.

BACKGROUND: TEXTUAL INTERFACES

Every text is a transcription: the result of transmuting actions in the real world into a text. The text you're reading right now, for example, is—at the most basic, physical level—a transcription of my having typed keys that created it. On a more abstract level, it's a transcription of my having written a document for a particular purpose (fulfilling a requirement for academic advancement), in a particular style, for a particular audience, and with a particular tool (a computer). A text results from any number of these physical, stylistic, and algorithmic decisions.

Because a text bears the traces of these decisions, it is possible to reconstruct from the text the actions that were the source of its transcription. As linguists Greg Urban and Michael Silverstein state in their introduction to *Natural Histories of Discourse*: "The text-artifact does indeed have a physical-temporal structure, precisely because it was originally laid down, or sedimented, in the course of a social process, unfolding in real time. . . . We seek the durational event of the laying-down process, insofar as traces of the original co(n)text in which a discourse fragment was configured are available to us." [18]

This is especially true of poetic texts. "A poem," according to poet John Ciardi, "is not simply something printed on the page. A poem is an event. . . . Reading a poem is an act of participation in the poem. By participating, the reader not only makes the performance whole, but makes it, in one essential sense, uniquely his" [5].

My methodology proposes a *mapping* between the poetic process (the "event") and the text that emerges from it. I contend that the reader participates in the text by "filling in the gap" between the process and the artifact.

Beyond the literal

Under this model, it becomes possible to talk about "interfaces" for creating text—interfaces that define the mapping between the writer's process and the text that results. The most familiar interface that meets this definition is the computer keyboard, which maps a physical gesture (pressing a key) to a literal, electronic outcome (a glyph appearing on the screen). This literal mapping between gesture and sign is the most obvious form of a textual interface, but it's not the end of the story.

A similar tendency toward literal interfaces occurs in electronic musical interface design. Joel Ryan of STEIM (the Studio for Electro-Instrumental Music) relates: "The impulse of many composers when they first use the computer is to. . . develop languages to describe sounds as accurately as possible" [15]—to develop, in other words, a literal interface for music. But the possibilities for expressive interfaces are not thereby exhausted. "Conveniently," Ryan continues, "the computer is just as well suited to the *invention of a process to generate music* as it is to the concept of a singing manuscript" (my emphasis).

The New Interfaces for Musical Expression movement is characterized by its constant invention of new processes—processes intended "to deny the habitual or the hackneyed by developing techniques to restrain or condition the immediate process of choice," as Ryan describes it. The approach I take here with textual interfaces is no different.

Models and mappings

In *The Design of Everyday Things*, Donald Norman provides a set of criteria for designing effective interfaces. They must, among other things, provide the following:

- a consistent *conceptual model* of how the system and its interface work;
- *natural mappings* between actions and results, to promote immediate understanding; and
- *feedback*, or clear information about the state of the system. [12]

The difficulty inherent to designing new, non-literal interfaces for text—as with designing new interfaces for music—is that "new processes" may not always have an intuitive mapping, and the conceptual model inherent to it may be abstract, abstruse, and unfamiliar. Because of this tendency toward the abstract, Joel Ryan explains, "[e]ach link between the performer and computer has to be invented before anything can be played."

This difficulty, however, opens a door of opportunity. "The physicality of the performance interface. . . stimulates the imagination," continues Ryan, "and enables the elaboration of the model using spatial and physical metaphors. The image with which the artists works to realize his or her idea is no longer a phantom." For poetic texts, the possibilities here are enormous. The physical interface not only opens up new modes of creating text, but also introduces new kinds of literacy. A broader range of processes can be transcribed in text, and readers have a broader range of knowledge to help "fill in the gap" between the process and the transcription.

The devices outlined in this paper serve as examples of the new literacies made possible by mediating the creation of text through novel physical interfaces. The processes they render visible are algorithmic, gestural, or synaesthetic in nature.

CONTEXT

A number of writers, poets, and artists have created work that falls into, or at least suggests, the framework suggested above.

Cut-ups and Constraints

Among the earliest examples of what I would term "textual interfaces" are cut-up techniques—famously first suggested by Tristan Tzara (in *How to make a Dadaist Poem*), and later honed by William S. Burroughs. This technique, intended to "produce the accident of spontaneity," consists of a simple algorithm: "Take a page. Like this page. Now cut [with scissors] down the middle and cross the middle. . . ." [4] The resulting text wears proudly on its sleeve the physical process by which it was achieved. Nick Montfort explains that the technique, notably, was not a means of "[g]enerating texts directly for readers," but instead a "an intermediate step in composition" [9]—an interface, in other words, between the writer and the text.

