THE NEURO-LOGIC OF SOFTWARE ART

WILLIAM HART

This paper considers programming in relation to the International linguistics of Roy Harris and the Expanded Mind hypothesis of Andy Clark. It is argued that the use of the term 'language' when referring to programming is a category error that has implications for the practice of Software Art. Developing a practice of Software Art requires an open-ended and heterogeneous approach to coding involving a range of cognitive states.

"*The computer is a kind of wishful self-portrait... a compendium of abilities we have as humans aspired to but are not very gifted at. We need a much clearer understanding of this complex relationship. Without this understanding we will be unable to find an appropriate partnership with our creations.*" [1]

Introduction

Over ten years ago I undertook my first software art project entitled The Writing Machine. At the time it had been some years since I had engaged in a serious programming project, and this was the first time aside from some scripting that I had undertaken programming with the object of an 'art' outcome, but I had had considerable previous experience in large and small experimental software projects. An initial prototype written in C for the artwork that had the essential behaviours was produced quite quickly, but then the project stalled - and failed to reach an aesthetic resolution. It was as though the material of the project became heavy, brittle, and eventually unworkable. I've since spent a considerable amount of time considering why. It would be easy to say that this failure was due to a poor software engineering methodology, but while making an artwork sometimes generates engineering problems, engineering does not describe the methodologies that an artist uses. Artworks rarely address a problem or provide functionality, in the way that engineering processes do. This has led me to explore the qualities of different programming languages, looking at processes for rapid development and flexible refinement, but that also allow for diversity of expression.

As part of this exploration I have looked closely at the relationship between humans and machines, between the formal constructs of programming and human language, and ask, what is relationship we have with the machine and what are the constraints on using it as a creative medium. I contend, that in considering programming as an expressive medium it is necessary to consider the types of cognitive states artists engage in and examine the concept of language as it relates to the domains of human discourse and computer programming. In contemporary times we are very comfortable with computing devices, with the idea that these devices occupy an intimate part of our lives, and the mainstream view is developing that it is valid for artists to be involved with, programming, hacking, tinkering - making and unmaking this machinery, as part of a creative practice that results in an experience called art.

The Artist Programmer

There appears to be a consensus that the artist engaged with programming as a creative medium should, as their skills develop, move from conceptually simpler to more rigorous (in a software engineering sense) programming environments. For instance from a visual programming with node based metaphors such as Max/MSP or Quartz Composer, through customised IDE's (Integrated Development Environment) with scaffolded programming support such as Processing graduating to C++ using API's (Application Programming Interface) such as OpenFrameworks or Cinder. A common rationale is that C++ applications are faster, more extensible, capable of increased complexity and have greater robustness. In this paper I argue that software art is not software engineering, that a different approach is required, to achieve the flexibility, richness and ease of comprehension that expressive software art requires.

It isn't necessary to look too far back beyond the past decade to find a general unease with the possibility of the coexistence of technical proficiency in computer technology and creative expression. Current histories of software art often claim a heritage from the conceptual art movement of the 1960s, but few are enthusiastic in claiming a lineage from the Computer Art movement of the 60s, 70s and 80s, a movement which many would still claim failed to produce art worthy of consideration. [2]

The Program as *Art*efact

Perhaps this is partially because the concept of what art is has changed since these times. We have seen a move from the emphasis upon the art object to an emphasis upon relation or process. I remember reading an article and looking at images produced by Harold Cohen's drawing software AARON in the early 1980s and being disturbed and slightly enraged. Partially my discomfort was due to the article (and Cohen) ascribing intelligence and creativity to the software - whereas all I could see was software that produced pictures with recognisable forms, but that were always the somehow the same. Years later as I pondered the problems of writing software and creativity I began to realise that while it is true to say that AARON failed to produce engaging art objects or to display 'creativity'; it is the project in its entirety and the knowledge it represents which is the artwork.

