A JAVA FRAMEWORK FOR DATA SONIFICATION AND 3D GRAPHIC RENDERING

Domenico Vicinanza
University of Salerno - Musica inaudita
Department of Mathematics and Computer Science
dvicinanza@unisa.it

ABSTRACT

Data audification is the representation of data by means of sound signals (waveforms or melodies typically). Although most data analysis techniques are exclusively visual in nature, data presentation and exploration systems could benefit greatly from the addition of sonification capabilities. In addition to that, sonic representations are particularly useful when dealing with complex, high-dimensional data, or in data monitoring or analysis tasks where the main goal is the recognition of patterns and recurrent structures.

The main goal of this paper is to briefly present the audification process as a mapping between a discrete data set and a discrete set of notes (we shall deal with MIDI representation), and look at a couple of different examples from two well distinguished fields, geophysics and linguistics (seismograms sonificaton and text sonification). Finally the paper will present another example of the mapping between data sets and other information, namely a 3D graphic image generation (rendered with POVRay) driven by the ASCII text.

1. ABOUT DATA AUDIFICATION

Data audification is the representation of data by sound signals; it can be considered as the acoustic counterpart of data graphic visualization, a mathematical mapping of information from data sets to sounds. In the past twenty years the word audification has acquired a new meaning in the world of computer music, computer science, and auditory display application development.

Data audification is currently used in several fields, for different purposes: science and engineering, education and training, in most of the cases to provide a quick and effective data analysis and interpretation tool [3]. Although most data analysis techniques are exclusively visual in nature (i.e. are based on the possibility of looking at graphical representations), data presentation and exploration systems could benefit greatly from the addition of sonification capabilities. In addition to that, sonic representations are particularly useful when dealing with complex, high-dimensional data, or in data monitoring tasks where it is practically impossible to use the visual inspection. More interesting and intriguing aspects of data sonification concern the possibility of describing patterns or trends, through sound, which were hardly perceivable otherwise.

Moreover in many cases human ears are used to discover slight changes in acoustic patterns. One example is the medicine who routinely applies acoustic data analysis when uses the stethoscope to listen to breath noise and heart tones. Audification may give information about the inner structure of the represented data using the power of an abstract description. Any kind of regularity in the original data set will be reflected to the aural signal generated by the audification algorithm.

One of the most important challenges is finding the proper balance between the amount of information that can be converted into an audio signal and the effective capability of that sound to communicate meaningful information to a listener.

2. MIDI ”MELODISATION” OF A DISCRETE DATA SET

The melodisation of a data set, i.e. the creation of a melody starting from a list of data is an interesting way to convert into aural signals, almost any kind of information. For the sake of simplicity we can imagine a set of m elements (a list or a discrete set of values) to be sonified. The sonification will provide a melody, a list of notes chosen among a set of n notes, to be drawn onto a pentagram. Without loose of generality, we can consider a standard, numerical, coding convention for the (well tempered) notes to be represented. In particular we shall refer to the MIDI[1] code, according to which the central ”C” note corresponds to the integer 60, ”C#” is 61, ”D” is 62, and so on (any semitone shift will add or subtract one from that previous value). The lowest acceptable MIDI value is 0 and the higher is 127, so there are 128 possible notes to represent our data.

From a mathematical point of view the MIDI melodisation (the set of possibile notes is 128, so $n = 128$) could be considered as a map between two discrete set,

$$\mathbb{Z}_m \to \mathbb{Z}_{128}$$

[1] The original Musical Instrument Digital Interface (MIDI) specification defined a physical connector and message format for connecting devices and controlling them in ”real time”. A few years later Standard MIDI Files were developed as a storage format so performance information could be recalled at a later date.

which have to preserve the structural properties of the original data (such as the invariance with respect to a certain symmetry group, as it will be shawn in the next sections).

Compared to other sonifications methods, the MIDI sonification has the great advantage to code information into one of the most widely used musical format. Almost all the applications for music analysis/sequencing are able to operate and play MIDI files in extremely customizable manners. Moreover many programming languages, as Java for example, have the possibility to load and use special libraries to efficiently manage them (see for example jMusic [2] [4]).

