On Not Taking What You Are Given

I, too, would like to acknowledge that we are on Treaty 7 territory. I hope my talk here will honour and respect all of the ancestors that have walked this land before me.

I want to thank Derek for inviting me to speak, and Stephanie for her assistance in getting everything arranged. And I want to thank all the previous speakers, for sharing their good minds with us. You have given me much to think about.

The title of the talk is "On Not Taking What You Are Given”. I hope its meaning will be clear by the time we finish.

[Slide: Choice]

I’ll start with a reading of an interactive poem of mine. Ashok has kindly agreed to accompany me on the interactive version of the text, which you’ll see on the screen. The title is: “No Choice About the Terminology”.

`{Performance: Choice}`

I’m trying several new things today. I don’t normally mix performance readings with my more academic-y presentations. But, as you can see, I’m going to mix them together.

The other new thing is in terms of content. Normally, when I speak publicly, I talk about one or the other of the two sides of my practice.

[Slide: Writing Complex]

One side is Writing Complex, a series of conceptual, computational and creative engagements with digital texts and typography.

[Slide: AbTeC]

The other side is Aboriginal Territories in Cyberspace, or AbTeC, a series of projects in the Indigenous community looking at how we use digital technologies to tell our histories, illuminate the present, and dream of the future.

[Slide: Writing Complex + AbTeC]

I'm using the topic of this conference, "Where Next?", as an excuse to do something I've wanted to do for some time, which is to articulate how these two, seemingly disparate trajectories are actually closely related to and feed off one another. And, really, to help me figure out 'where next?' for my own work.

I hope you'll bear with me as I attempt this. It's going to require several moving parts; we'll see how they hold together.

[Slide: Pretty Jesus]

Now, my first creative practice was and is poetry. Why poetry? At root, who knows? I like language. I adore the way it can be sculpted and moulded, the way in which I can inflect it using tense and pronunciation, that moment when I've finished editing the perfect line, one so dense and tightly coupled that you couldn't slip a molecule between its adjacent components. I am enamoured with the alchemical process of transforming thoughts into words, which then get transformed into thoughts in somebody else's head.

[Slide: D Niatum The Years in the Blood]

In hindsight and forced to articulate a reason, I would say it's also because poetry forces one to pay very close attention to how language is constructed. Each word is precious, every bit of punctuation considered, and every rephrasing examined like a jeweller strategizing how to cut a diamond. Syntax is a puzzle with multiple solutions, each one of which carries with it its own set of connotations. I can use different formal microstructures—rhyme, meter, assonance, dissonance— and different macro-structures—sonnet, sestina, rondel, villanelle—each triggering a different configuration of sonic and semantic resonances in your brain. Words, syntax, form—these are the material out of which poems are made.

[Slide: Unique Sequence Code]

When I reached university as an undergraduate I was forced to take a programming class. It was a revelation. Here was a form of poetry. The rules were strict, yes. The syntax strange, indeed. But it was a compressed, concise, expressive language that required a sustained and eye-wateringly detailed engagement with structure. An elegant recursive function is breathtaking in its efficiency. A well-constructed data structure is a thing of beauty. And it is all functional, language that makes things *happen*—pixels switch on, notes get played, rockets fly. Alchemical to the core.

My interest in programming grew into a more general fascination with computation and its material qualities. I wanted to understand how the different strata necessary for computation affected what I could do with it. I wanted to understand how I could use those material qualities creatively.

[Slide: WordNozzle video]

I started exploring the place where poetics and programming came together. Given that I am speaking at the Alberta College of Art and Design, I thought it would be fun to show my thesis project from when I was in art school. The work was called WordNozzle; here is a video from 1996—almost twenty years ago! The question I was pursuing with it was, what would happen if you were to take the paint inside of a spray can and replace it with letters and words and sentences? What would it be like to spray text onto a surface, manipulating the font and size and color as you went? What kind of new texts might I write? What kind of new readers might those texts engender?

Developing WordNozzle brought me face-to-face with a core *material* problem. At that time, you had programs that treated text as ASCII, allowing you to edit the text as language. You had programs that treated text as a collection of pixels, where you could edit the visual appearance of the text. But you didn't have anything that allowed you to do 'live text', or text that could be manipulated linguistically and visually.

