AN EXPLORATION

OF THE VISUAL

MIND OF THE

SOFTWARE

ART

IS

T

EWAN STEEL

Introduction

Digital technologies have changed profoundly the ways in which the artist can develop a visual language. They have expanded the vocabulary of gesture and composition, the means by which, traditionally, visual artists have chosen to communicate complex ideas and emotions. They have created new forms that blur the boundaries between disciplines; video can be a mixture of cinema narrative and painting; 3D graphics, a coming together of sculpture and animation; sound becomes form; form manifests itself as sound. All is interactive. The technology provides an opportunity to continually experiment with the merging of forms and exploration of content. The creative potential for those artists who wish to interact with technology seems endless. And in this mutability and variety of form lies the digital arts greatest weakness. The artist is like a small child contemplating the convoluted workings of a mechanical toy, eager to play with it but frustrated by its complexity. The digital artist must embark on what seems an endless technological learning curve or rely on computer specialists to realise creative ambitions. Time and energy better spent in creative exploration.

The astonishing pace of software and hardware development, where one new function is replaced by another, more powerful function, one new piece of hardware better than the last, almost it seems, simultaneously, creates the impression that the digital artist is working on shifting sands. The artist is beholden to the computer industry, his development linked to the vagaries of competition in the computer market.

For those of us who trained as traditional painters and sculptures or designers whose individual crafts utilised relatively stable technologies, computer based art can seem an alien medium. The individual artist/designer had the opportunity to develop a personal style over time without the onerous task of relearning the basic mechanics of their chosen medium or the need to adapt their work to suit new forms of distribution. You could concentrate on developing drawing skills using limited media. The work became very personal. You developed mark-making techniques that were your own. You had intimate knowledge of your medium. Variety of form was created out of simplicity. Your skills could evolve over time and the history of this personal evolution became a visual resource. Can the computer offer a comparable artistic experience? Can an understanding of the aesthetics of programming and the creative experience of the programmer be of benefit to the artist? Should programmers draw?

Development of Craft

For most of the history of art, the visual artist has been seen as the individual most *adept* at depicting reality or imaginative states through the act of drawing, painting and sculpture.

Photography, video and computer technologies have challenged this assumption. An artist is still an *adept*, but the medium he or she uses has changed so radically that what constitutes an artist's *necessary* skill base has inevitably changed. This is a profound change. In essence, the physical skills, the co-ordination of hand and eye movement and ability to understand and control concrete medium are no longer the *necessary* skills.

New technologies create new experts and it would be foolish of the traditionally trained artist to assume that he or she can utilise creative software as a visual tool without understanding the medium. The draughtsman develops his drawing skills through intimate knowledge of medium and process. It is logical to assume that the computer artist who utilises a programming language as a creative tool will develop his skills in a similar fashion. It is my contention

that the programmer is the new *adept*. For want of a better word I will refer to this acquisition of programming skills as *craft*.

Craft is usually used to describe a skill requiring manual dexterity. It seems strange to use it to describe the development of programming skills. One notion of *craft* is that it is used to control and modify physical form, the other organises and manipulates computer data for a defined purpose.

What is of real interest here is not the medium or even the end result but the thinking processes involved. Are there any common experiences that inform both disciplines? Can a common language be found that illuminates these processes?

The creative mind

For the purposes of argument I am going to make a comparison between the traditional artist's experience of drawing/mark-making and the computer artist's experience of making a mark on the screen using a programming language. This is, in essence, a record of my own experiences of both disciplines. I am a painter/sculptor who has a working knowledge of Director Lingo and Flash Actionscript. I realise that in programming terms these languages are of limited use but it is simplicity I am interested in, so any simple language suits my purpose. I am not interested in discussing the uses that can be made of the computer to create complex visual and intellectual statements that utilise a variety of programming, software/hardware and output devices. This over-involved use of technology has distanced the artist from an understanding of process, this understanding being central to the creative act. We are not thinking creatively if we do not feel a connection between mind and process. The minimal trace on the screen best represents the mind of the programmer as the simple pencil drawing best represents the artist's mind.

The experience of drawing is analogous to the workings of our creative mind. A drawing is the paradigmatic representation of the interplay between inner symbol and conscious thought unique to the individual. The minimal drawings of artists such as Sol LeWitt, Christine Hiebert, and Agnes Martin are revelatory examples of

this fusion between concept and instinct. The creative processes that these artists are involved in when drawing are models for the computer artist. Le Witt has stated that his drawings are a systematic exploration of all possibilities within a defined set of determinants. This is a statement of creative intent that reveals the creative mind of both visual artist and programmer.

When we encounter the drawings of great artists such as Da Vinci, Rembrandt and Picasso we are overwhelmed by the complexity of form and message. But if we look beneath the veneer of complex vision and observation we can see simple variety of mark within an iterative drawing process. The art forgery is made possible by identifying and copying the recursive style of an artist. What makes these artists great is their ability to use this simple, repetitive process to build complexity of form and meaning. Mind, meaning and process are integrated in great art. Craft is intimately linked with creativity.