Finally several tools have been developed to analyze and process MIDI files, to discover self-similarities, repetition of microstructures, statistical diagrams (one for all, the well known ”MIDI toolbox” written by Petri Toiviainen [6]).

3. MIDI SONIFICATION OF MT. ETNA VOLCANO SEISMOGRAMS

Figure 1. Satellite snapshot of a typical Etna volcano eruption. Picture from the Catania INGV website (www.ct.ingv.it)

3.1. Mt. Etna volcano and eruption forecasting

Mount Etna (fig. 1) is the Europe largest volcano (its volume is at least 350 km^3), and one of the most active volcanoes on Earth. It is situated along the eastern (Ionian) coast of Sicily and occupies a surface area of around 1200 km^2 with a perimeter exceeding 135 km. Its summit height varies frequently (from about 3315 to 3350 m) depending upon the eruptive activity or minor collapse events at the summit craters.

A seismogram is a particular representation of the (more or less) regular oscillations of the volcano. Following this approach, we can consider the volcano as a huge musical instrument, which can produce a sound not hearable by humans since its frequency is too low. The larger is the mountain, the lower are the frequencies emitted. Seismograms are the graphical representations of these infrasonic sounds.

One of the possibile audification of the seismic data is a straight conversion of the seismic wave to a listenable waveform. Such a waveform generation, both using CSound [1] as acoustic compiler and a Java package (running both on single PCs and on clusters like GRID [2]) for Mt. Etna volcano digital signals, has been widely discussed in some previous papers [12], [13]. In those cases the spectrum of the generated sounds will explicitly reveal regularity properties, such as hidden patterns or long range correlations.

The MIDI sonification discussed here in this paper moves the point of view, saying it pictorially, from the ”microscopic word” of the waveform construction (and analysis) to the ”macroscopic (and multidimensional) environment” of the MIDI notation (i.e. the musical score). Now, any regular behaviour in the original data would become a regular melodic pattern, a sort of musical phrase which could be isolated and whose occurrences can be studied with standard MIDI analysis tools. In this sense patterns and long range correlations become macroscopic, they can be stretched and enlarged using different beats metro, recognized and played with different emphasis by different instruments.

This process can be used to drive, for example, an automatic procedure for orchestration, which could be created algorithmically taking into account the several occurrences of the musical phrases.

3.2. From seismograms to score

The MIDI sonification of seismograms could be described in a simple manner as follows. Let us start from an empty score, as the one shown in fig. 2. We can than imagine to superimpose the seismogram to the score, as in fig. 3, (the scale, i.e. how many musical notes or octaves are involved is one of the parameter of the sonification, one could just take into account all the 88 piano keys, or all the 128 MIDI notes, or some other subset like all the white keys, …) and draw the notes according to the shape of the seismogram (fig. 4). Again the time scale in another parameter of the sonification. As we were discussing above, one can tight or enlarge the rhythmical structure to enlighten some particular phenomenon. The score is hence ready to be played (fig. 5).

The Java package developed for this kind of sonification works just in this way, starting from a discrete data set (the digitalized seismogram) to have the MIDI score. The metro is customizable, as well as the musical scale (i.e. the subset of 128 MIDI notes to be used) and the playing instrument.

[2] Established in 1998, jMusic is a project designed to provide composers and software developers with a library of compositional and audio processing tools. It provides a solid framework for computer-assisted composition in Java, and is also used for generative music, instrument building, interactive performance, and music analysis.

Figure 2. From volcano seismograms to score: we start from an empty score

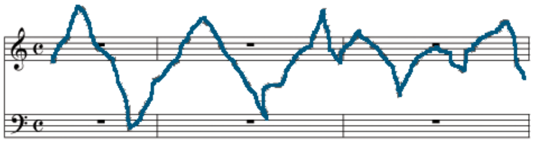

Figure 3. From volcano seismograms to score: we superimpose the seismogram

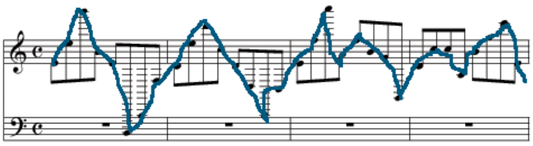

Figure 4. From volcano seismograms to score: we put notes according to the shape of the seismogram

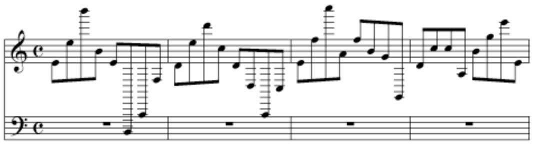

Figure 5. From volcano seismograms to score: finally we get the melody

Following this procedure, we shall get a melody whose profile follows perfectly the shape and the behaviour of the seismograms, reproducing the oscillation amplitude variations and giving a quick and effective insight to their overall structural properties.