Throughout the history of artists' engagement with computers, there has been recognition that something 'new' or novel is happening, a new medium, a novel approach to making and disseminating art - a different mode of expression than had been seen before.

The Computer Art of the 1970s and 80s suffered the disregard of the mainstream art establishment as much because of its inability to demonstrate the value of this 'newness' in ways other than novelty, and because of its association with the dehumanising program of hard AI and the military industrial complex. With the advent of the personal computer revolution and development of desktop publishing and image editing, the computer demonstrated an ability to simulate processes of traditional media (at first quite crudely and was met with a degree of scepticism by practitioners skilled in traditional analogue media). It was this ability to firstly approximate through simulation the processes of traditional media and then such as with the case of video quite quickly surpass the processes of analogue media that led to the general acceptance of the computer as a tool for artists - not its capacity for novelty.

The novelty of Computer Art?

What is this new thing though? Dominic Lopes in "A philosophy of computer art" makes a distinction between Computer Art and Digital Art, the latter he defines as art which rely upon the ability of the computer to encode and manipulate data, and is not a new art form - a digital image is still an image, digital music is still a kind of music. [3]Computer Art he defines as art works, which when run on a computer have a quality of interactivity. So for Lopes the 'new' thing is the quality of being interactive. Interactivity as the prerequisite of difference alone seems to hark back to the new media of the 1990s as a way of distinguishing the 'newness' of art made with the computer. For the purposes of this paper I would like to instead explore this idea of 'newness' as being a cognitive loop between artist and computer, rather than as a dialogue the artist has with the computer or something that the artist expresses through the computer. Digital Art is now the mainstream, it is ubiquitous and its tools of production are invisible. There are numerous social and political problems with the solely media centric approach of digital art, but they are not what I want to address here. Instead I want to look more closely at the processes and context of the engagement between human and computer we call programming, and how it could result in a thing we call art, particularly where the artist and programmer are the same person.

I would assert that this activity we call programming is more than a utilitarian or instrumental one, that the requirements for a programming environment are different for an art activity than for an engineering one. An art project may have non-specific goals and will certainly have different criteria for determining points of completion. Art historian Barbara Stafford places analogy at the heart of the art experience, in making, as much as receiving art we are engaging in 'analogical thinking', satisfying art experiences generally engage through either an unstable and shifting meanings, or in a richly nuanced web of correspondences. [4]

Analogy, Language and Programming

Analogy / Metaphor is very powerful, it is arguably the essence of cognitive process, it gives the power to create and manipulate complex abstractions. Where analogy is in constant use it becomes invisible, and we cease to be aware of the underlying assumptions. We all know that programming languages are not the same as human 'natural' languages, but implicit in almost all theoretical discussion is the idea that on some level that it is. We talk about codes, and encoding interchangeably between analogue and digital, between organic and machine systems. But what if there is no correspondence between language and programming dialects.

There have been a long history of technological metaphors used to describe aspects of human cognition; many have become deeply embedded in consensual language and understandings. For instance the telegraph and telephone exchange were common metaphors for brain function in the early 20th Century, the photographic image as an analogue for visual perception is still a widely held naive conception. The computer as an analogue for the brain, and software for the mind, has been widely used for the past 50 years, although it is also a metaphor that is violently rejected by many. Sometimes terminologies that have been analogised from one discipline of thought to another create a further implicit confusion. Take for instance the term 'ontology' it has distinct (but analogous) meanings, whether you are discussing philosophy, cognitive science or computing. In philosophy ontology is concerned with the existence and relationships of categories of being, whereas in information science ontology is a representation of a domain of knowledge based on a formal description of concepts and relationships, taxonomy and properties. So in one discipline ontology is metaphysical; open to interpretation and in the other it is formal; precise and closed. When we discuss ontologies in terms of art and programming there is naturally a bleed from one conception to the other, a confusion between metaphysical and material modes of expression.

