Interactive Visualisation of Musical Form using VRML

Alan Graves*, Chris Hand* and Andrew Hugill†

** Department of Computer Science*

† Department of Visual and Performing Arts

De Montfort University, The Gateway, Leicester, UK. LE1 9BH

***e-mail**: arg@dmu.ac.uk, cph@dmu.ac.uk, ahu@dmu.ac.uk*

This paper describes the design and implementation of a virtual environment which provides an interactive visualisation of MIDI music data. 'Benediktine Cyberspace' and composite dimensions are discussed, along with a description of how visualisation techniques derived from them were used to overcome some of the difficulties associated with representing large numbers of data attributes simultaneously.

Keywords: music, visualisation, spirals, MIDI, VRML, synaesthetic metaphor

1. Introduction

Computer-based tools for musicians typically work at very low levels and provide little opportunity to visualise the overall structure of a piece. Music has many attributes which change over time, and so represents an interesting domain in which to explore visualisation techniques. The work described here aimed to produce a virtual environment (VE) in which the materials and structures of a musical composition – at both micro and macro levels – are made explicit and manipulable as objects. The system is intended as an aid for composers in the realisation of their ideas, particularly in relation to form, and as an analytical and educational tool for the exploration and understanding of existing works.

This paper describes the design and implementation of a VE which presents an interactive visualisation of musical data supplied in standard MIDI (Musical Instrument Digital Interface) format. A prototype, known as MidiVisualiser, was produced using Silicon Graphics Inc's Cosmo Player VRML97 plug-in with Netscape under Windows95, making use of the VRML External Authoring Interface (EAI) as well as code written in Java and JavaScript.

The structure of the paper is as follows. Section 2 discusses relevant previous work on visualisation and music, while section 3 describes in detail how principles of 'Benediktine Cyberspace' shaped the design of the prototype and the visualisation techniques used. A brief overview of the implementation and evaluation are presented in sections 4 and 5, while section 6 discusses future work and conclusions.

2. Background

2.1 Visualisation Techniques for Complex Information Spaces

One problem inherent in the visualisation of music is the sheer number of data attributes which may be presented to the user, such as pitch, timbre, velocity and harmony (our list drawn up during early stages of this work included some 26 properties). A number of techniques presented in previous research into VEs suggest some guidance upon how to approach this problem. Feiner and Besher's (1990) 'Worlds Within Worlds' technique allows a potentially infinite number of information dimensions to be represented, although only a limited number are visualised at any one time. Cowperthwaite *et al*. (1996) propose a method for 'viewing in context' by warping space along the user's line of sight. If the importance of context is adopted as a design principle for data visualisation VEs, it would tend mitigate to some extent against unfolding techniques. Cowperthwaite's work does not suggest a suitable technique by itself but its rationale serves to strengthen the case for using context-preserving techniques. Drew and Hendly (1995) present a technique which involves enclosing the low-level structure of data in capsules which become transparent when the user requires. This represents a seamless technique of traversing levels of information whilst preserving context.

2.2 Previous Work on Visualising Music

Fakespace Music[1] is an immersive VR system which runs on SGI machines using a stereoscopic head-mounted display for entertainment purposes, using music to drive virtual environments described as "immersive music videos". One such creation is "Vacuii" by Christian Greuel which immerses the viewer in a Virtual Environment in which objects are animated in response to cues extracted from the music. As opposed to other visual experiences which are generated solely from existing music, such as Cthugha[2] or Bomb[3], Fakespace Music environments are themselves designed and then programmed to react to musical input.

Malinowski's Music Animation Machine[4], intended mainly as an educational tool, is a two dimensional graphics program which aims to allow the structure of music to be understood by people who lack any understanding of conventional musical notation. Instead of a conventional score the Music Animation Machine displays an "animated score" composed of coloured bars which represent the notes. These bars scroll across the screen in time with the music, lighting up as the corresponding notes sound.

The Structured Event Editor is a 2D/3D music visualisation system described by Kunze and Taube (1996). Some 2D and 3D visualisations based around the fundamental mappings of time and pitch to x and y co-ordinates are described, however these only deal with single

---

[1] <URL: http://www.fakespace.com/index.html>

[2] <URL: http://www.afn.org/~cthugha/>

[3] <URL: http://www.cs.cmu.edu/~spot/bomb.html>

[4] <URL: http://www.well.com/user/smalin/mam.html>

instrument visualisations. Kunze and Taube's work is interesting for the future areas of research which they discuss, for many of these are explored in the research described here. These include the ability to manipulate, as well as visualise music data, and the role of animation.