Some examples of this sonification can be listened from the Mt. Etna volcano sonification web site, `http://grid.ct.infn.it/etnasound`, where it is also possible to download the PDF scores of such sonifications.

4. ANOTHER EXAMPLE: TEXT SONIFICATION

We have already observed that any kind of regularity of the original data set will be conveyed to the aural signal generated by the audification algorithm. Let us study a particular case of audification, where this ”information transfer” is particularly easy to observe: the text messages sonification.

During the first decades of the XI Century, Guido Monaco, also known as Guido d’Arezzo (990-1050 ca), a Benedictine Monk who invented the system of staff-notation still in use, proposed (and used in several musical pieces) a method to associate a note on the pentagram to each vowel of text to sing. The duration of the notes was depending upon the number of consonants between each vowel. The following figure represents an excerpt of a manuscript (it is the ending part of a Gloria) in which the words ”saeculorum Amen” have been sonified according to this algorithm (”ae” has been sonified as ”e”). The algorithm was not strict, in the sense that there was a certain degree of freedom in the choice of notes (according to the composing rules of that age).

In the text sonification, the most natural choice is maybe the following one: ”the same set of characters will return exactly the same musical phrase” (i.e. the same sequence of notes), so that one might easily (acoustically) recognize words which start with the same root, or end with the same letters, as rhymes, for examples. In some sense one can listen to the metric, using melodies.

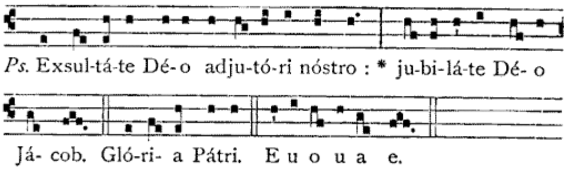
Ps. Exsul-tá-te Dé-o adju-tó-ri nóstro : * ju-bi-lá-te Dé- o
Já- cob. Gló-ri- a Pátri. E u o u a e.

Figure 6. Manuscript of the final section of ”Gloria Patri” by Guido d’Arezzo. ”Euouae” indicates the vowels of the syllables of ”saeculorum Amen”[10].

There are many audification possibilities, thanks to different possible algorithms to associate notes to the messages, keeping valid the covariance principle. One of the first we have implemented is a sort of Java version of the original Guido’s method. We proposed other possible sonification algorithms: standard dodecaphonic (chromatic), whole tone (Debussy style), pentatonic, major diatonic, blue scale, plus some other ancient scales (Greek modes).

In the chromatic mode, the pitch will be chosen according to the position of the sonified letter (namely its ASCII code) into the English alphabet, so a ”B” will be a semitone higher with respect to a ”A”, a ”D” a third minor higher, and so on. The algorithm is case sensitive, so that a ”B” will be lower than a ”b”. In addition to that, the duration of the consonants the one of a 1/16 note in the MIDI file, while the duration of vowels is a 1/8 note. Examples can be listened to the website `www.musicainaudita.it`.

In particular they have been developed two different version of the program, a first one is able to run offline, taking a text file as input, tested on single computer and large clusters (a working demo on clusters is available within the GRID [2] tutor website[3] at the URL `https://grid-tutor.ct.infn.it`). A second implementation is an applet (simplified and didactical) directly playable on the web at the URL `www.musicainaudita.it/sonification`, where it has been setup a two voices canon generator (the second voice starts exactly one measure after the first one) using the ancient Greek scales as pitch sets.