[Slide: ASCII-Wall]

This forced you into a very cumbersome process, where you composed the text in a word processor, then rendered it into a collection of pixels that you would run through a series of special effects. Once you did this, though, the language-ness—the semantics of the text—was lost. The special effects software had no way of knowing that this pixel belonged to an 'h' and that to an 'x'. And the loss of language-ness meant that you could no longer edit the text as a writer.

This was a huge barrier to writing drafts and sketching at the same time. Once I had applied visual transformations, I couldn't then decide to edit the text—re-order the words, add a line, change the capitalization—without starting again from scratch. This killed the ability to experiment with both word and image at the same time. And without that ability to experiment in both dimensions, it was very difficult to get the visions in my head of a 'living text' onto the screen.

So I decided to write my own software. I knew the problem was solvable at the computational layer, meaning I knew it could be done. It's just that nobody else had thought it important enough to do.

[Slide: Ingredients]

I and my team brought together techniques from digital typography, three-d modeling, word processing, vector graphics, and computational linguistics, and remixed them into a new material.

[Slide: It's Alive!]

The first result was the software application It's Alive!,

[Slide: It's Alive! video]

which we called "the bastard offspring of Microsoft Word and Adobe After Effects". You could add visual and interactive behaviours to the text as easily as changing the font or size.

[Slide: Mr. Softie logo]

It's Alive! led to Mr. Softie,

[Slide: Mr. Softie Video]

which added the ability to output video animations and high-resolution stills.

[Slide: NextText logo]

They both fed into NextText, an architecture for manipulating text that we implemented in a number of languages.

[Slide: NextText structure]

This work immersed me deeper into the materiality of computation.What data structures were appropriate for working with language vs. image, how *this* programming language made this over here easy, how *that* programming language made it hard, how different operating systems influenced the kind of work done with them, how different hardware demanded different sorts of engagement.

My goal was to able to write, design, and program at the same time. It took me almost fifteen years to get there, where we had created the tools I needed that allowed me to experiment freely in semantic, visual, and computational dimensions simultaneously. That freedom allowed me to weave the computational material in a way that was expressively powerful in precisely the way that I wanted it to be.

[Slide: PoEMM Logo or college]

The cumulation of this research-creation trajectory is the P.o.E.M.M. Cycle, a series of eight touch-driven text works. The poems I'm reading tonight come from the Cycle. Time for another one: "The Summer the Rattlesnakes Came". Ashok?

```
{Performance: Rattlesnakes}
```

[Slide: Hermeneutics]

While I was a undergraduate, I also studied philosophy. Specifically German philosophy. Hegel, Heidegger, and the whole subspecialty of Hermenuetics. Hermenuetics, at its core, is the philosophy and methodology of interpreting texts. When you start thinking hard about how to interpret a text, you run pretty quickly into the question of context, or what is the background against which you need to understand this particular text. And, when you've tripped, hard, on that question, you find yourself face down in a puddle of epistemology—the study of the nature and scope of knowledge itself.

[Slide: Symbolic Systems logo]

I was lucky that, while I was discovering these two strange new worlds of philosophy and programming, a new degree program called Symbolic Systems had just been started at Stanford.

[Slide: Symbolic Systems quote]

Symbolic Systems was founded to explore how "computer systems, robots, and people are all examples of symbolic systems, agents that use meaningful symbols to represent the world around them so as to communicate and generally act in the world. The notions of symbol, meaning, representation, information, and action are at the heart of the study of symbolic systems." The program provided a home where asking the question of how poetry, philosophy and computation intersected wasn't seen as weird, or, worse, dilettantish How it was all, in some way, the study of the nature and scope of knowledge. And it was also home to great group of faculty and fellow students interested in understanding how computational systems both reflected and shaped the human experience.

[Slide ?]

Fast forward twenty-five years, and that set of concerns has proven prescient. We are enmeshed in computational structures, to the point they determine as much, if not more, of our daily lives than other such society-wide symbolic systems such as the law. But what does not receive much attention is how firmly our computational systems embed a certain worldview, a certain set of assumptions about humans and how we operate in the world. That we are rational, that we compartmentalize and categorize cleanly, that we separate reasoning from emoting, the human from the animal and the natural, that our goal, always, is to dominate the world rather than live in it.