2.3 MidiVisualiser

The research presented in this paper, embodied in a prototype application known as MidiVisualiser, differs from the examples described above in a number of ways. Unlike Fakespace Music and other systems geared towards entertainment or performance (*e.g.* Tod Machover's *Brain Opera* [5] or Jaron Lanier's *The Sound of One Hand* [6]), MidiVisualiser is intended as an analytical tool, either for educational purposes or as a compositional aid.

Many of the ideas seen in the Music Animation Machine are also seen in MidiVisualiser. The emphasis on animation and the method of conveying pitch information makes them seem quite similar on some levels, however the Music Animation Machine is not a VR system and does not allow the manipulation of music. The Structured Event Editor is the most similar system overall. It places a strong emphasis on musical analysis as does MidiVisualiser, although many of the features implemented here are only suggested or proposed by Kunze and Taube.

One of the aims of MidiVisualiser is to augment the visualisation by using animation alongside the play-back of the music. Malinowski claims that by watching the scrolling representation in-time with the music, people who lack formal musical training can realise the "inner structure" of music. Baecker and Small (1990) emphasise the under-used nature of animation in applications and distinguish between a number of different uses of animation; MidiVisualiser uses what Baecker and Small refer to as "animation of process". The fact that animation enhances the viewer's experience is not surprising as it adds dynamic information to the static visualisation. Brown (1996) describes the technique of using movement, or "changing form over time", in VEs as generating a fourth dimension of a visualisation. (Form changing over time is, of course, a fundamental aspect of music.)

To categorise MidiVisualiser within this context, it is a desktop VR system which allows the analysis of musical pieces, and the alteration of those pieces through direct manipulation with the virtual representation. Although animated display of the visual representation is provided in MidiVisualiser this is intended as an analytical tool rather than being for entertainment purposes.

3. Representation and Visualisation

The two types of information represented in MidiVisualiser are macro-level (structural) and micro-level. Structural information is typified by characteristics such as the density of notes in a passage, or the coherence of a run of notes as a distinct entity. Micro-level information may be data such as the pitch of a single note, or even a representation of a wave-form analysis upon the instrument sound. So we must represent these two classes of properties in ways which reflect their level of relevance, while also considering the ease of manipulation of these properties. An attempt has been made to satisfy these two demands by using a single visualisation technique.

---

[5] <URL: http://brainop.media.mit.edu/>

[6] <URL: http://www.well.com/user/jaron/vr.html>

Below we describe how a classical Benediktine approach to the problem would involve mapping each musical attribute to an intrinsic or extrinsic dimension. The current implementation of the application extracts eight 'raw' musical properties from MIDI data. These raw properties may be distinguished from more abstract properties as they are readily available from the data file without any real analysis (which must be driven by musical domain knowledge). These properties are the start time of the note, its length, stereo placement (pan), volume, instrument, pitch and key on/off velocity.

It is relatively straightforward to populate a VE with a set of objects representing notes with these properties, for example using the arbitrary shown in Figure 1. On studying this mapping one might ask whether properties such as 'key on velocity' should be visible at this level. Whilst it is evident that some dimensions (particularly the extrinsic ones) are more apparent, and therefore implicitly more dominant than others, all dimensions are effectively equal as they are all continually outwardly visible. More generally the problem is one of how to incorporate both macro and micro-level structure in the same framework at the appropriate places. Also, with a number of musical attributes mapped to such obscure intrinsic dimensions such as 'orientation in Z dimension', it is probable that this approach has exceeded a threshold of intuitive intelligibility.

Spatial Dimension	*(maps to)*	***Musical Property***
X-Z dimension	↔	Beginning time of note
Length along X-Z dimension	↔	Length of note
Orientation in X dimension	↔	Key on velocity
Orientation in Z dimension	↔	Key off velocity
Y dimension	↔	Stereo placement
Colour	↔	Pitch
Shape	↔	Instrument
Size	↔	Volume

Figure 1: Arbitrary mapping of spatial dimensions to musical properties.