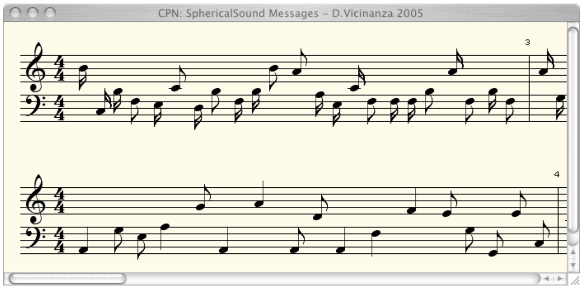

Figure 7. The output of the applet version of the text sonification program

4.1. An example: SATOR AREPO TENET OPERA ROTAS. Covariance of the MIDI audification under transformations

Let us consider another example, to make some consideration about the sonification process. The sentence ”SATOR AREPO TENET OPERA ROTAS” (sometimes called the ”sator square” or ”Templar sqare”) is a Latin palindrome (the most ancient have been found in the Pompeii archaeological site, but many of the same palindromes have been discovered in several other places), the words of which, when written in a square, may be read top-to-bottom, bottom-to-top, left-to-right, and right-to-left, returning the same result. A representation could be given as follows:

S	A	T	O	R
A	R	E	P	O
T	E	N	E	T
O	P	E	R	A
R	O	T	A	S

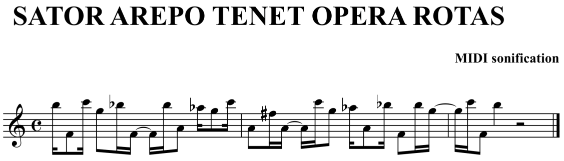

Figure 8. Score from ”SATOR AREPO TENET OPERA ROTAS” sonification

Using a MIDI audification, so a melodisation, we can associate to each of the five letters which appears in the text a note. An example is shown in fig. 8; in this case the character-note map associates the note B to the letter ”S”, F to the letter ”A”, C to the letter ”T”, G to the letter ”O” and Bb to the letter ”R”. The duration of the notes has been chosen as follows: vowels duration is set to 1/8 and consonants duration is set to 1/16.

Since this melodisation maps linearly the ASCII code of the characters to the MIDI notes, is clearly covariant, for example, to the same character is remains associated the same note (with the same duration).

We can observe, for example, that the musical score in fig. 8 has exactly the same palindromic structure (both for the melody and the rhythm) of the text SATOR AREPO TENET OPERA ROTAS, so listening to it one can recognize the same structural properties with which the text has been composed.

Moreover, we can try to act on the text by transforming it[4], for example by using an historical cryptographic coding, namely the one adopted, during the XIX century, by one of the secret associations of that age. This coding works by exchanging each letter of the alphabet with the one with the closer ”sound”, so the A will be exchanged with a O, B with P, C with G, D with T, E with I, F with V, L with R, M with N, S with Z, leaving unchanged the H, K, J, Q, U, W, Y, X. Just to fix the idea, the word ”Hello!” would become ”Hirra!” or the sentence ”Sonification is interesting” becomes ”Zamevegodeam ez emdilizdemc”.

Our sentence once coded will be: ”ZODAL OLIBA DIMID ABILO LADOZ” which sonified will give the score represented in fig. 9
We can observe that melody remains palindromic, the rhythm is exactly preserved (since the coding does not exchange consonants and vowels). One can compare in fig. 10 the two musical scores quantitatively, plotting the two histograms of the occurrences of the musical notes for the two musical phrases.

[3] GRID is the Europen project for the distributed computation

[4] To be more precise, we have to act on the text with a transformation which ”respects” the symmetry properties of the text, in other (mathematical) words, which is invariant with respect to the symmetry group of our text, namely the Klein group.

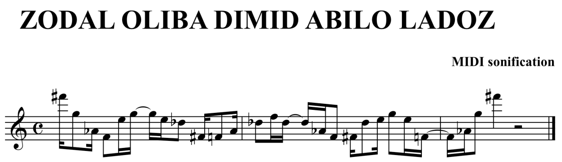

Figure 9. Score from ”ZODAL OLIBA DIMID ABILO LADOZ” sonification

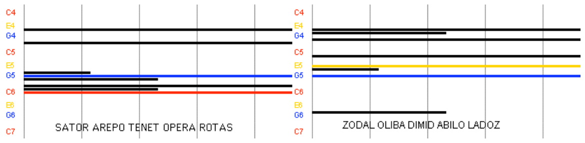

Figure 10. Histograms of the MIDI notes occurrences for the two sonifications

Finally, fig. 11, gives another sonification example, showing the melodisation of several anagrams of the word ”aster”. It is interesting to observe the particular structure which is assumed by the score.