[Slide ?]

My experiments with text and typography are an example of this written small--the software developers of the time simply did not envision that somebody would have a different approach to text than that which was expected by the desktop publishing industry or film affects industry. Small potatoes, of concern only to me and a very small group of people that live at the intersection of poetry and interactivity.

[Slide ?]

But when we start writing large, when we begin, in the words of computational philosopher and poet David Jhave Johnston, when we begin 'writing the thoughts of systems', things get political.

[Slide: Loretta Todd]

They become about how power is exercised. In the same way the law embodies the dominant culture's expectations about peoples' behaviour, computational systems materialize the dominant culture's expectations of what counts as data, what kinds of operations you can perform on that data, and what is a valid outcome.

And in a North America where it is increasingly difficult to do anything without touching on a computational interface of some sort, the decisions that developers are making *all the time* have profound consequences for how we live our lives.

[Slide: AbTeC logo]

This brings me, finally, to Aboriginal Territories in Cyberspace, or AbTeC. The artist Skawennati and I founded AbTeC because we saw that these computational systems were being used to define a new territory, the archipelago of websites, social media services, shared virtual environments, corporate data stores and multiplayer video games we call cyberspace. We—Indigenous people—needed to stake out ground in it. Cyberspace was clearly going to become central to our lives, and we wanted Indigenous people out there in it, exploring the blank spaces and filling them up with our voices, our faces, and our dances.

[Slide: Skins logo]

AbTeC approached this challenge from two directions. The Skins

Workshops on Aboriginal Storytelling and Video Game Design train Indigenous youth in the tools of digital production, transforming them from simply consumers to producers of work in this new world.

[Slide: Skins Walkthrough Video]

We designed the workshops to integrate storytellers and storytelling from the community with creative and technical instruction. We ran four of these workshops, and each produced a game where the students remediated stories from their oral tradition into an interactive, playable narrative. The process helped the students' connect their cultural knowledge, passed down from the past, with the knowledge of the technology it takes to shape the conversation in the present.

[Slide: TimeTraveller™ image]

The second direction was to actively imagine ourselves into these new territories. Skawennati's TimeTraveller™ project was the biggest effort we took in this direction. This is a series of nine videos made using machinima, or machine cinema, where you use virtual environments to make videos.

[Slide: TimeTraveller™ Video]

TimeTraveller™ is the story of a young Mohawk man from the 22nd century who 'visits' events of importance to First Nations history. As he experiences the other side of the one-sided fairy tales that pass for history in this country, the series constructs a counter-narrative that calls into question the basic legal and cultural structure of the mainstream society.

[Slide: IIF logo]

We are now working on the Initiative for Indigenous Futures, or IIF. IIF builds on AbTeC, but re-oriented to facilitating a conversation in the Indigenous community about who we want to be seven generations from now--or even further out along the timeline.

[Slide: Momoday quote]

At the same time, we're going to push further into computational systems than AbTeC did. Whereas the previous Skins workshops taught students how to manipulate the software necessary for constructing experiences, we want the new Skins workshops to go further down the stack.

[Slide: Stack]

We want to start teaching participants how to make software applications. Then how to make libraries and other middleware that those tools depend on. Then how to write their own operating system kernel. And we'll keep going down, to the hardware itself.

Ultimately, the dream is to invent new computational methods and tools that can accommodate world-views different than the Cartesian logic out of which our current computational machinery springs.

The goal is to be the creators of the materials that we are shaping. The goal is to restructure computation to reflect the way we want to live in the world. The goal is to look forward, to teach ourselves not only how to use these technologies but also how to *make* new technologies.

[Slide: Veregge]

We have the opportunity and the obligation to involve ourselves intimately in the shaping of the systems and structures in which we will be living for the next five hundred years. We have the opportunity to *not* take what we are given, and instead fashion something anew.

[Slide: Rattlesnakes]

I'll close with a final reading. This is "The World That Surrounds You Wants Your Death".

{Performance: The World That Surrounds You Wants Your Death}

Thank you.

[Slide: Veregge]