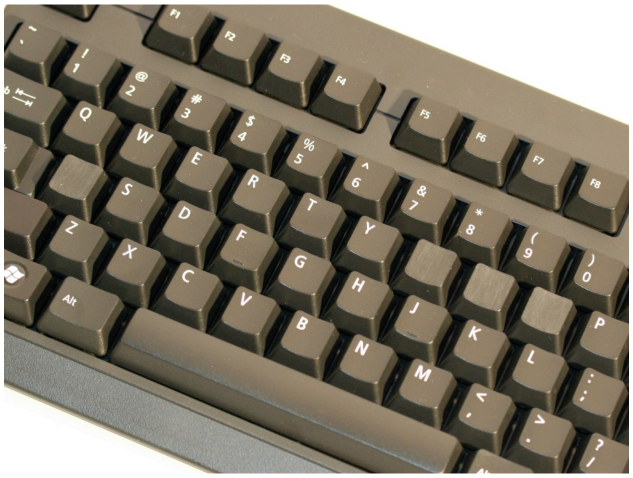


Figure 1. The Oulipo Keyboard.

Physical interfaces and performance

Contemporary poets and media artists are beginning to incorporate physical interfaces into their practice and performances. One example is Aya Karpinska's *Nobody knows but you*, an poetry installation and performance piece that uses a computer game controller to change the configuration of a screen-based text in realtime, in tandem with music and a spoken word performance. MIDIPoet, developed by Eugenio Tisselli Vélez, is a generalized software tool for composing interactive texts that "respond to external impulses, such as MIDI messages" [19]; this interoperability with MIDI allows any number of existing physical interfaces (e.g., music controllers) to affect the shape of the generated text.

METHODOLOGY: THE INTERFACES

The following five interfaces are intended to demonstrate what a "new interface for textual expression" might look like, and establish a basic taxonomy that will guide future investigations in the field. They were created over the course of my final semester at New York University's Interactive Telecommunications Program.

Oulipo Keyboard

Our first New Interface for Textual Expression, the Oulipo Keyboard (depicted in figure 1), is a commonplace computer keyboard—with a twist. A number keys on the keyboard have been rendered inactive—namely, every vowel key except *e*.

The Oulipo Keyboard is an example of an interface that NIME theorists John Bowers and Phil Archer might call an "infra-instrument": an existing instrument that has been broken or restricted. [3] It's the simplest implementation of a New Interface for Textual Expression. The writer must make tactical adjustments to their writing practices in order to compensate for the unexpected affordances of the interface. The resulting text bears the traces of the interface through which it was realized.

The name of the interface refers, of course, to the *Ouvroir*

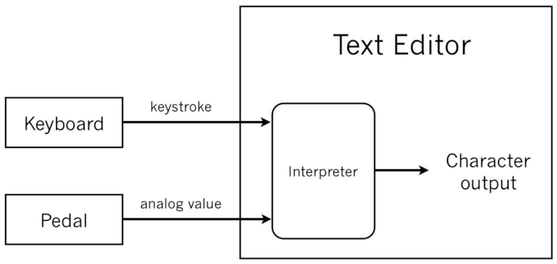


Figure 2. Entropic Text Editor: System Diagram.

de littérature potentielle, or *Oulipo*, an experimental writing collective originating in the 1960s which sought to (in the words of member Raymond Queneau) “elaborate a whole arsenal in which the poet may pick and choose, whenever he wishes to escape from that which is called inspiration.” [10] One weapon in the arsenal was the *lipogram*—writing a text using only a subset of the alphabet. George Perec’s *La Disparition* [14], for example, was written entirely without the letter *e*. The Oulipo Keyboard makes this a physical (rather than strictly ruled-based) constraint. (The particular constraints of the Oulipo Keyboard, however, owe more to Christian Bök’s *Eunoia* [2], in which each chapter contains words using only one of the letters that designate English vowels.)

