Return to Index

Vision Statement
By Dennis Miller

Music without instruments, music without walls, music without music?

When thinking about the direction music will take in the future, I see a natural move by composers into other media that are related to music in some ways and that can serve as additional creative outlets for their work. In particular, time-based media, such as video and animation, seem to be most suitable for this type of expansion, and indeed, many composers are now exploring these avenues.

After spending 25 years in the service of music composition, I too have recently begun to add a visual component to my work. Typically, the result is an animation that combines original music with a sequence of 3D visual images. These works, which are intended for projection in a concert venue, reflect my interest in enhancing the musical experience with an additional dimension - they are multimedia in content and multi-sensory in intent.

Unlike some mixed media works, where parameters of the music directly control aspects of the picture (or vice versa), I view the music and animation as two separate but equal streams that happen to coexist in the same time frame. My goal is to apply the same principles of continuity and development to the visual element as I do to the musical component. But how does one transpose, invert, augment, diminish, or otherwise apply "musical" transformations to a moving picture? That is the challenge of this work, and it's what makes the composition of these animations such an exciting and provocative endeavor.

A brief introduction: I first considered adding a visual element to my music after participating in a conference dedicated primarily to graphic artists working with personal computers. Each year at this conference, a small number of electronic music composers is invited to present their work. My piece was for a solo electronic keyboard player triggering samples from a Kurzweil sampler; but though the piece used massive textures swirling and panning throughout the auditorium, the performer had only a few notes to play and the audience appeared disinterested. I quickly realized that like many of today's audiences, these graphics people had not enough to occupy themselves during the performance; there was simply not enough to look at.

In an effort to enhance the work, I first attempted a collaboration with an animator on the faculty of my university. The person had a significant background in sculpture and had recently turned to computer animation as a means to realize both real, physical works he had created in the past, and new, "virtual" works that could only be rendered on a computer. Unfortunately, the grand total of his output over nearly a two-year period was 90 seconds - strikingly beautiful though it was. Alas, we were unable to create any "concert length" pieces that could be submitted for performance, which was my ultimate goal.

Subsequently, I decided to explore the visual world on my own, and in 1998, after evaluating several commercial applications, I stumbled upon the POVray graphics compiler. I was struck immediately by its remarkable similarity to Csound and felt it might be an excellent platform on which to begin my explorations. POVray, which can best be described as a "scene description language," offered the type of flexibility and control that I needed and used a simple command syntax with which I felt completely at home.

For example, to build a scene, one starts with a camera, which must have a location and point on which to focus. Like other elements in a scene, the location of the camera is specified as a 3D vector: <0,0,0> is the center of the universe. Then, lights are added, a sky or plane or some virtual backdrop is designed, and the principal visual elements - boxes, spheres, cylinders, cones, etc. - are placed in the scene. Every element must have a pigment at a minimum, but can have a highly elaborate, morphing texture, complete with a finish on its surface (metallic, for example) and a surface normal (bumps, dents, wrinkles, et al).

Every characteristic of every element, including the camera itself, can be varied over time by running against the internal clock. For example to move the camera from some starting point to a position 10 units above that point, one would type:

camera <0, 0 + (clock *10), 0>

Even the clock itself can be accelerated and decelerated, much like the Tempo statement in Csound, though not necessarily in a linear fashion.

Like Csound, conditional statements can be employed. To place a large number of boxes in the scene, each "transposed" by a successively larger amount from the next, one could use the following code:

```
#declare r1 = seed(13);                 //translates y position
#declare xcounter = 0;                  //increments x position
#while (xcounter < 50)
#declare zcounter = 0;                  //increments z position
#while (zcounter < 50)
box {<-.25,-.25,-.25>, <.25,.25,.25> //coordinates for outer corners

        translate <xcounter, rand(r1), zcounter>
        pigment { color rgb <1,1,1> } finish {metallic}
}
#declare zcounter = zcounter + 1;
#end
#declare xcounter = xcounter + 1;
#end
```

As the x counter advances from 1 through 50, the box will be moved one unit farther to the right (X-axis). For every increment of x, 50 new positions for z will be defined via the zcounter (see Fig. 1). The value used to offset the box on the Y-axis is random; if animated, the randomness could change throughout the segment.

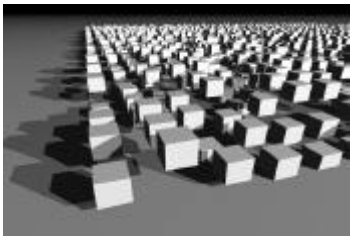

Figure 1 Use of a #while statement in POVray

Mathematical surfaces can be defined easily using the isosurface function found in the unofficial patch known as MegaPov (www.nathan.kopp.com). For example, a sphere is defined as

function { x^2 + y^2 + z^2 - 1 }

"Sphere" is a built-in function in POVray, however, so simply typing

function { "sphere" <1> }

would be allowed. The <1> specifies the radius of the sphere. There are many dozens of other predefined surfaces, any and all of which can be manipulated in a vast number of ways.

Morphing isosurfaces are the basis for every image found in the work I will present tomorrow. A still of the principle image in the work is shown in Figure 2, along with a complete listing of the code used.

```
#declare Vortex = function {((y*clock) + (z*(1-clock)))+(1/(sqr(x)+sqr(z)+(1-clock
*.90)))  // y+(1/(sqr(x)+sqr(z)+0))+0.3 +sin(atan2(x,z)*(7*clock) + ((x*clock*10)
+(y*10* (1-clock))))*1.25 // +sin(atan2(x,z)*(7*clock) + ((x*clock*10) +(y*10* (1-
clock))))*1.25 }
```

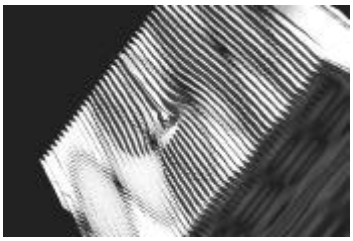
Figure 2 An image created with the isosurface function and the code that it uses
I would be untruthful if I did not admit that this image resulted from considerable trial and error.

