VENTRILOQUISM IN CODE-BASED PARTICIPATORY ARTWORKS

VICTORIA BRADBURY

CRUMB, University of Sunderland
United Kingdom

vebradbury@gmail.com

Keywords: Code, Speech, Ventriloquism, Participation, Artist-Programmer, Translation, Voice, Performativity

This paper uses an analogy of ventriloquism to reflect on the roles of code, coding artist, and visitor in participatory new media artworks. Ways in which theorists and practitioners have viewed code and speech are considered while two of the author's artworks, *Toast and Ventriloquisms for Fun and Profit*, are used as case studies. Here, the projects are described and insights that emerged from their implementation are proposed as results. Throughout the text, ventriloquism diagrams are used to illustrate possibilities for directional transmission of speech that occur in the artworks being discussed.

1. INTRODUCTION

DUM: *Hey, what's goin' on here? What's the idea of wearing the white coat?*
VENT: *Well, you see, you are the patient, and I am the dentist.*
DUM: *(Calmly) Oh, I see … (Suddenly leaps up) WHATTT? (Winchell 1954)*

A ventriloquism analogy can be used as a model to consider the roles of coding artist, code, and participant in a new media artwork. A general model for a code-based participatory artwork looks like this:

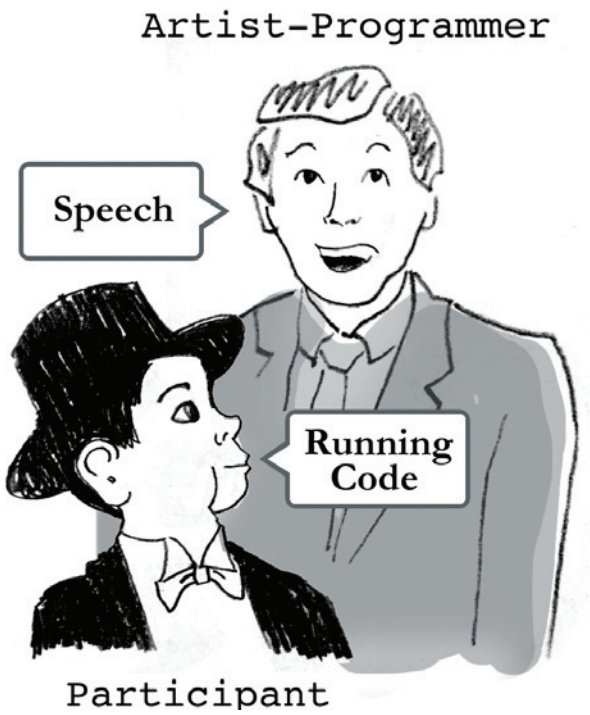

Fig. 1 General ventriloquism diagram for code-based participatory artworks

Computer code is written, compiles, and then runs through an end user. In the act of ventriloquism, the voice is thrown so as to appear to be coming from somewhere other than the original source. A study of ventriloquism in relation to new media art considers the origin of voice (code), the phenomenon of one entity speaking through another, and potentialities of control in computational systems. This paper examines ways in which new media artists and theorists have previously discussed ventriloquism while using the analogy of ventriloquism to re-consider the roles of code, coding artist, and visitor/participant in two of my new media artworks: *Toast*, which explores translation and *Ventriloquisms for Fun and*

Profit, a performance with audience involvement. Ventriloquism diagrams are used to simplify complex scenarios. They are intended as a tool to discover the types of entities that emerge when code is run and interpreted.

A ventriloquist is responsible for both sides of a conversation while dummies offer the illusion of autonomy (Clair 2007). A dummy appears to be a sovereign being while the audience knows that he is an extension of the ventriloquist. Thus, a doubling occurs and a loop is established:

```
{Ventriloquist → Dummy → Ventriloquist →}
is equivalent to (==)
{Ventriloquist → Ventriloquist → Ventriloquist →}
```

Alexander Galloway (2004) writes about the importance of language to communication. He defines language as shared protocol. For two nodes on a distributed network to communicate, they must speak the same language. Galloway states that "Code is the only language that is executable" and "[code] is a machine for converting meaning into action."