3.1 Benediktine Cyberspace

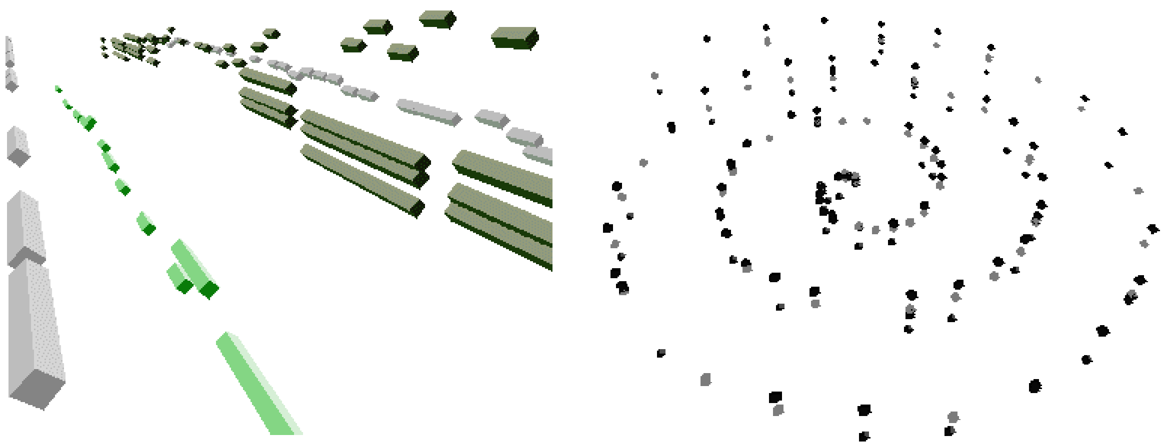
The approach taken here towards visualising musical data is to ground it in a wider context of VR visualisation. The most generally applicable concept for the application of VR to visualisation techniques is proposed by Benedikt (1991) and is therefore often referred to as 'Benediktine cyberspace'. This is founded on the difference between Euclidean space (which is defined in terms of zero dimensional points at given locations) and objects which exist at a location in space, but whose tangible characteristics are regarded as other dimensions.

Objects in Benediktine cyberspace exist in extrinsic dimensions (x, y, z co-ordinates) and in intrinsic dimensions (colour, size, shape, *etc*.). As such, an apparently three dimensional object can be thought of as having many more than three dimensions. A series of objects representing items of data may be clustered in 3D space, with each of the three dimensions being mapped onto various attributes of the items. Further information can then be meaningfully encoded into the space by using intrinsic dimensions such as colour and size. The user would still be able to identify the structure of the data relating to the other dimensions as well as the spatial structure. Benedikt claims that a large number of dimensions can be meaningfully represented to users in this way. Some of the issues arising from the use of the Benediktine approach in this research are discussed in the remainder of this section.

3.2 Composite Dimensions

A common way of visualising musical data is the 'piano roll' mapping, which maps time on to one dimension and pitch onto another. The use of these two target dimensions results in displays which are visually intelligible and which can correlate with our natural metaphorical interpretation of musical form. Given that at all times the user has a specific location within the VE (the avatar's position), the user may move backwards or forwards through the piece. However, such mappings tend to produce visualisations which appear very long and thin (see Figure 2(a)), making poor use of space in the VE and requiring the user to spend much time navigating in order to view objects which are distant or occluded by other objects. When considering MIDI-based music this is due to the fact that with the exception of the time of a MIDI event, all other parameters expressible with the format are finite and generally have only a small number of possible values (usually 0-127).

An interesting possibility concerning Benediktine Cyberspace is the 'composite dimension', a number of visual dimensions which have been collapsed into one effective information dimension by a mapping function. In order to make better use of 3D space, a mapping function may be created which derives both the x and z co-ordinates from the beginning time of a note. The dimension of time in the data stream is therefore represented in the VE by a 'composite information dimension'; each absolute time in MIDI clock ticks maps to a unique point on the X-Z plane. The function chosen here to implement this mapping was based on the Archimedean (or linear) spiral shown in Figure 2(b).



(a) Linear Visualisation (b) Spiral Visualisation

Figure 2: Two possible visualisations of MIDI music data.