The Entropic Text Editor

The Entropic Text Editor is a tool for creating concrete, nonsensical poetry. It consists of a keyboard, a repurposed analog expression pedal (originally intended for use with an electronic musical keyboard), and a text editor. The text editor is programmed to modulate the text according to the position of the pedal: as the pedal is pressed down further, more randomness is applied to the character being typed. Figure 2 illustrates the relationship between the components.

Three kinds of “randomness” are applied to the text in response to the pedal’s position. The first is the letter’s identity—i.e., whether the letter comes out as itself, or a letter nearby in the alphabet. The kerning of the characters and the weight of the typeface are also affected.

Figure 3 shows an example of a text created that I created with the interface. I gradually pushed the pedal down until it reached its full extent at line 7. Then I gradually moved it back to the “normal” position by the last line of the text.

The Entropic Text Editor is another example of a simple New Interface for Textual Expression, what might be termed an “augmented” textual interface (by analogy with augmented musical instruments, such as Dan Overholt’s Overtone Violin [13]). The hands are free to engage in the familiar act of typing, but another channel of information is added, which modifies how the typing works. The artifacts that result from the Entropic Text Editor incorporate not just the literal content of the text, but also a history of the writer’s gestures.

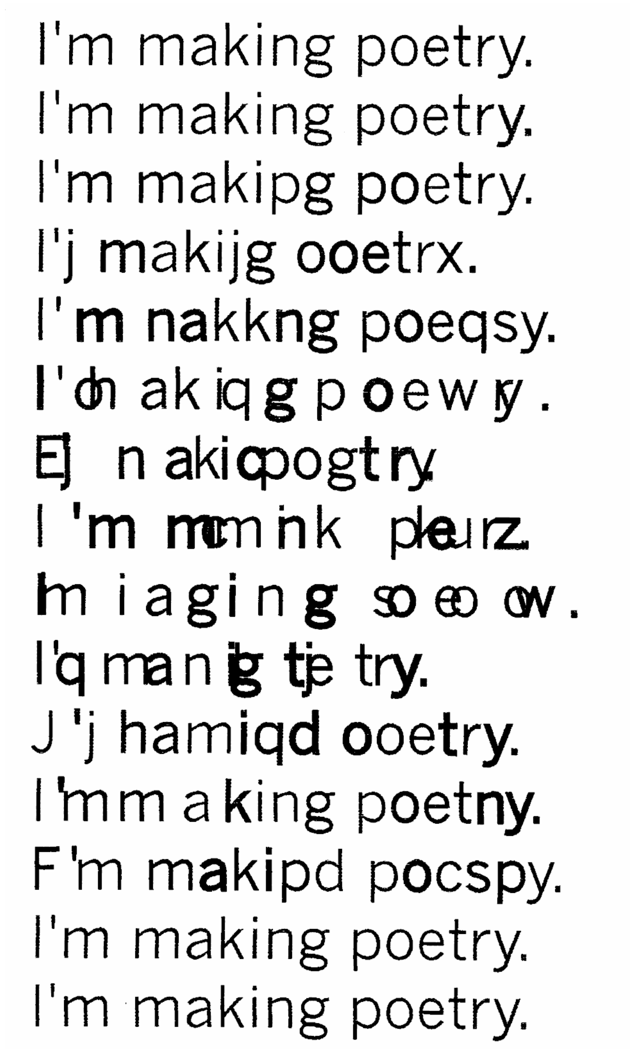

I'm making poetry.
I'm making poetry.
I'm makipg poetry.
I'j makijg ooetrx.
I' m nakkng poeqsy.
I'dn ak iqg p oew ry .
E] n akicpogt ry
I 'm mcm ink pleurz.
Im i aging so eo cw .
I'q man ig tje try.
J 'j hamiqd ooetry.
I 'mm a king poetny.
F'm makipd pocspy.
I'm making poetry.
I'm making poetry.

Figure 3. Entropic Text Editor: Sample text.

The key idea behind the Entropic Text Editor is that it modulates text (an inherently digital technology) with an analog input. Many aspects of the text could be modulated according to this input—for example, some of my early prototypes experimented with font size and lexical replacement. However, I wanted to show how a physical interface could provide a reader purchase with even the most concrete and (seemingly) asemic poetry.