In *Interaction/Participation: Disembodied Performance in New Media Art*, *Dead History*, *Live Art?* Beryl Graham states:

> *Conversation is a highly elaborate skill involving exchange, evolution, creativity, interpretation, empathy and ambiguous language. Computer logic may just about be able to manage the first two factors, but beyond that it needs firm rules and predictable structures. (Graham, 2007)*

Languages, both coded and spoken, provide the power to communicate. Code is a language that acts, but as Graham states, it is not capable of carrying out complex conversation. In our technology-dependent culture, code mediates and controls conversation between humans. Just as audiences watching a ventriloquism performance ignore the objecthood of the dummy in order to be entertained, users similarly ignore the coded infrastructure beneath computational devices. This reinforces the power of the person, company, or government that writes or owns the code, while those using the technology are locked out of adjusting it or accessing its inner-workings.[1]

1 The experience of technology, for the majority of people, does not include the creative act of writing code, but only the consumption of a final interface with no entry into its inner workings. This is especially true in the case of tablets and smartphones.

2. VENTRILOQUISM IN A NEW MEDIA ART CONTEXT

When an artist writes lines of code for a work that invites participation, the code (the voice of the artist) is speaking (is thrown) through the visitor's actions. But what role does this visitor take when interacting with the code? Is he simply a dummy, acting as a medium for the coder's voice? Curt Cloninger (2010) states that "computers don't execute code in a transcendent, metaphysical vacuum... Code is run on physical hardware in lived and present (albeit massively accelerated) time." In order to come into being, code "has to be read by and run on something – a person or a computer." During this performative moment, the code is united with both the hardware on which it runs and with the person who interprets the result of this running. If this is true, then each time a program runs, a unique organism emerges.

When asked in an interview about the word "ventriloquism" in relation to his work, Jonah Brucker-Cohen stated that it is relevant "if you are making work that allows the user to be heard through some other object - ie. not themselves." (Brucker-Cohen 2012) In this description, the participant's voice, rather than the artist's, is being thrown through an object:

Fig. 2 Diagram with participant and object

The ability or privilege to speak grants power because the voice can be used to direct others to take certain actions, to persuade, or to assert oneself as an individual in the world. Geoff Cox and Alex McLean's book *Speaking Code* begins with a quote by Theodor W. Adorno from "Institute for Deaf-Mutes" that contextualises ventriloquism within social power structures:

> *While the schools drill human beings in speech…, the pupils become increasingly mute… In the all-embracing system conversation becomes ventriloquism. (Adorno in Cox, 2013)*

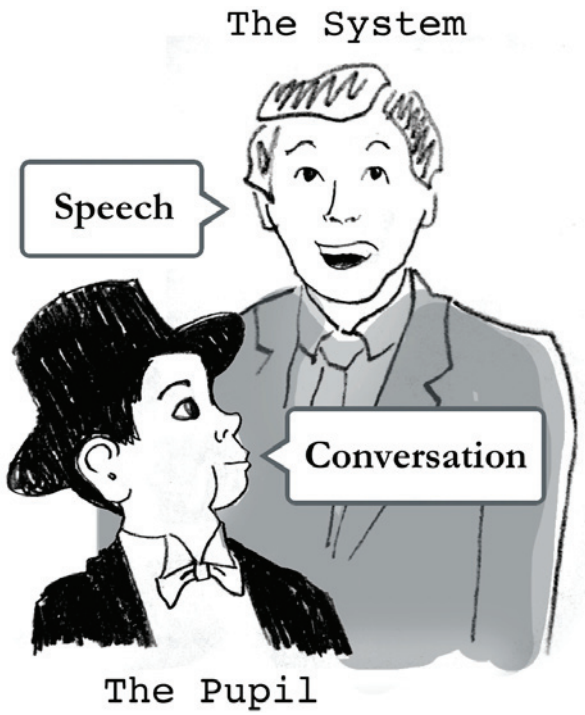

Fig. 3 Diagram for system-pupil relationship

In Figure 3, the pupils' mouths speak someone else's words. Like a ventriloquist's dummy, the pupil does not have the autonomy to articulate his own thoughts with his body and voice.