Taking the centre of the spiral to be $t=0$ the visualisation may now expand outwards to occupy the XZ plane, just as the previous mapping would have allowed the visualisation to extend outwards along the Z dimension. The remaining spatial dimension, Y, is then available to have a musical attribute mapped onto it. As this will almost certainly be a value in the range 0-127 it is possible to visualise the spatial structure which results from all possible (t, y) pairs. The T dimension which is curved in two-dimensional space (on the XZ plane) becomes extruded at a normal to this plane in the Y dimension. The resulting conceptual 2D plane curved in 3D space is shown in Figure 3.

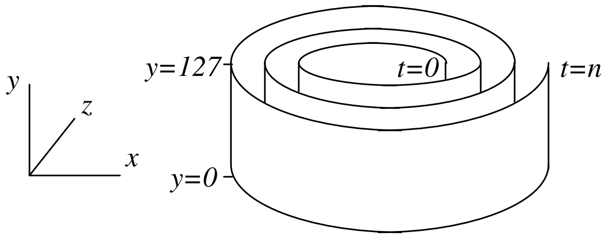


Figure 3: A visualisation of the t-Y plane which results from an extrusion of an Archimedean spiral along the Y axis.

This mapping makes much better use of 3D space and allows the user to view the musical structure from above (looking down onto the XZ plane) with few problems of occlusion. This would not be the case if this principle of composite spatial dimensions were taken to its ultimate conclusion and a space-filling curve function (Sagan, 1994) were used in order to map the one dimension of time into the form of a filled 3D cube. Whilst such a structure would make optimum use of the 3D space, both navigation and interpretation of the structure would be virtually impossible.

On the other hand, the shape of the Archimedean spiral as the backbone of the visualisation is further validated by its reminiscence to the grooves of a vinyl record. These allow listeners to place the stylus at the beginning of a required track because the representation of the sound is actually visible on the surface of the disc. Likewise, quiet portions of a song can be visually differentiated from other sections. (The fact that SGI's Cosmo logo uses a spiral galaxy is entirely coincidental.)

3.3 Benedikt's Principles

Benedikt proposes two fundamental principles to adhere to when implementing this form of virtual environment: the Principles of Exclusion and Maximal Exclusion:

1. The Principle of Exclusion

Two non-identical objects having the same extrinsic dimensions and dimension values, whether at the same time, or including time as an extrinsic dimension from the outset, is forbidden, no matter what other comparisons may be made between their intrinsic dimensions and values.

2. The Principle of Maximal Exclusion

Given any N-dimensional state of a phenomenon, and all the values - actual and possible - on those N-dimensions, choose as extrinsic dimensions - as "space and time" - that set of (two, or three, or four) dimensions that will minimise the number of violations of the Principle of Exclusion.

The first principle means that no two different objects may occupy the same point in space, and the second that the extrinsic dimensions should be chosen in order to separate different objects as much as possible. Benford and Mariani's (1994) use of multi-user VEs to facilitate the sharing of databases liken extrinsic dimensions to the key attributes in databases; no two objects may share the same extrinsic dimension values (key values), even if their intrinsic values are equivalent.

This has an implication on the mapping functions described above. Given that the spiral-shaped TY plane allows values to be located at any (t, y) point where y is another musical attribute, it is not possible to satisfy the Principle of Exclusion. Consider the following notes:

Note 1	Note 2
t: 200	t: 200
instrument: 20	instrument: 20
pitch: 82	pitch: 78

Notes 1 and 2 represent a chord; two notes played at the same time, with the same instrument at different pitches. The attributes mapped onto the extrinsic values are time and instrument, but this is insufficient to disambiguate the notes on the TY plane. However, consider the case where the pitch attribute is mapped on to the Y co-ordinate and the following situation arises:

Note 3	Note 4
t: 200	t: 200
instrument: 35	instrument: 20
pitch: 82	pitch: 82

Here notes 3 and 4 represent the same notes played on different instruments at the same time. Once again the objects representing these two notes will occupy the same point in space. Fortunately, because the instrument and pitch attributes both only require values in the range 0-127, it is possible to generate a mapping function which can elevate notes above the TY plane. This effectively generates a (t, y, u) co-ordinate system where $t \in \{0..+\infty\}$, $y \in \{0..127\}$ and $u \in \{0..127\}$ (see Figure 4). It is only possible to use this co-ordinate system because u values have known limits and therefore the mapping can ensure that each (t, y, u) co-ordinate resolves to a unique position in (x, y, z) space which is displayed to the user.