CHOMSKY'S UNIVERSAL GRAMMAR AND ARTIFICAL INTELLIGENCE

In the early 1950s, at a similar time to the development of the first computer programming languages such as FORTRAN and LISP, Noam Chomsky developed his linguistic theory of Universal Grammar, that is, there is a universal structure of grammar hardwired into the brain, with the implication that all human languages share a common set of fundamental rules, a formal description. Programming languages are 'Turing Complete', that is formally a logical construction in one language can also be represented in another, even though the code and logical structures of the programming languages may be quite different. Chomsky also constructed a hierarchy of formal languages, with Turing complete formal languages at the lowest level, and several levels of more complex formal grammars above that. The implications from Chomsky's theories were clear; all human languages were syntactically equivalent to each other, as were all programming languages, with programming languages being a simplified version of the languages of humans. So the idea that we think and communicate through code entered the public consciousness and was firmly embedded through the use of terms like programming 'language'.

To understand how prevalent was this assumption it is instructive to look at the history of machine language translation, the optimism in the field in the 1950s repeatedly asserted that machine translation was a problem that would be solved in the near future. [5]

Cognitive Psychologist and author Steven Pinker currently champions the idea of a Universal Human grammar, and popularised Chomsky's theories widely in his 1994 book "The Language Instinct." [6] But the theories and debates around the acquisition and representation of language in the mind are complex and vigorous, and there are a number of opposing theories to this idea that language is in someway hardwired into the brain.

BUT IF LANGUAGE IS NOT A CODE...

One of the theories that Chomsky's Universal Grammar displaced was the Saphir-Whorf hypothesis or linguistic relativity. The hypothesis, briefly stated, is that the language you speak with determines what and how you think; the form and structure of different languages can be unique, which lead to different conceptions and understanding of the world. In recent years the credibility of the Saphir-Whorf hypothesis has been revived by the work of Cognitive Psychologist Lera Boroditsky. Boroditsky looks at indigenous peoples and their languages, and how constructs available in the language they use affect the way they interact with each other and their environment. For instance she describes how some languages such as Kuuk Thaayorre spoken by indigenous people in Northern Australia, do not have concepts for relative spatial location such as left and right, but instead always use absolute directions, for instance they would not say you are in front of me, they would say you are to the north. Consequently people who natively speak this language always know where they are orientated, a characteristic not shared by people who use languages with concepts of relative location. [7]

AND COMMUNICATION IS MORE THAN LANGUAGE...

Integrational Linguistics is a branch of linguistics that was developed by oxford scholar Roy Harris during the 1980s. Harris criticises what he denotes as the "language myth" that dominates Western linguistics. Integrational Linguistics rejects several implicit assumptions in conventional linguistics, particularly the idea of communication occurring through a system of signs that have a meaning and existence separate from the context in which the communication is occurring, or that human communication occurs through a mechanistic process of mental formulation – the idea that meaning is encoded into speech where is it transmitted to the listener who decodes it into a mental representation. Consider an example of integrated communication, if I am carrying a load of heavy boxes, and I ask you for help, you assess my difficulty and take a box from me, to which I respond 'thanks'. We have engaged in an integrated act of communication involving spoken word, gesture and physical action. For the Integrationist, communication is always a collaborative creative act steeped in the context of the participants, it relies as much on physical acts or responses as it does on verbal or signs. [8]

Cyborgs: thinking through objects

Embodied cognition is the theory that a body is essential for thought, that physicality, sensation and emotion are essential components of mind. Philosopher Andy Clark takes this a step further with the theory of the Extended Mind. In the extended mind hypothesis the mind can extend beyond the brain and body into the external environment through objects such as notebooks as part of the cognitive process. In the original paper Clark and Chalmers argued that if a person, Otto, has Alzheimer's disease and routinely relies on a notebook to act as his memory, the notebook is not merely an instrument, but under Clark's parity principle is cognitively equivalent to a normally functioning memory. As Otto's cognition extends to his notebook, it is effectively his memory. Clark contends that humans are naturally cyborgs, creatures who routinely cognitively co-opt artefacts. [9] Interviewed on Australian radio Clark relates a story about how once when he lost his laptop it was the cyborg equivalent of having a stroke, which left him shaken, dazed and confused, its impact was worse than when he had an actual mild stroke several years later. Most people who have lost a laptop and through it engage with the world of ideas will relate to Clark's anecdote.