Any artist who has been interested in developing his or her *craft* will tell you that at some point the exercise of *craft* is an almost unconscious act. It is as if this knowledge is stored in some inner database. In the hierarchy of information stored in this database, drawing can be seen as a *primitive* data. It is the building block of every other form of mark. It contains all the component information needed to create complexity. The data is stored as variations in the quality of mark made with any particular medium on a particular surface, a unique combination of the physical gesture needed to make a particular mark with any given tool and the specific property of the chosen medium. One definition of the creative artist might be that they are individuals who are best able to retain a memory of every basic mark made and to utilise a combination of these marks in the creation of representations of inner and objective observations.

We are all drawn to make marks on a surface at some stage in our lives. These marks are undoubtedly an expression of some complex emotional state. But they can also be seen as a more prosaic manifestation of recursive brain activity. Doodles and abstract pattern making are, to some extent, symbols of our mind's delight in the

manipulation of numbers. Howard Gardner in his book on the significance of children's drawing writes:

> *On the one hand we have encountered a cadre of young children whom we have come to call patterners. These youngsters analyze the world very much in terms of the configurations they can discern, the patterns and regularities they encounter, and, in particular, the physical attributes of objects—their colors, size, shape and the like.*[1]

Gardner was interested in identifying psychological types in his analysis of the children's drawing. But it is obvious to me that he is describing an iterative thinking process that is shared by artist and programmer, whether they be extrovert or introvert, fantasist or pragmatist. Artists like programmers take pleasure in a conscious understanding of the uses that can be made of a combination of certain functions acting sequentially to change property values. Although the creative activity of the traditional artist is essentially linear in nature (a mark made is not easily removed from a layered drawing or a painting and sculpture is an unforgiving art form) they must also possess a non-linear understanding of the nature of their creation. If they are to create anything more than a series of more or less random marks they must be able to form a value driven understanding of their creative process prior to and during the creative act of making. This amounts to something resembling a programming structure. The more complete the understanding, the more complex the abstract reasoning, the more *adept* the artist. This is not intended as a description of mere technique. There are plenty of technically gifted but essentially banal artists around. Artists, whatever their chosen medium and process, must find a synthesis between the technical and expressive. Art that captures the imagination is always a balancing act.

But extracting the emotional/psychological significance from any analysis of mark making is pointless and anathema to our conception of human creativity. We all experience a mixture of satisfaction and frustration when we make a mark or mould a material for some purpose. And the programmer feels the same mixture of emotions

when he or she develops a programming structure. An aesthetic pleasure can be gained by contemplating the structure of code and witnessing a simple function in operation. This is more than pleasure gained from abstraction. The same variety of responses that the traditional artist had to the properties of soft charcoal (pleasure in the wonderful variety of mark available and frustration at the difficulties experienced in controlling the medium) can be gleaned from a programming structure. A traditional artist using sophisticated graphics software does not experience this direct connection to the medium. He or she is only concerned with end results. There exists little intimacy between artist and tool. The software structure is an unseen stranger, not considered part of the creative process.

The programmer who developed the particular algorithm used is not recognised as a creative partner. This, in my view, exhibits a limited understanding of the significance of software in the creative act. It places the artist at a distance from his or her medium.

In my research I have tried to create simple software that best realise, on screen, sketchbook drawings. Visit ww.steelreel.co.uk/spin.html for an online example of one such software drawing in action. When making the drawings, I am, to varying degrees, actively conscious of my abilities to realise the mark in terms of a recursive programming structure. But I am also making aesthetic judgements as I draw. Drawing and programming are merged in the creative process. The resulting screen version is not a complete surprise, although it is often much more beautiful and fluid than the original drawing. The concept that informs the creation of this software is that the programming, interface and output capabilities are all created within a common aesthetic structure and design logic. It is important that I am the sole creator of the software, functioning as both conceptual designer and programmer, so that I can experience, at first hand, the logic and aesthetics of the programmed mark making. Artists/computer scientists have tried to integrate some level of this kind of intention into their auto generated computer drawings with varying degrees of success. I am of the opinion that artificially generated art will never satisfy the demands of creativity and that

computers should remain creative tools rather than be seen as autonomous creators. Art is an examination of the human.

The complimentary disciplines

Drawing and programming are complimentary disciplines. It is my belief that both disciplines share a common logical thinking process which focuses on building broad conceptual structures through which complexity can be created and controlled. This logic can be understood in both concrete and abstract terms. It can be represented visually as pattern, where complex rhythm and repetition are forced to adhere to the logic of overall compositional structure. The programmer may not need a visual representation of data structure; the visual designer may not see the connection between a data structure and the balance in a visual abstraction. It is essential to any computer arts education and conceptual understanding of the computer as a creative tool that both are seen as equally valid expressions of creative thinking. Programmers who draw with code would benefit from some experience of physical mark making and traditional artists should not dismiss the importance of software code.

The computational theory of mind says that…

…beliefs and desires are information, incarnated as configurations of symbols. The symbols are the physical states of bits of matter, like chips in a computer or neurons in the brain. They symbolize things in the world because they are triggered by those things via sense organs…[2]

This is an understanding of the dynamic of human consciousness as being comparable to a computer network hardwired to our experience of the physical world. If we accept this as a valid explanation of human consciousness it is not difficult to see how programming can be seen as a reflection of human experience, software art and drawing as naturally occurring human adaptation

.

[1] Gardner H.: *Artful Scribbles.* New York, 1980, p.47

[2] Pinker S: *How The Mind Works.* U.K, 1999, p.25