Specifically, this interface was inspired the following excerpt from David Melnick's *PCOET*, which appeared in the first issue of *L=A=N=G=U=A=G=E*: [8]

thoeisu

thoiea

akcorn woi cirtus locqvump

icgja

cvmwoflux

epaosieusl
    ~~cirtus locquvmp~~

a nex macheisoa

In the accompanying explanatory text, Melnick expresses "doubt that any statement will mediate between *PCOET* and its audience," taking "new delight in not needing to explain [the work]." Of course, there *is* a means for readers to understand the piece—by reconstructing the physical act of writing, the position of the fingers over the keyboard, the gesture of typing a key. The Entropic Text Editor is intended as a tool to make texts that explore this same aesthetic, while bringing the process of writing the text into the foreground.

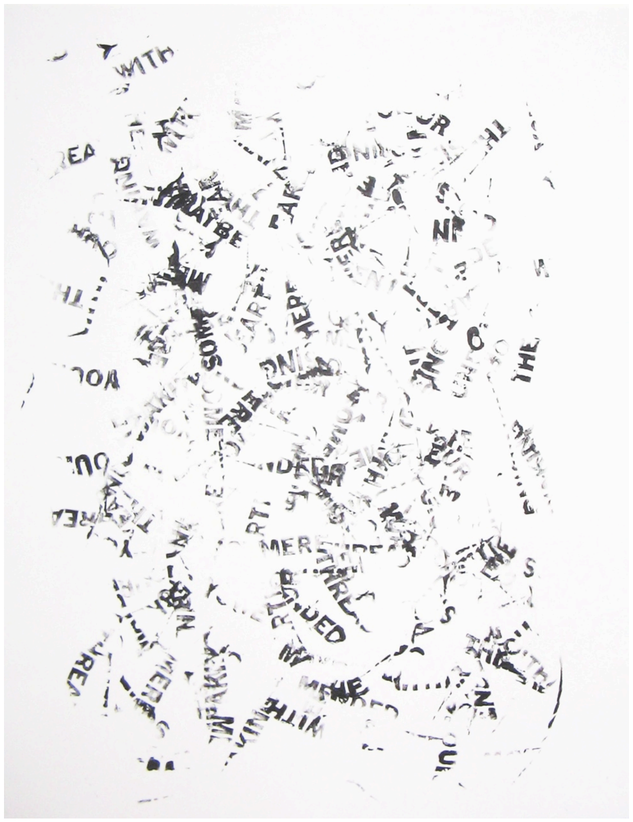


Figure 4. Poem Sphere: Sample text.

Poem Sphere

The Poem Sphere is an example of a non-digital textual interface: purely physical, it operates without assistance from a computer. It consists of a four-pound medicine ball with linocut words glued to the outer surface. When you ink up the ball and roll it across a surface, it creates a composition: fragments of words, spread across the page.

The Poem Sphere provides a new way of making text that is expressive but not literal. Using this tool, the writer's choices about how the text unfolds involve tactile and choreographic decisions, rather than decisions about letters, words, and sentences. The goal was to create concrete poetry, in the vein of Apollinaire's *Caligrammes*, the French Letterists [17], and bpNichol [20], while emphasizing the physical process that brought the text into being.

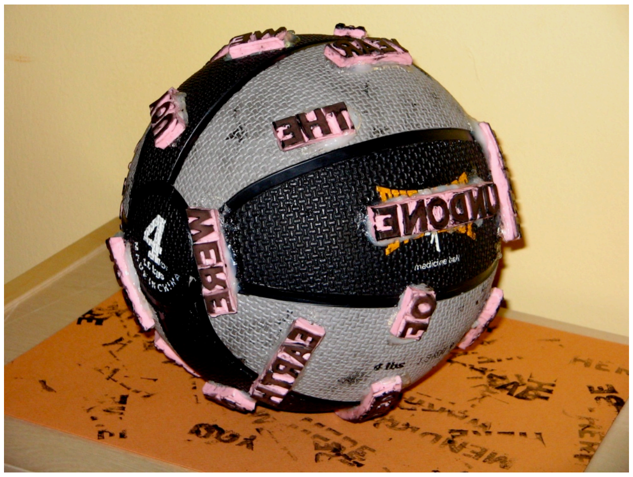


Figure 5. The Poem Sphere.

Markov Live!

Markov Live! is a physical interface for an algorithm: namely, a Markov chain. A photo of the interface is provided in figure 6, and the functions of the interface are shown in figure 7.

The software portion of Markov Live! works like this.