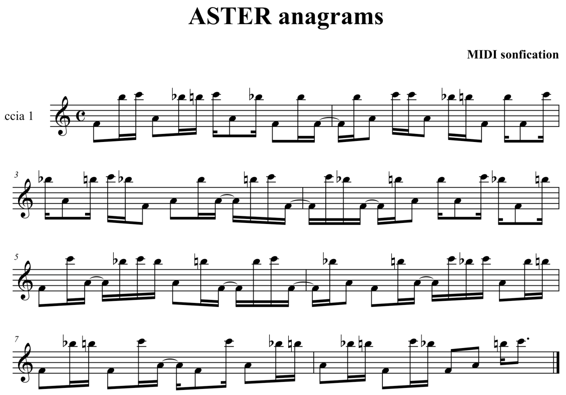

Figure 11. Another example: score from ”aster” anagrams sonification

5. 3D IMAGE RENDERING AS DATA REPRESENTATION

In the previous sections we have discussed the MIDI sonification of a certain given data set. Generally speaking, as it has been observed before, a sonification procedure is a map between two discrete sets of values. Going ahead, we con consider another example of mapping, this time addressed to graphical images.

5.1. 3D rendering using POVRay: a first example

The Persistence of Vision Raytracer (POVRay[11]) is a high-quality, totally free tool for creating three-dimensional graphics. It is available in official versions for Windows, Mac OS/Mac OS X and i86 Linux. The source code is available for those wanting to do their own ports. The rendering engine is controlled by a sort of special programming language with primitives which define the elementary objects, such as spheres, lines, planes, regular shapes, . . .

Here follows an example, about how to get a simple sphere on a reflecting background.

```
union{
 sphere{z*9-1,2}
 plane{y,-3}
 finish{reflection{,1}}
}
background{1+z/9}
```

This code has been rendered in fig. 12. The first line declares that the object which are going to be defined will be joined together. The second and the third ones declare a sphere in the second line (radius=2) and then an horizontal plane. The finish command, set the surfaces behaviour as reflecting/metallic. Finally the last line defines a gradient background to complete the scene.

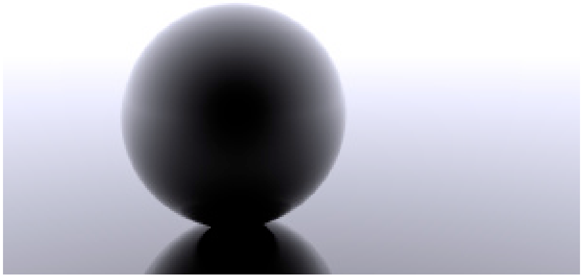

Figure 12. A POVRay example, a sphere rendering (from the POVRay shortest code contest: http://astronomy.swin.edu.au/ pbourke/raytracing/scc3/)

5.2. 3D graphical rendering and data representation of three characters

The Java package developed for this purpose, takes as input a text file and returns as output a 3D source file to be rendered with the ray-tracing program POVRay (also in this case a full working demo running on the PC cluster of the European project for distributed computation, GRID, is available at the URL: `https://grid-tutor.ct.infn.it`). Again, to each character of the message will be associated a graphical object, a sphere in this case. Spheres position, texture, index of reflection, surface properties and degree of transparency will be determined as a function of the position of the letter into the alphabet. An example is shown in fig. 13 where three characters have been rendered using the Java package and POVRay.