In Yvonne Rainer's *Carriage Discreteness*, part of *9 Evenings of Art and Engineering* (1966), Rainer stood offstage, choosing the actions and placement of people and objects from a pre-determined list of possibilities, then communicating these as instructions to her performers via walkie-talkie (Morris and Bardiot 2006). Choreographer and new media artist Kate Sicchio (2013) takes Rainer's idea into a contemporary context in her *Hacking Choreography* body of work.

In December 2013, Sicchio described *Hacking Choreography* at an *Into Practice*-run *Datarama* event at *Pixel Palace* in Newcastle, UK. In this presentation, she commented that her dancers don't always conform to her code as they can choose to either follow or disobey the given commands. In Rainer and Sicchio's works, the choreographer assumes the role of the ventriloquist while the dancers assume the role of the end-user/dummy. In both cases, however, the dummy has autonomy to act outside of the programmer's intent.

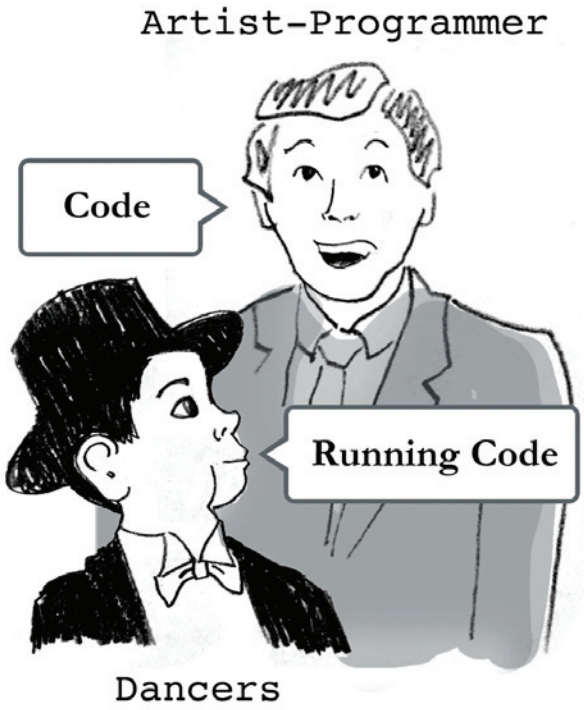

Fig. 4 Diagram for *Carriage Discreteness* and *Hacking Choreography*

Figures 1 through 4 illustrate directional transmissions of speech. Figure 1 shows an artist-programmer as the ventriloquist and the code running through the participant as a dummy. In Figure 2, the participant is the ventriloquist whose speech moves through an object. Figure 3 illustrates systems of power and social control as the system as ventriloquist dictates the speech of pupils. Finally, Figure 4 shows an artist-programmer articulating code that runs through the bodies of dancers who become dummies with an option of autonomy.

A ventriloquism analogy has been considered in two ways. First, the dummy as a powerless object who simply channels the voice of the ventriloquist through his mouth. Second, the dummy as a double of the ventrilo-

quist, the same voice appearing to emerge from another body. These models can allow us to consider the roles of participant and artist in new media artworks and whether the participant has autonomy in an interactive scenario. Through their interaction with the code, are they simply a channel for the artist-coder's voice or do they beome one entity with the artist and the code?

Brian Masummi (2002) considers code to be strictly protocol while bodies are analog and continuous. Thus, code can "potentialize, but only indirectly, through the experiential relays the reception of its outcomes sets in motion...Whatever inventiveness comes about, it is a result not of the coding itself but of its detour into the analog." This means that the body is that which translates the strictly pre-determined code into the analog.

The following two case studies discuss my own practical projects that were created to investigate the roles of a coding artist and a participating audience. Here we will consider channels of ventriloquism present in the works.

3. TOAST

Toast uses a coded translation device to mediate the speech of a performing participant. The project was initiated in 2011 while I was living in China with Mandarin language ability that limited me to only simple utterances. Although I could speak enough to purchase food at a local market, to relay basic directions to a taxi driver, or to tell someone my occupation and nationality, the attempts at discussion that ensued after these basic exchanges discouraged further conversation. In Beijing, I quickly became interested in making a translation device that would allow me to take conversation to a more complex level while highlighting the ridiculousness of using a machine to communicate rather than taking the proper steps to learn a language.