1. A source text is loaded into the program; the text is broken up into tokens (words).

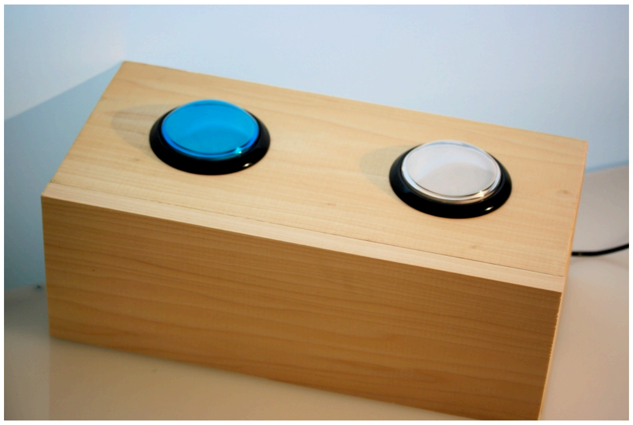


Figure 6. Markov Live!

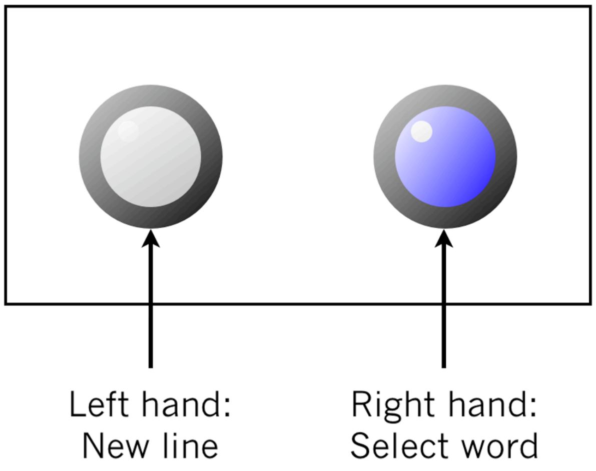
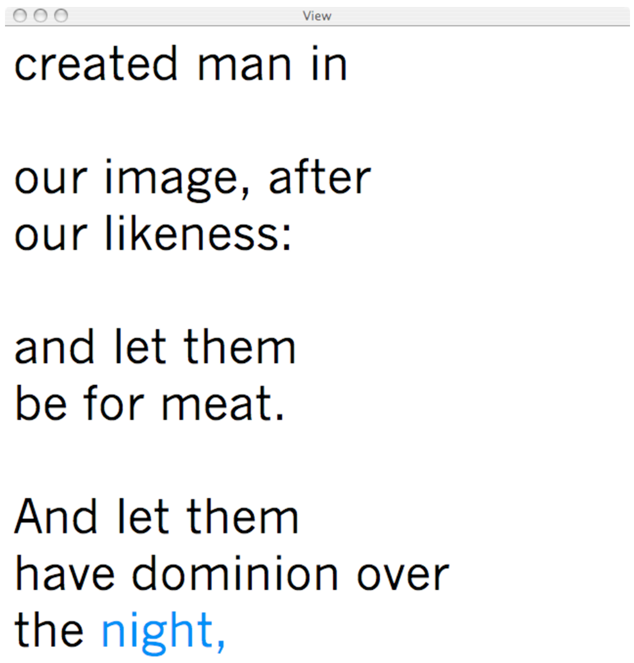


Figure 7. Markov Live! Interface diagram.



created man in

our image, after
our likeness:

and let them
be for meat.

And let them
have dominion over
the night,

Figure 8. Markov Live! screenshot.

2. The writer/player pushes the "Select word" button to start the program; two sequential tokens, selected at random from the text, are printed to the screen.

3. The text is searched for all occurrences of the previous two tokens; the software builds a list containing all tokens that *follow* those two tokens, wherever they occur in the source text. Those tokens are then flashed to the screen one-by-one, in place.

4. The writer/player pushes the "Select word" button when the desired alternative is being displayed. Step 3 is then repeated, but using the most recently generated word along with the word before it.

5. At any time, the writer/player can push the "New line" button to advance the text one line.

A screenshot of the program in action is provided in figure 8, using the first chapter of Genesis from the King James Bible as a source text. The last word in the screenshot, *night*, was one of several alternatives being flashed to the screen, sequentially. (The others were *cattle*, *fowl*, *day*, and *fish*.)