There are a number of other parameters in the form of scalar values which control the overall shape of the spiral and the placement of objects within its co-ordinate space. By parameterising the function (t, y, u) → (x, y, z) with a number of values which may be altered dynamically, the user may manipulate the form of the visualisation at runtime in order to generate the required spiral form which most aids visualisation. These values include the number of seconds represented by each revolution of the spiral, the height and the gap between revolutions of the spiral.

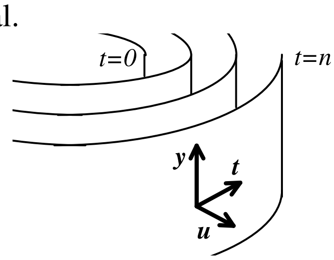


Figure 4: The (t, y, u) co-ordinate system.

By using musical domain knowledge it is also possible to utilise the property of angular rotation of a point on the spiral. Just as closeness in XYZ space implies some form of close relationship between two points, in the spiral form described above closeness in the XYZ co-ordinate space (*i.e.* the result of a similar number of degrees of rotation) can be interpreted by the user in addition to closeness in the TYU co-ordinate space. It is therefore possible to synchronise the angular rotation of the spiral to the time signature of the piece of music, such that, for instance, each first beat of a bar appears at the 0° point on the spiral. This effectively gains the user another spatial visualisation dimension, however, what it represents must be chosen carefully as the relative positions of two points in terms of their angular rotations are tightly coupled to their more orthodox positions in the TYU space. The synchronisation of time signature to angular rotation is only one possible visualisation option (this was not used in the prototype system).

We feel that spiral representations are under-exploited in the field of data visualisation in VEs, with the possible exception of the “Document Spiral” (Cugini *et al*, 1996). Spirals make much better use of 3D space in situations where one information dimension is infinite whilst the others are bounded (*i.e.* essentially some stream of information, either temporal, as in this instance, or some other form of ordering). Additionally, where the data may be regarded as having some repetitive aspect, the use of a spiral may provide a way of viewing the data in such a way that the repetitive aspects can be seen whilst retaining the sequential nature of the data.

3.4 Size as an Intrinsic Dimension

One of the obvious intrinsic dimensions to use as a target for a number of possible source attributes is an object’s size. In the musical visualisations described here it would seem highly advantageous to map the musical property of volume onto an object’s scale. The mapping between large objects and loud sounds seems highly intuitive and in-line with general synaesthetic and metaphoric concepts (Classen, 1993; Day, 1996). This mapping is logically reinforced by the fact that an object which has zero scale is invisible, which correlates with the fact that a sound with zero volume is inaudible.

However, object size is not as useful a target dimension as it might first appear. In fact Benedikt (1991) warns against its use, partly because larger objects (either in all dimensions, or along one dimension) can tend to occlude other objects in 3D space. This occurs when one object’s location in extrinsic space falls within another object’s volume as defined by the spatial intrinsic dimensions which are also expressed in extrinsic space. Effectively a conflict arises because pairs of information dimensions are forced to share the same perceptual substrate in order to be expressed. Even with objects of a constant size it is possible for objects to partially occlude one another, for if they are to be useful the objects must have some volume unlike Euclidean points. However, with objects of variable sizes it is harder to limit the effects of overlapping objects.

A further reason for abandoning the use of scale as an intrinsic dimension rests upon a more practical consideration. When using VR systems with non-stereoscopic displays the viewer relies upon a number of monoscopic depth cues for locating virtual objects in 3D space, many of them movement-related cues such as motion parallax and kinetic interposition. A great deal of work performed within MidiVisualiser will entail the avatar remaining motionless while operations are performed on nearby objects, hence motion-related depth cues will not be available for a great deal of the time, possibly resulting in objects being ambiguously interpreted as both small/close and large/distant unless they are all of a uniform size.

3.5 Visualising Tonality

The system’s default function for generating the colour of notes is the *tonality mapping*. This function generates a value representing the tonality of a single note with respect to the key of the currently loaded piece. The MIDI file specification already defines a meta-event for representing the key signature of a MIDI file (or a ‘song’ within a MIDI file) and if one of these events is found within the file this is used, otherwise the user must input the key signature.