POVray's camera can be manipulated by using a normal statement. This effect acts somewhat like a "3D filter" and can produce startling results. For example, the image shown in Figure 3 is a rendering of the scene in Figure 1 but with a normal added to the camera statement.

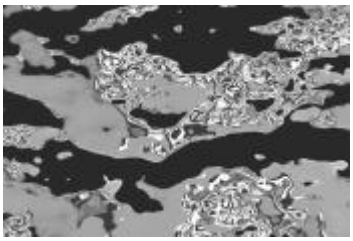
Figure 3 A normal added to the camera

This is one of the techniques I explore in my newest work. I am also employing the image map feature a considerable amount, which allows me to use one or more preexisting bitmaps in a scene (shades of "soundin.") The images can be mapped onto POVray primitives as textures and can be manipulated using the exhaustive processes the compiler offers.

Second Thoughts
In *Second Thoughts* (2000), an 8-minute animation with original music, my goal was to explore the inside of a virtual object with no landmarks or signposts available to provide the viewer with an initial perspective or context. What is the scale of the image? Unknown. What are its size and dimensions? Determined at the viewer's discretion. Like a truly abstract piece of music, the non-representational visual elements would be identifiable only through their physical characteristics; color, most readily, but also shape and texture. The viewer's perspective, whether it be considerably back from the object, a few inches from it, or even inside the thing itself, is also a trait that is developed.

Visual "motives" recur throughout the work. Most prominent among these is the color red. Different shades and hues of red are used in the vast majority of scenes, and one of the climactic moments of the piece is a long morph between a swirling, decomposing image and the recapitulation of the opening material. The final moment in this morph occurs when the color of the decomposing image matches the pure red of the opening. A black, vertical bar that scrolls across the screen in the opening segment of the work also recurs in various guises. For example, at around 1:30, we see broad vertical bars of red,

gold and black scrolling in a similar fashion, and at other times, the vertical bar becomes a horizontal one, or is skewed into a curved image.

There are also shapes, all created using isosurfaces, that become referential in the work. For example, the image shown at the top of Figure 4 appears initially in a vertical orientation. Later in the piece, it becomes the central image in a dense scene where it is horizontally oriented and repeated hundreds of times (shown in Figure 4, bottom).

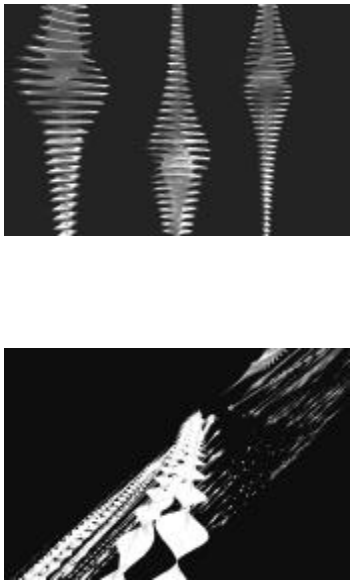

Figure 4 A recurring shape (with modifications) in *Second Thoughts*

The object moves on a vertical and horizontal path in its first appearance, and later shoots into the scene along the Z axis, where it also exhibits a "flocking" behavior.

Yet another recurring idea is the interplay of dimensions. All of the opening scenes use only two dimensions, and the first climactic moment occurs when the camera pulls back from its orientation to reveal the 3D isosurface created with the "vortex" function shown above. The viewer then discovers that all the images seen to that moment have been from the inside of the 3D isosurface, surely a startling revelation! From that point on, 2D and 3D images alternate until the work closes. (An analogy to this interplay in the visual realm is the interplay of spatial qualities in music, for example, moving between a "wet reverb" and a "dry" quality.)

Summary

Animations are one of the most fruitful areas of exploration for composers as they can require no additional collaborators. Unlike a choreography, and of course a live musical performance, animations can be done solely on the desktop, and give the composer an additional element to employ in their work. Without much effort, composers can map

musical parameters to visual ones, though in my experience, that particular method of associating the two media does not often produce stimulating or provocative works, and often degenerates into predictable cliches.

Undertaking the study of a language like POVray gives the composer a vast new realm for creative endeavor. Anyone familiar with Csound or even basic tents of programming will find the language to be quite intuitive. An enormous newsgroup exists to support those starting out, and massive numbers of tutorials and examples, in addition to a thorough and up to date user's manual, also provide support.

I anticipate that more composers will begin to incorporate other media in their work. Though I have not given up on the concept of writing the perfect piece - one that is so compelling and engaging that audiences will need no visual cues, no performer to maintain their interest - I believe we musicians are at a tremendous disadvantage given the constant visual stimuli that bombards modern society, and the expectations audiences now have as a result.

It is my hope that other composers will bring a musical perspective to bear in time-based media such as animations and video. A composer's experience and sensitivity in dealing with questions of timing, pacing, continuity and form is often missing from many of the abstract animations one sees today, perhaps because many animators began life as still graphic artists or even sculptors. This cross-fertilization will benefit the field of visual arts, leading to new and better-structured works, and will no doubt give the composer new aesthetic challenges, not to mention new performance opportunities.