Markov Chains are a well-known tool for generating poetic texts, being first used for that purpose no later than Hugh Kenner and Joseph O'Rourke's Travesty generator, released in 1984. [6] Gnoetry, "an on-going experiment in human/computer collaborative poetry composition," [16] uses a similar algorithm. The Markov Live! interface and Gnoetry are similar in that they both use human input to "guide" the algorithmic process. Markov Live! introduces a temporal dimension—decisions about the text are made in real time—and an element of performance: the writer/player must time

Figure 10. Ben Leduc-Mills plays Beat Poetry.

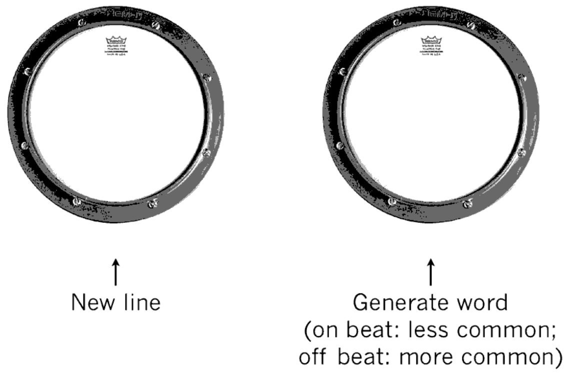


Figure 11. Beat Poetry: Interface diagram.

their button presses carefully in order to achieve the desired text.

Beat Poetry

Beat Poetry is an example of a synaesthetic textual interface: it maps the act of playing a musical instrument (specifically, drums) to the act of writing text. The interface consists of two electronic drum trigger pads, two drum sticks, a computer screen, and an audible indication of the beat.

Here's how it works: One drum pad is designated as the *generator*, and hitting this pad will produce a word. The other drum pad creates a new line when hit. At the beginning of the performance, the software reads in a chosen source text, and ranks the words in the text according to their frequency; the top ten percent are labeled "common," and the bottom ninety percent are labeled "rare." Hitting the generator pad on *on the beat* will produce a random word from the "rare" set, while hitting the pad *off the beat* will produce a random word from the "common" set.

Words from the "common" set tend to be shorter, monosyllabic words from the grammatical classes of article, pronoun

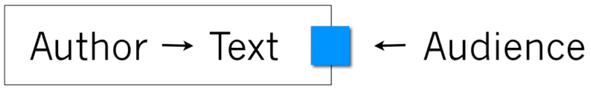


Figure 12. Traditional model of digital poetics. (The small box represents the interface.)

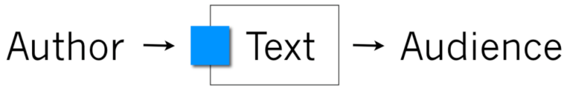


Figure 13. Model of relationship between author, text, interface, and reader, proposed by NITE.

and preposition (e.g., *the*, *he*, *to*, *in*, etc.), while those in the "rare" class tend to be heavier, polysyllabic nouns and verbs (e.g., *refrained*, *instinct*, *dissimulation*). Mapping rare words to the beat and common words off the beat tends to reproduce the natural lexical rhythm of English text (disregarding semantics).

A sample text generated from a Beat Poetry session is given in figure 9.[1]

EVALUATION

NITE versus contemporary approaches in digital poetics

NITE stands in contrast to some contemporary approaches in digital poetics, in its emphasis on creating new ways of *writing* text, rather than new ways of interacting with it. The following passage, found in an essay by poet and critic Talan Memmott, contains language and metaphors that will help elaborate on this point. "Digital poetry presents an expanded field of textuality that moves writing beyond the word to include visual and sound media . . . . Its performance or poetic emergence requires *the participation of a user or operator* to initiate the computational process *encoded by its author*. Like a musician playing an instrument, a user could be said to play an application." [11] (My emphasis.)

Although Memmott's passage ends up in the same neighborhood as NITE—asserting that a text can be "played" like a musical instrument—the model he suggests for digital poetics is very different. That model looks like this: in digital poetry, the author creates a work (an *application*), then *encodes* it behind an interface; the user's job (the term *user* is especially important, opposed here to *author*) is to *decode* the work obscured by the interface. The interface and the text generated by the interface are considered to be an inseparable unit. Digital poems, in this model, are standalone machines—*ergodic texts*, to use Espen Aarseth's terminology. [1] This model is illustrated in figure 12.