The tonality mapping currently implemented uses an orthodox interpretation of tonal relationships – a cycle of fifths (Malinowski’s Music Animation Machine also uses a colour coding system based on this concept). A function is used which maps from the number of semitones above the key of a particular note to a value between 1 (tonal) and 0 (atonal) representing the degree of tonality as prescribed by the cycle of fifths. This value is then used to calculate the surface colour of the virtual note. The mapping function also embodies a similar sequence of tonal relationships for calculating the tonality value of notes when the key signature is minor (*i.e.* a simple transposition of the cycle of fifths sequence into a minor scale).

Experience of using the MidiVisualiser prototype has highlighted the technique of using colour to represent tonality as having possible applications in the tuition of music theory. From the authors' personal experiences, one area which novice musicians find hard to understand is the structure of chords, and relating actual fingering positions to the sound which results. Using the mapping described above, different types of chords appear to have quite different colour characteristics in the VE, unlike traditional notation. More importantly,

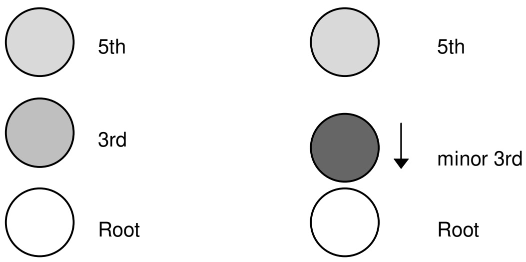


Figure 5: Interactive visualisation of tonality. By dragging a 3rd note downwards by a semitone the overall colour of the chord changes.

by looking at a chord it is actually possible to identify the notes which are responsible for the chord sounding major or minor. Also the interactive nature of the tonality analysis is advantageous as it allows the structure of a chord to be altered and the results immediately visualised *and* heard (see Figure 5).

3.6 Quantization Visualisation

An alternative musical analysis function which may be selected to provide the colour of notes is the *quantization mapping*. Just as the tonality mapping generates colours from the key signature and the pitch of individual notes, the quantization mapping generates colours from the time signature and the beginning time of individual notes. In fact there are a number of quantization mappings to allow the piece to viewed in relation to a number of different timing resolutions – those supported in MidiVisualiser are quarter, eighth, sixteenth and thirty-second notes.

The function which maps the timing of a note to a specific colour uses as its inspiration the astronomical phenomena of red-shift and blue-shift. When a note occurs exactly on the given quantization value it is colour coded white. The earlier the note is located before the quantization value the more its colour is shifted towards blue. Conversely, the later the note is located after the quantization value the more its colour is shifted towards red. This is mapping is in fact made up of three functions, one each for the red, green and blue components of the colour.

When pieces of music are viewed with a quantization mapping selected, series of notes which are 'mechanistic' in nature (*i.e.* play exactly on the beat) will appear predominantly white. However, pieces which have a more human feel (*i.e.* play around the beat) will be characterised by different shades of blues and reds.

4. Implementation

This section supplements the theoretical discussions above by providing a very brief overview of the practical programming issues encountered during implementation. Two alternative views of the software system are shown in Figure 6 – the system is described in this section from the first of these two perspectives.

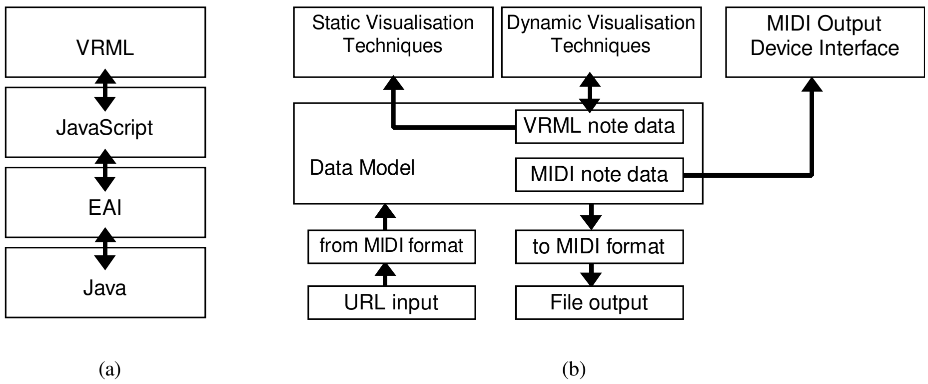


Figure 6: Two views of MidiVisualiser's software architecture. (a) the programming architecture underlying the system, and (b) a functional view of the main conceptual components.