Work began by moving directly into the code using translation in a Processing[2] sketch, drawing on libraries to handle the speech-to-text functionality and the Google Translate integration.[3] Next, I began to search for agents of performativity that were already present in Chinese culture that could help participants to overcome potential shyness of speaking into the device.

During the initial stages of project research, I was attending various functions in and around Beijing and Shanghai including gallery openings, private dinners,

2 Processing is a Java-based programming language created at MIT Media Lab by Casey Raes and Benjamin Fry primarily used by artists and designers to create animations, generative images, or interactive artworks.

3 Florian Schulz's 2011 STT Library was used for speech to text (*http://www.getflourish.com*) and the Google Translate API (*https://developers.google.com/translate/*) for Google Translate integration.

and banquets. It occurred to me during these occasions that there was something special going on in the performance of a toast. In a landscape where public expression is not widely encouraged, the toast provided a forum for a person to express his views and emotions about the occasion at hand and his gratitude to guests or hosts.[4] I decided to draw upon the social code of the toast in my emerging project, as it was a performative gesture with cultural precedent.

Iteration, a property inherent to media artworks because of code's flexibility, was important to *Toast* because the project resulted in a series of tests rather than a single work. These tests included an audience-performer format at Barcamp Shanghai, a series of one-on-one experiments at Shanghai's Xinchejian Makerspace, and an installation prototype at the *Feijiacun Shangri-La Art Community Open Studio Exhibition* on December 1, 2012. This final iteration is described below.

Fig. 5 Feijiacun, Beijing iteration of *Toast*, Open Studio Exhibition curated by Filipa Reis (2012)

The Feijiacun *Toast* installation included instructions for visitors that were posted on the wall in both English and Mandarin. The text asked a participant to address his toast to an adjacent photograph of a common restaurant table, set with empty chairs in the round. This table image served as a blank canvas on which the visitor could imagine people seated for a meal.

The participant approached the computer, read the instructions, picked up the microphone in one hand (and optionally an empty wine glass in the other), and then spoke a toast to his fantastical companions at the dining table. His words were sent through the Processing sketch. Here, speech was turned into text in the spoken language. This text was sent to Google Translate where it

4 In Britain, a prominent part of a wedding ceremony is the series of toasts traditionally made by men in the wedding party (the best man, groom, and father of the bride). This proclivity of men to make toasts over women, in both Chinese and British culture, situates the toast as an official forum in which expressing emotion and sentiment is made socially acceptable by the formality of the performative act.

was translated it into the "opposite" language (English <> Mandarin), and then it was sent back to be displayed on the screen. Throughout the interaction, a web cam picked up a live-feed of the speaker's face, which was situated next to a speech bubble containing the final result of the translation.

In *Toast*, the translations returned by the code were almost always inaccurate and not a representation of what the speaker had actually said. This defeated the initial purpose of the project: to help a non-native speaker to be better understood. Instead, it highlighted the ineffectiveness and humour of machine translation and created an instance in which the code *does not* act as an Austinian performative,[5] saying what it does and doing what it says.

The following excerpt of code is activated if there is a button-press by a user, at which time the code "hears" the spoken language, turns it into a string, sends the string to be translated, then returns the result to be displayed:

```
println(utterance);
result = utterance;

if (buttonCaller == 1){

  String translatedText=Translate.
  DEFAULT.execute(result, Language.
  ENGLISH, Language.CHINESE_SIMPLIFIED);
  println(translatedText);
  result = translatedText;
  }
```

In this code snippet, the variable *result* initially represents *utterance*, or the words spoken by the participant. Inside of the if statement, *result* becomes equivalent to the translation (*translatedText*). While the translation algorithm considers the original utterance and the result of the translation to be equivalent, the human participant knows that the final translation is often quite distant from what was actually said.

