Originally published in the newsletter of the Liaison of Independent Filmmakers of Toronto, 2003. The interview was conducted by Vicky Chainey Gagnon

Algorithms and Chance: an Interview with R. Bruce Elder

Perhaps there are film forms that can restore freedom to the viewer. Such forms might for example, offer spectators an array of elements that they can freely combine and recombine…... (R. Bruce Elder)

Your most recent film, *Eros & Wonder* was created using digital technology. How did you become interested in using digital technology in your filmmaking?

My interest in computers started early, and grew out of my fascination with the fact that beautiful patterns are often mathematically elegant. There is an entire field of design that explores the beauty of mathematical patterns, and I was fascinated by it; from the time I was boy, I read about Fibonacci series, and the golden mean, and logarithmic spirals -- various topics of that sort. There is an another sizable field of investigation, this one rather flaky (to be sure), known as spiritual geometry, which uses the mathematics of harmony and both an image of and a means for tuning the soul. I spent a lot of reading in that disreputable field of well. You might be surprised how many artists of the last 100 years have.

Your use of digital technology in *Eros and Wonder* is quite different, however. There you used computer technology to process digital images. I believe that you write the computer programs you use to make your films. Could you tell us about these programs?

When I decided to use digital processes in my art making, I started by studying the requisite fields of mathematics and computer science; I went back to night-school and took classes for engineers. Working with the knowledge I was able to garner, I developed a computer application that would allow me to collaborate with the machine to produce “visual compositions” -- that would allow me to use many of the same principles that I have employed in my filmmaking,, but would help eliminate subjective whim.

I developed a rudimentary application that stored a set of images into a database along with a set of image descriptors (“meta-data”) and a set of image processing algorithms. The application's function was to decide what image-processing methods to apply to the images in the database, and to apply them. At first the method for selecting the processing methods to be applied to the images was pretty simple: Images were partitioned in groups based on the similarities indicated by their descriptors, as were the image processing methods (my decision on which methods most closely resembled other methods was completely informal and subjective); the image processing methods to be applied to a reference image were chosen at random “ the operator got to approve the selection, and if he or she approved it, then the methods most similar to the randomly chosen method were applied to the images in the database that most resembled the reference image.

I used this application in a film I finished almost three years ago now, “Crack, Brutal Grief.” This way of using image processing methods in film/video production interested me enough (and, I thought, the results were good enough) that I wanted to work further on this application.

It was obvious what refinement I should introduce first: using image descriptors as I did was awkward and introduced an unnecessary subjective element that conflicted with the ideal of avoiding authorial imposition. I quickly realized the application would need to use of methods to "compute" the similarity between the two images algorithmically.

Can you elaborate on the ideal of avoiding authorial imposition?

John Cage protested against the idea that an artwork is the product of an artist's feelings, believing instead that the creative process should imitate nature in its manner of operation. Cage was among the first composers to make the use of chance operations central to his compositional processes -- and he developed a variety of aleatory techniques that allowed chance and indeterminacy to play key roles in shaping musical results. Cage insisted that aleatory operations mimicked natural processes and that by imitating the operation of natural processes, the composer could bypass his or her limiting ego and allow a larger system or set of systems to shape the work. This principle has been very important to me. The richness of Cage's writing helped make the use of aleatory techniques common among composers. The rigour of writings by Iannis Xenakis and James Tenney -- composers who, like Cage, took an interest in stochastic methods and the power of their works re-enforced this influence.

Over the past few years, I have worked on projects that explored the possibility of extending these composers' ideas to the visual domain. The initial framework for this exploration was drawn from composer James Tenney who made extensive use of measures of similarity in the analysis of music structures in his book "Meta+Hodos." I was intrigued by the possibility of developing analogous compositional procedures for working with sets of images and, in particular, by the possibility of using measures of similarity to constrain random processes.

How does your computer programme calculate image similarity?

The process takes place in a number of steps:

1) Load the "key image" or "query image" (the image for which we want to find similar images).

2) Utilizing methods of feature extraction, measure a number of features of the key image. This stage creates a "signature" for the image.

3) For every image in the database, load and generate a signature.

4) Calculate the Euclidean distance between the signature for the key image and the signatures for each of the database images. Sort and store these values -- what results is a list that shows the proximity (based on its signature) of each of the database images to the query image.