The use of the VRML External Authoring Interface (EAI) for controlling the behaviour of a VRML world generates many options for the centralisation of control in an application. It is quite possible to use a VRML browser simply as a rendering engine for use by a Java application communicating through the EAI, although Marrin *et al.* (1997) point out that there are a number of disadvantages to this approach and advocate a distributed model with functionality divided between Java applets and VRML prototype/script nodes. The use of Java applets for high-level control is encouraged, with low-level behaviours being delegated to VRML scripts. An approach similar to this is adopted here, although with a slightly different emphasis.

The EAI proposal is still at the Working Group stage and current implementations are relatively immature. On the other hand the JavaScript facilities within CosmoPlayer are assumed to be far more stable and many of the features provided by the EAI can be duplicated by JavaScript. In order to reduce the amount and complexity of the EAI workload in this application a great deal of functionality has been allocated to a main JavaScript Script node. The main Java applet holds a reference to this node and does not manipulate the VRML scene graph or event network itself; these are manipulated entirely by the JavaScript portion of the application. For example, when creating a note object, the EAI performs the `createVrmlFromString()` call which returns a reference to a `vrml.external.Node` object. This node is then passed through the EAI to a function in the main script which inserts that node into the scene graph and links it up with event routes so that the node can communicate back to the main script. This architecture also greatly reduces the potential storage requirements of the Java applet due to the ability of a JavaScript expression to write directly to exposedFields of other nodes. If the EAI were to be used to modify the exposedFields of note objects directly (translation, rotation, scale, diffuseColor, transparency, *etc.*) it would require the instantiation of EventIn objects for each manipulable field. In order to reduce performance overheads these would ideally be permanently stored, increasing the storage required for every single note in the application.

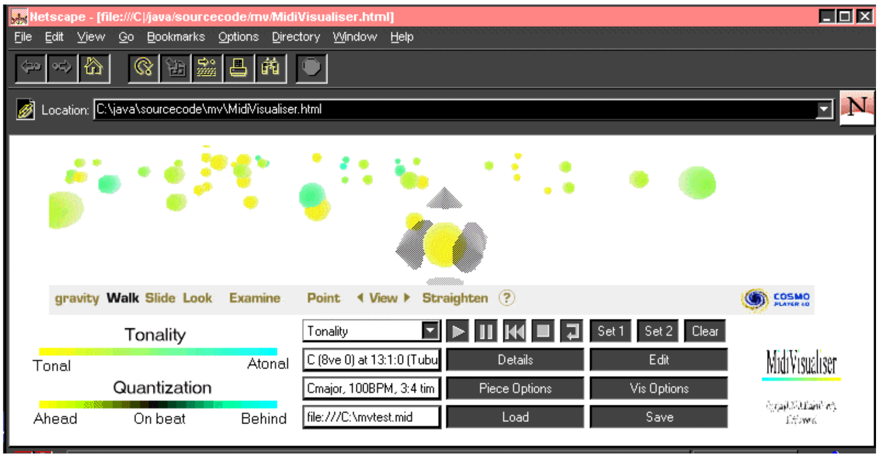


Figure 7: MidiVisualiser java applet and embedded VRML world.

However, the main script is more sophisticated than a simple proxy for the applet. It stores as a part of its state the currently selected note object, so that a sequence of translation and rotation instructions can be sent through the EAI and routed to the previously selected note by the main script, thus minimising EAI throughput when performing repetitive actions such as moving a note. The script also has responsibility for simple object behaviours such as the selection and display of Translation and Valuator Widgets. The more complex object behaviours, such as the object colours which are generated from musical analysis functions are calculated in the Java portion of the application.

5. Evaluation

A brief series of user trials was undertaken to gain some insight into the effectiveness of the visualisation techniques and metaphors chosen. Although the size of the user group was small (five unpaid subjects), a relatively wide range of musical experience was represented, ranging from an accordion player to a music teacher. A 30 second test piece in C major was presented and subjects were asked to view a static visualisation, then to view a visualisation with sound and animation, next they manipulated the structure of the music and finally discussed their reactions.

The overall impression given by all subjects was that the visualisation was understood very quickly and that no dissonance was experienced in correlating the static and dynamic structure with their own natural interpretation of musical form. The tonality mapping was interpreted correctly by all subjects without even having heard the piece – the match of blue-tonal/red-atonal therefore seems to be a good match which is easily understood. Most users pointed out musical features in the data set such as chords and arpeggios without actually being prompted to identify any features. One subject reported that when watching the animation the separate experiences of *hearing the sound* and *seeing the colour* began to fuse together. The implication is that the experience can be one of music and image combined as a tightly integrated single entity – further work will investigate the role of synaesthesia in these visualisations in more detail.

