Creating Three-Dimensional Computer Animations Using Spectral Data and OpenGL

Tim Kreger
Australian Centre for the Arts & Technology
Institute of the Arts
Australian National University

Abstract

The author has been exploring the mapping of spectral data to the visual domain for artistic purposes. The technique involves analysing sound and music using Fourier techniques and using the data to control an OpenGL animation environment. The result is an "automatically" generated animation which is synchronised with the original sound file. This gives the composer of electroacoustic music a relatively quick and simple way to produce "video accompaniments" to their work should they wish to. The process raises many interesting problems and issues about the perceptual relationship between music and image. These issues will be discussed and examples of the work-to-date will be presented.

1 Tools

1.1 OpenGL

OpenGL is fast becoming the standard API for three-dimensional graphics development as it provides a streamlined, hardware-independent interface to graphics hardware on most commercial platforms. The programmer requires no knowledge or understanding of the host machine's graphics hardware as the OpenGL libraries on that machine provide the necessary optimisation and hardware calls to ensure the graphics are rendered as quickly and as efficiently as possible. The Macintosh version of this software actually uses MacMesa, a freeware 3-D graphics development API which is syntactically identical to the OpenGL libraries and is available on most popular platforms.

1.2 FFTW

FFTW was used for analysis as it provides a quick, easy to use environment for doing Fourier analysis. FFTW doesn't require the number points in the analysis to be a power of two which can be useful in the context of visualising sound data. The libraries are available on most popular platforms and are licensed under a GNU licensing agreement.

2 Process

The technique employed involves analysing a sound file using the FFTW library and using the magnitude values of each bin to control primitive objects in the OpenGL animation. One analysis is done per frame of animation. The number of analysis points is arbitrary and determines how many primitive 3-D objects will appear in the animation. Some issues arise when the frame size of the analysis is significantly smaller than the sample rate divided by the animations number of frames per second as important sonic events may be missed by the analysis and hence not appear in the animation. In practice this seems not to be too problematic as the ear-eye-brain percept is not acute enough to notice these inconsistencies.

On each analysis and visualisation the visual frame is stored into a SGI-type RGB image file which is later compiled into a Quicktime, AVI or whatever movie format is required. It should be noted that the sound is resynched to the image in post-production. This approach was taken so as to increase the flexibility of the process in the context of producing animations.

2.1 Mapping

Although the process is relative simple, much thought has gone into how the 3-D objects are placed in the space of the visual domain. Initial experiments use the magnitude parameters of the analysis to control the
radius of a sphere, one sphere per analysis band. The
spheres are laid out along a line sequential in
accordance with their position in the analysis. This
approach works well in terms of seeing spectral
activity but is visually quite boring. Also if one uses
a large analysis frame then it is difficult to see all of
the spheres due to the limited real-estate of the
graphics screen. The spheres will either be too small
to see any detail or large enough to see detail but not
enough of the spheres can be seen so the entire
spectrum is not visible. This is the area which is
recieving the most research at the moment and will be
demonstrated during the session.

3 Software Parameters

3.1 Input Settings

The input settings are used to set the input sound file
and the number of bands in the analysis. The sound
file must be a raw 32 bit floating-point file.

3.2 Amplitude Settings

The amplitude parameters are perhaps the most
important for achieving a meaningful animation. These
parameters determine how the analysis information
will be used to control-the primitives.

The Amplitude Scalar is multiplied by the magnitude
component of each of the bands in the analysis so that
the primitives will appear at the optimum size for
viewing, this is very different for each sound file and
takes some time to get right for each sound.

The Amplitude Threshold is used to reduce visual
noise and rendering time by making sure that only
spectral components with amplitudes above this value
are drawn.

The Amplitude Clamp is used to limit the upper size
of the primitives. Without this parameter some
primitives will overly dominate the image, particularly if the Amplitude Scalar is high.

3.3 Texture Settings

The texture parameters determine whether the
primitives are to be texture maps and if so how the
texture is to be applied. The texture has to be an RGB
image file. One can use an animated sequence of
RGB's as the texture which make it possible to use
dynamically changing textures.

There are three texture modes in the OpenGL
environment:

Object Linear is when the texture is mapped onto the
object so that it moves with the object surface.

Eye Linear is when the texture remains in a fixed
position with respec to the viewer.

Sphere Map is when the a spherical model of the
texture is mapped on the entire environment. This is
useful for simulating reflective surfaces.

3.4 Camera Settings

The camera settings are used to position the camera
within the space which is useful for positioning the
camera in the optimum space for each animation. It is
hoped that these parameters will be scriptable at a later
date so that the camera can be animated.

3.5 Light Settings

The light settings are used to position the light with
the space. This parameter needs to be used in
conjunction with the camera position so that the model
will appear in the rendered image.

3.6 Render Settings

The render settings are where the frame rate, render
range and name of the output files are set. The frame
rate is useful for running quick tests at lower frame
rates to ensure all other parameters are set correctly.
The render range facility makes it possible to produce
animation in smaller chunks enabling one to make
animations from larger sound files when disk space is
an issue.

The software outputs a sequence of RGB images with
names in the format filename.XXX where XXX is
the number of that file with respect to the sequence.

4 Conclusion

The process is so far quite a simple one and the author
hopes to develop this framework into a larger more
flexible enviroment for creating animations. It is
envisioned that it will provide a tool for composers to
explore the possibilty of producing visual
accompaniments to their music or sound work.
5 Acknowledgements

The author would like to thank Larry Polansky and the Bregman Electronic Music Studios at Dartmouth College, USA for providing the opportunity to carry out this work. The majority of this work was done during a residency at the above mentioned facility.

References