The features I used for creating an image signatures were the intensity of the image, its dominant colors, the mean and standard deviation of image's RGB values, the frequency of change in RGB values, the number of defined areas ("pixel groups") enclosed within a well-defined boundary, the compactness of the principal (i.e. largest) pixel group, the major and minor axis of the principal pixel group, its circularity and its perimeter.

The challenge was -- and remains -- to select image features and a distance function such that the resultant distance really is a measure of image similarity: ideally the distance between the

images, gauged on this metric would correspond to our subjective assessments of image similarity. Measuring the distance between two images which we judge to be alike would result in a relatively low aggregate value, while measuring the distance between two images which we judge to be quite different would result in a larger aggregate value.

How do you use these measures of similarity to help you decide what effects you will apply to images?

B: First, I wanted my program to emulate the filmmaking methods to which I have become accustomed. To this end, I formulated some loose rules that would capture some of my experience in deciding what image processing algorithms might be appropriate to images that possess a given set of features. (Examples of such rules are: if there are a large number of pixel groups in the image and there are many changes in color between adjacent pixels, then sharpening the image is not highly recommended; if the image is of very low contrast, then reducing the intensity of the image is seldom valuable; if the average size of pixel groups is large, then applying algorithms that enhance the texture of the image is a less valuable choice.) I created a program that employed a constrained random process -- the constraints based on these rules as well as on the image's signature -- to decide which image processing algorithm or algorithms would be applied to images.

The program looks at images and assesses their features, and based on what it discovers, decides which processing procedures most likely suit the image, and what procedures will be less likely (and how much less likely). Different features of an image are assigned different weights, and those features that are assigned greater weight are given a greater role in deciding which image processing methods are desirable or undesirable (and how much less desirable or undesirable). The application then chooses, by chance operations, a set of processing methods to apply to the database images.

Where would you like to take your work with this computer programme? How do you want to improve it?

B: I want to introduce better means for modeling a film- or video-maker's working methods, for capturing a filmmaker's (or videomaker's) understanding of what characteristics of the image make certain image-processing appropriate and other's inappropriate. The way I modeled one's estimation of the appropriateness of a particular method to a given image was far, far too simple. What I did was simply to imbed in the program a “seat-of-the-pants” “guess-timate” of how undesirable a certain feature made a particular algorithm. For example, having a certain property might make using given image-processing methods either “slightly undesirable,” or “moderately undesirable,” or “very undesirable” (each represented by a different weight), and more precise measures of a filmmaker's sense of the appropriateness of a method need be introduced.

I also incorporated a kludgy sort of “fail-safe” provision into the application. After applying the constraints I have described, the program selected one or more image processing methods to apply to the image, processed the image and displayed the result. The user was then asked to confirm that what he or she sees is satisfactory -- thus, instead of modeling the film- or video-maker's knowledge, I simply called upon it (and used it interactively). If the result was deemed satisfactory, the program applied a similar treatment to a set of similar images and saved the result to film.

All this needs to be drastically reworked. My “fail-safe” method of allowing the operator to interact with the program conflicts with my goal of refusing immediate authorial imposition. Further, I need to develop means to capture the “fuzzy logic” involved in these decisions. This could be done by building a learning component into the program that would enable the program to correlate the features an image possesses with the image-processing methods a particular film- or video-maker finds appropriate. Further, to make the program more flexible and better able to accommodate different ways of working, the user should be given the choice as to which sets of features, from a broader array of features than I now employ, would be relevant to determining which image-processing methods might be applied to the image.

Introducing fuzzy and neural learning into this application would have this benefit as well: the assumption that there can be standardized metric that corresponds to all users judgements of image similarity is a doubtful one -- just as it is doubtful that all film or video editors take into account the same set of features when they are creating “plastic” cuts (edits based, essentially, on the similarity of images), or even that an individual editor takes the same features into account on all occasions. Creating a system that would adapt to individual users (and, perhaps, even to individual circumstance) by being “re-trained” could allow for these variations.

Despite its current limitations, however, I believe the programme is a novel way of using image processing in film and video production. I also believe that the Cagean compositional ideas on which this application is based are rich and this makes me eager to continue to develop the project.