Although the code imposed translations on the *Toast* participants, they were free to interpret the text and image, drawing additional meaning or humour from the juxtapositions provided. Because most of the people attending the Feijiacun exhibition could speak some English and some Mandarin, among other languages, the

5 In *How to Do Things with Words*, J.L Austin defines the linguistic performative, saying it "refers to a class of expressions that are not descriptive and have no truth value, but rather in their very utterance, do something (I bet, I promise…)." (Austin, 1962)

participants were aware of these missed translations. This understanding led to them becoming actively engaged with the piece, gathering in groups, and creating a playful performative atmosphere around the spoken utterances, the doubling of a participant's likeness on-screen, and the floating speech-bubble translations.

A ventriloquism diagram for *Toast* looks like this:

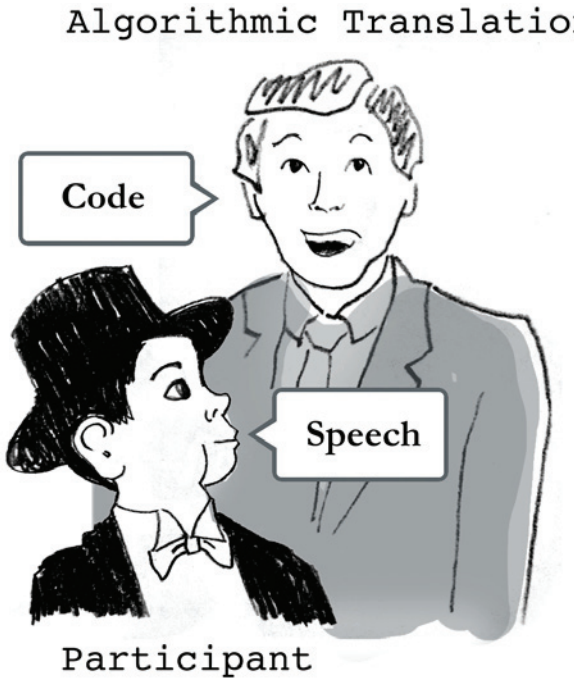

Fig. 6 Diagram for *Toast*

In Figure 6, the ventriloquist embodies the translation. This translation is dictated by an algorithm, which sends speech back through the mouth of the participant.

4. VENTRILOQUISMS FOR FUN AND PROFIT

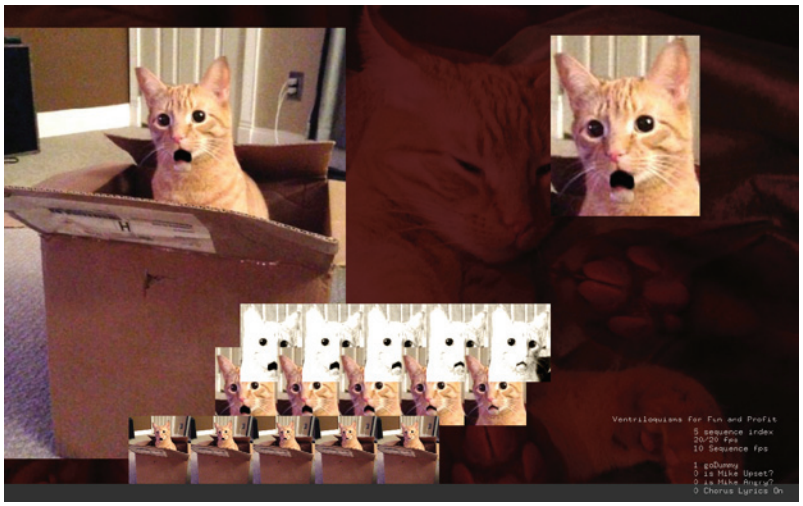

Fig. 7 *Ventriloquisms for Fun and Profit*, audio asset: *http://blurringartandlife.com/vb/ventriloquisms.html (2013)*

In *Ventriloquisms for Fun and Profit*, I took the role of coder and performer as an audience was invited to participate. My voice was thrown through a self-coded puppet, while a call and response song made both performer and audience into dummies. In this performance, the dummy was a puppet of a cat, coded in openFrameworks rather than built with wood, strings, and glue.