After pondering the idea of the dynamic and creative process of communication that Integrational Linguistics describes, which relies little on the transmission of coded signals but upon body, environment, history and context to develop understanding, or the way that the Extended Mind hypothesis argues that we use the environment around us, it is hard to continue to think of artworks as static objects, but as dynamic cognitive systems which draw us into a dialogue of analogy and sensation, or as Barbara Stafford describes them, "echo objects." [10]

Software art is interesting philosophically and cognitively not just because of its potential 'newness', but because it sits at such a distinct interface between formal logic and fuzzy expression.

Conclusions

When we engage in programming, although we use terms like 'language' and 'writing' we are not engaged in discourse with or through the computer, but are engaging in an act of cognition. This extended cognitive state involves a variety of different modes as the software develops through; inspiration, discovery and exploration, experimentation, play, construction, reflection, analysis and intuitive modelling. When engaging in a creative process through crafting software, it is not enough to just string a sequence of function calls together, or to solve a complex technical problem, or to demonstrate the capabilities of an algorithm. It is the engagement with the full spectrum of cognitive modes that leads to a resolved creative outcome.

Many of these cognitive modes are common to all forms of art practice, some like the one I have labelled discovery and exploration are more specific to software art. To move a project beyond its initial inspiration may require lengthy exploration to find a way to proceed or a form to follow. Intuitive modelling or tweaking is unique to technical processes, as a program is developed there are often a large number of arbitrary variables that control behaviour and other qualities. Mathematicians refer to this as a parametric phase space, the multidimensional space describing the range of possible behaviours the program has. A key (and often very time consuming) aspect of the development of the program as an artwork is exploring and experimenting with different values and combinations of parameters. It is through this play that an intuitive sense of the shape and form of this phase space develops which leads to the final selection of values and consequent behaviours.

As an artist-programmer I advocate adopting hybrid approaches mixing languages with high levels of abstraction with low level high performance code based on the type of cognitive engagement being undertaken. I am not trying to engineer an application or create aesthetic code. I am struggling to enact a process of communication with the viewer.

References and Notes:

1. *David Rokeby, "The Computer as Prosthetic Organ of Philosophy,"Dictung-Digital 29, no 3 (2003).*
2. *Grant D Taylor, "The Machine That Made Science Art: The Troubled History of Computer Art 1963-1989" (Thesis, University of Western Australia, 2004).*
3. *Dominic McIver Lopes, From the Author's Perspective: A Philosophy of Computer Art, American Society for Aesthetics Newsletter 29, no. 1 (Spring 2009): 1-3.*
4. *Barbara Maria Stafford, Visual Analogy: Consciousness as the Art of Connecting (Cambridge, MA: The MIT Press, 1999).*
5. *John Hutchins, "The precursors and pioneers" in Machine translation: past, present, future (Chichester: Ellis Horwood, 1986), 6-20.*
6. *Steven Pinker, The Language Instinct: The New Science of Language and Mind(London: Penguin, 1995).*
7. *Lera Boroditsky, "How Language Shapes Thought," in Scientific American(February 2011): 63-65.*
8. *Roy Harris, Introduction to Integrational Linguistics (Oxford: Pergamon, 1998).*
9. *Andy Clark and David Chalmers, "The Extended Mind," in Supersizing the Mind: Embodiment, Action, and Cognitive Extension, ed. Andy Clark, 220-232 (Oxford; New York: Oxford University Press, 2008).*
10. *Barbara Maria Stafford, Echo Objects: The Cognitive Work of Images(Chicago; London: University of Chicago Press, 2007).*