6. Conclusions and Future Work

This paper has described how visualisation techniques for music were derived from Benedikt's principles, leading to the exploration of spiral forms and composite dimensions for representing MIDI data.

Possible future work may include extending the software to include alternative 'pluggable' visualisation mappings other than the tonality and quantization mappings described in sections 3.5 and 3.6. It would be also be interesting to include support for MIDI input – the composition task within MidiVisualiser would be far more easily accomplished if MIDI devices such as keyboards and pedals could be used for input.

User trials pointed to the fact that the technique for modifying instrument sound does not appear as natural to use as those of the other musical attributes. This poses a dilemma, because instrument sound must be expressed spatially (on the U dimension of the spiral) in order to satisfy Benedikt's Principle of Exclusion, therefore this configuration must be assumed as a constant. However, the attribute of instrument sound is not one which lends itself to being manipulated by spatial gestures. Currently this issue is unresolved.

7. Acknowledgements

Alan Graves gratefully acknowledges the financial support of EPSRC during this work. Thanks also to Howell Istance for his support, and to Michael St. Hippolyte for making available his Java MIDI package as used in the MidiVisualiser prototype.

8. References

Baecker, R and Small, I (1990), "Animation at the Interface", in Laurel B (ed), *The Art of Human-Computer Interaction*, Addison-Wesley, 1990. pp 251-267.

Benedikt, M (1991), "Cyberspace: Some Proposals", in Benedikt, M (ed), *Cyberspace: First Steps*, MIT Press, 1991. pp.119-224.

Benford, S and Mariani, J (1994), "Virtual Environments for Data Sharing and Visualisation - Populated Information Terrains", Research report CSCW/5/1994, Lancaster University.

Brown, R D (1996), "Virtual Unreality: The Modulation of Time, Space and Energy", in Hand, C (ed), *3rd UK Virtual Reality Special Interest Group Conference: Short Paper Proceedings*. De Montfort University, Leicester, 3rd July 1996. pp23-26.

Classen, C (1993), *Worlds of Sense*, Routledge, 1993.

Cowperthwaite, D J, Sheelagh, M, Carpendale, T, and Fracchia, F D (1996), "Visual Access for 3D Data", in *ACM CHI '96: Conference on Human Factors in Computing Systems*, Short Paper, Conference Companion, pages 175-176, Addison-Wesley, 1996., pp175-176.

Cugini, J, Piatko, C and Laskowski S. (1996) "Interactive 3D Visualization for Document Retrieval", CIKM '96, Workshop on New Paradigms in Information Visualization and Manipulation, November 16, 1996, Rockville, Maryland. <URL: http://zing.ncsl.nist.gov/~uicd/viz.html>

Day, S (1996), "Synaesthesia and Synaesthetic Metaphors", *PSYCHE: An Interdisciplinary Journal of Research on Consciousness*, **2**(32).

Drew, N and Hendley, B (1995) "Visualising Complex Interacting Systems". In *ACM CHI '95 Conference Companion,* May 7-11, 1995, Denver, Colorado, pp. 204-205
<URL: http://www.cs.bham.ac.uk/~nsd/Research/Papers/CHI95/nsd_bdy.html>

Feiner, S and Beshers, C (1990), "Worlds Within Worlds, Metaphors For Exploring N-Dimensional Virtual Worlds", in *Proceedings of UIST'90*, October 3-5, 1990, Snowbird, UT, pp.76-83.

Kunze, T and Taube, H (1996), "SEE- A Structured Event Editor: Visualizing Compositional Data in Common Music", in *Proceedings of the ICMC'96*, Hong Kong, pp. 63-66.
<URL: http://cmn19.stanford.edu/~tkunze/papers/icmc96-see-paper/see.html>

Marrin, C, Kent, J, Immel, D, and Aktihanoglu, M. (1997) "Using VRML Prototypes to Encapsulate Functionality for an External Java Applet". Electronic publication.
<URL: http://www.marrin.com/vrml/papers/InternalExternal/Chris_Marrin_1.html>

Sagan, H (1994) *Space-filling Curves*, Springer-Verlag, 1994.