The performance took place on April 26, 2013 at Datarama, Pixel Palace, Newcastle, UK. The piece began by engaging the audience in a song by instructing them to repeat the phrase "Oh Mona" after each artist-led line of verse. Between verses, they sang along with a chorus, "Oh Mona you shall be free...". The text to be sung was displayed onscreen. When written in pseudo code, these instructions to the audience create an if-else-statement:

```
if (line of verse is complete){
Sing "Oh Mona";
};

else if(entire verse is compete){
Sing chorus;
};
```

At Datarama, the audience willingly participated, singing along and "joining in" or following the instructions. When everyone in the room was singing, social codes enforced individual participation. This call and response established the following loop between the audience, the artist, and the running code:

Artist → Code → participatingAudience → Code →

To begin the performance, I changed a Boolean value in the code from false to true in order to get the dummy "working" (See Figure 8). This moment of live coding referenced a common act in which a ventriloquist takes apart his dummy's head and tinkers with it using a spanner, pretending to get a non-functioning part, such as the mouth, moving again.[6]

When I changed the value of makeDummy from false to true, the dummy appeared to have suddenly gained the ability to move his mouth (the change doesn't actually serve a functional purpose within the running code, but acts as a visual gag for the audience). The code-saavy Datarama audience laughed at this moment.

Fig. 8 The "Make Dummy" code

```
//makeDummy
makeDummy = true;
```

After the song, I performed a ventriloquist act, *At the Dentist* from an instruction manual for aspiring ventriloquists titled *Ventriloquism for Fun and Profit* (Winchell 1954). Winchell's *At the Dentist* sketch follows a common trope of ventriloquist performances that create humour through violence between the ventriloquist and the dummy.

At the beginning of *At the Dentist*, the dummy discovers that he is a patient at a dentist office and the ventriloquist is the dentist (see the quote at the beginning of this paper). The humour here lies in the fact that the dummy is surprised to find out that he is going to be subjected to a potentially painful procedure. This threat of violence between versions of the self is analogous to our interactions with code through an avatar which allows us to embody violent actions that we would not enact in our everyday lives. There is also an underlying power structure between a dentist and a patient similar to the pupil/system dichotomy seen in Figure 3. In these relationships, the dummy, patient and pupil are at the bottom rung of the power structure, subject to the speech of the ventriloquist, dentist, or a system of authority.

A *Ventriloquisms for Fun and Profit* diagram might look like this:

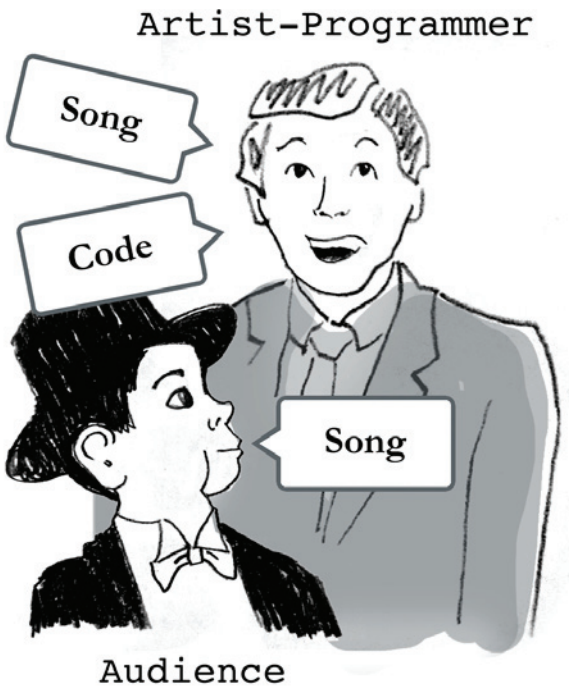

Fig. 9 Diagram for *Ventriloquisms for Fun and Profit*

6 My grandfather, Burke Bradbury, an amateur ventriloquist, often performed this trick with his dummy Oscar, who would continue to speak and protest throughout the "operation".

In Figure 9, the ventriloquist is equivalent to the artist-programmer while the dummy represents the audience as they sing the chorus of “Oh Mona”.