The model proposed by New Interfaces for Textual Expressi-

[1]This text serves not just as a transcript of the textual performance, but of the *musical* performance as well. This fact has some interesting implications. In my utopian textual world, a bootleg of a musical performance won't be an audio recording—it'll be a printout of the text the musical performance created.

steadily that because every
cautious

first to

heard mind
bed mouse
this imagine
them madman

shone saw eye absent

my into old what tone such
night by whole grew

whos upon you wild with
pushing

destroyed tin eye upon own

of with eye again is louder
manner
stood waited when stood

black hark bed but arose have
hastily so did

film with eye vulture him by

Figure 9. Beat Poetry: Sample transcript. The source text was Edgar Allan Poe's *The Tell-tale Heart*.

on (pictured in figure 13), on the other hand, places the interface between the text and the author. The interface mediates the text's creation, but is not synonymous with it. The "user" of a New Interface for Textual Expression *is* the author, and the role of the audience is not to decode the interface, but to witness (whether during or after the fact) the author's act of creating the text. Although the texts created with these interfaces require unique methods of reading in order to be fully understood, they need not be screen-based, multimedia, or interactive.

Future directions

Having established a basic taxonomy of textual interfaces, a number of future directions present themselves. Some possibilities include physical interfaces for other common text generation algorithms (such as a context-free grammar), or for rearranging other formal units of text (sentences, paragraphs, verses, acts). Experimentation with the form of the interface—haptic, wearable, collaborative—also seems like a fruitful area of research.

The possibility that interests me the most, however, is public performance of text. NITE's new configuration of author, interface and audience suggests that the creation of text is itself a kind of performance. With the right interface, the writer—essentially a performer of text-creation—could become a (public) performer on the same level as the dancer, actor, and musician.

REFERENCES

1. Aarseth, E. *Cybertext.* John Hopkins University Press, Baltimore, MD, USA, 1997.
2. Bök, C. *Eunoia.* Coach House Books, Toronto, Canada, 2001.
3. Bowers, J. and Archer, P. Not hyper, not meta, not cyber, but infra-instruments. In *Proceedings of NIME 05*, Vancouver, Canada, 2005.
4. Burroughs, W. and Gysin, B. *The third mind.* Viking, New York, USA, 1978, 29-33.
5. Ciardi, J. and Williams, M. *How does a poem mean?* Houghton Mifflin, Boston, MA, USA, 1975.
6. Hartman, C. *Virtual Muse: Experiments in Computer Poetry*. Wesleyan University Press, Hanover, NH, USA, 1996, 54.
7. Karpinska, A. *Nobody knows but you.* Available at: http://technekai.com/nobodyknows/
8. Melnick, D. A short word on my work. *L=A=N=G=U=A=G=E*, 1:12-13, 1978.
9. Montfort, N. The Cut-Up Method of Brion Gysin [Introduction]. In *The new media reader*, Montfort, N. and Wardrip-Fruin, N. eds. MIT Press, Cambridge, MA, USA, 2003, 89.
10. Lescure, J. Brief History of the Oulipo. In *The new media reader*, 172.
11. Memmott, T. Beyond taxonomy: digital poetics and the problem of reading. In *New media poetics*, Morris, A. and Swiss, T. eds. MIT Press, Cambridge, MA, USA, 2006, 293-306.
12. Norman, D. *The design of everyday things.* Basic Books, New York, USA, 1988.
13. Overholt, D. The overtone violin. In *Proceedings of NIME 05*, Vancouver, Canada, 2005.
14. Perec, G. *La disparition.* Gallimard, Paris, France, 1989.
15. Ryan, J. Some remarks on musical instrument design at STEIM. *Contemporary Music Review*, 6(1):3–17, 1991.
16. Trowbridge, J. and Elshtain, E. *Gnoetry*. Availabile at: http://www.beardofbees.com/gnoetry.html
17. UbuWeb. French Letterists 1940–1970s. Available at: http://www.ubu.com/historical/letterists/index.html
18. Urban, G. and Silverstein, M. *Natural histories of discourse*. University of Chicago, Chicago, IL, USA, 1996.
19. Vélez, E. *MIDIPoet*. Available at: http://www.motorhueso.net/midipeng/
20. Young, K. The visual poetry of bpNichol; a brief sketch. Available at: http://www.thing.net/ grist/l&d/bpnichol/ky-e-bp.htm