5. REFLECTION

During the March 2014 CRUMB discussion on the topic of *The Performativity of Code*,[7] new media artist Jack Stenner (2014) stated that humans are “the “neuronal” support for technology.” Stenner wrote that “By “othering” technology we can absolve ourselves and shift responsibility. It’s an unnecessary binarization of a more complex relationship.” Stenner sees our selves and technology (code) as one and the same. This union of code and body is reflected in the analogies of participation and ventriloquism described in the above case studies.

In *Toast*, while participants speak through a translation device that doesn’t translate accurately, the code becomes the ventriloquist, imposing meaning on the speaker’s image. The code places words in the mouth of the participant who is left to interpret the translation as it hovers beside his face-image (a doubling of the self). While code is the ventriloquist, the participant is the dummy with a sense of interpretive autonomy.

The Ventriloquisms for Fun and Profit performance situates the artist and audience as performers within a system dictated by artist-written code in which underlying social codes influence audience members’ participation. While the audience is the dummy in Figure 9, the artist is also a dummy during the performance, as both parties are controlled by the code and the code (as ventriloquist) speaks through them.

6. CONCLUSIONS

In a conversation, words are spoken by one party, then heard, considered, and responded to by another. This exchange continues in a loop. In a toast, one person speaks to a group in a performative moment. An audience hears this speech and clinks their glasses, initiating a consecration of the words. In a call and response song, one person holding the power of performance sings a line and a group responds with a pre-established, repetitive phrase. In ventriloquism, the ventriloquist speaks as himself, but simultaneously and in another voice, channels his speech through the dummy.

7 CRUMB (*http://crumbweb.org/*) run a new media curating discussion list that proposes month-long discussion topics with list members and invited participants. The March 2014 topic was *The Performativity of Code* and was mediated by CRUMB researchers Victoria Bradbury and Suzy O’Hara with 17 invited respondents. The full discussion may be found on the CRUMB online archives.

In viewing a ventriloquist performance and while interacting with code, an audience or participant often accepts and ignores the workings behind the scenes in order to accept the illusion. In ventriloquism, the trick is obvious, but with code, layers of obfuscation, translation, and compilation hide the source, making it unclear exactly how the programme controls the participants' actions.

In each of the analogies of ventriloquism described in this paper, code is not the other, but is equivalent to the author and participant as it runs through all of the entities within the system. Bodies and voices are not separate from code.

REFERENCES

Austin, J. L. *How to do Things with Words*, Cambridge MA: Harvard University Press, 1962.

Brucker-Cohen, Jonah. *The Performativity of Code.* Interviewed by Victoria Bradbury [Email], June 8 2012.

Clair, J.M.S. *Novel Listening: Background Sound and the Technologies of Aural Control.* The University of Iowa, 2007.

Cloninger, Curt. 2010. *GltchLnguistx: The Machine in the Ghost / Static Trapped in Mouths.* [online] Available at: http://lab404.com/glitch/ [accessed November 6, 2011].

Cox, Geoff and Alex McLean. *Speaking Code.* Cambridge MA: MIT Press, 2013.

Galloway, Alexander R. *Protocol : How Control Exists After Decentralization.* Cambridge MA: MIT Press, 2004.

Graham, Beryl. Interaction/Participation: Disembodied Performance in New Media Art. In: J. Harris eds. *Dead History, Live Art?* Liverpool: Liverpool University Press, 2007.

Massumi, Brian. *Parables for the Virtual : Movement, Affect, Sensation.* Durham NC: Duke University Press, 2002.

Morris, Catherine, and Clarisse Bardiot. *9 Evenings Reconsidered: Art, Theatre, and Engineering*, 1966. 1st ed. Cambridge, MA: MIT List Visual Arts Center, 2006.

Sicchio, Kate. 2013. *Hacking Choreography.* [online] Available at: http://blog.sicchio.com/?page_id=236 [Accessed 18 December 2013].

Stenner, Jack. CRUMB New Media Curating discussion list: The Performativity of Code [Online discussion]. Message posted to: https://www.jiscmail.ac.uk/cgi-bin/webadmin?A1=ind1403&L=new-media-curating March 17, 2014.

Winchell, Paul. *Ventriloquism for Fun and Profit.* Baltimore: I. & M. Ottenheimer, 1954.