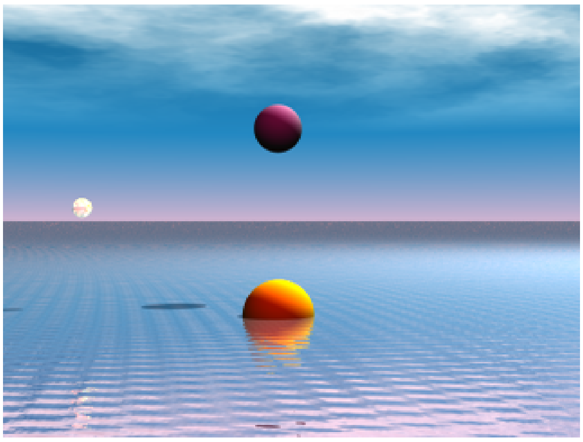

Figure 13. 3D representation of three charachters. The x, y, z position, the radius of each sphere, as well as its surface properties (reflection index, refraction index, diffusion coefficient) have been calculated according to the position of the letters within the English alphabet

```
#include "colors.inc"
 ... settings, background, camera

sphere {<8,33,18>,5.935057
  pigment {colour
     red 0.65882355
     green 0.65882355
     blue 0.65882355
     filter 0.0
   }
   finish {
      reflection {0.0}
      ior 0.0
      phong 0.0
   }
}

sphere {<-16,13,-23>,2.2259622
  pigment {Yellow filter 0.0}
    finish {
       reflection {1.0}
       ior 0.0
       phong 1.0
    }
}

sphere {<31,-10,-5>,7.56183
  pigment {Yellow filter 0.0}
    finish {reflection {0.0}
    ior 0.0
    phong 0.0
  }
}
 ...
```

Four rendering sets of parameters have been chosen to provide four different representations (14 - 17): two of them with glassy spheres onto a sea-and-sky landscape, one with a low-key background and distorted objects, a least one with distorted glassy spheres on a water background. In the POVRay code shown above, it is easy to recognize the declaration of three spheres, generated according to the three mapped characters of the example. Each sphere command will draw a sphere on the screen once the POVRay source file is compiled. The first algorithm will put the graphical objects on an algorithmically generated sky and sea background, without any processing, while the other rendering algorithms will apply a (wavy) distorsion to the final picture. The POVRay sphere command allows to operate on the following parameters: x,y,z coordinates of the center, radius, color (RGB code), coefficient of reflection, index of refraction (ior in the program), phong (size of the reflection spot on the sphere surface), texture.

6. CONCLUSIONS

Two data sonification and a 3D rendering Java packages have been presented. All the packages are able to run offline taking as input a data file or a text file both on single PCs and on clusters (see the next section for more detailed instructions).

The sonification applications are also able to run interactively, embedded into website as applet (see `http://www.musicainaudita.it/sonification` to have an example).

7. ACKNOWLEDGMENTS

The author is particularly grateful to Emidio Giorgio and Giuseppe La Rocca from INFN Catania for their precious contribution to set up the GRID interactive version of the MIDI sonification (of both text messages and seismograms data) and 3D rendering. The online version of those software is available at the URL: `https://grid-tutor.ct.infn.it`. To run one of the jobs, click on left menu on the link ”VO Services”, then after the login procedure (the user has just to click on the login button in the page which will appear), it will be shown a menu on the left side where it appears the link ”Sonification”.

This web interface allows any user to connect and run sonification and 3D representation directly on the GRID, the European largest distributed cluster for massive computation in a transparent and simple way.

8. REFERENCES

[1] CSound acoustic compiler, `http://www.csounds.com`

[2] Foster, I. and Kesselman, C. (2001), *The Anatomy of the GRID*, Int. J. Supercomputer Applications, 15(3)

[3] Lane, D.M. (1999-2002), *Development of sonification design theorgy: Metaphors, map-*

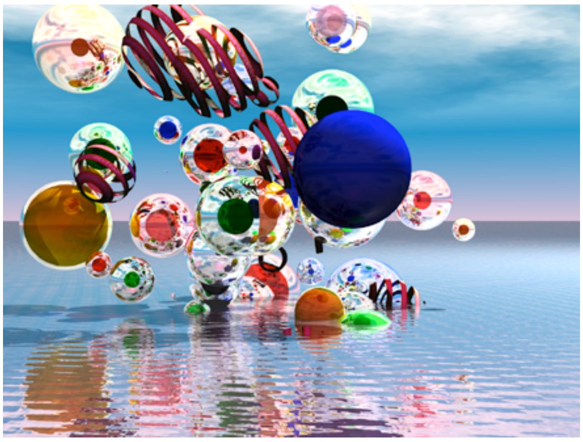

Figure 14. 3D rendering (of the POVRay source file) of the message: SphericalSound Message Sentences sonifications and 3D rendering (first algorithm).

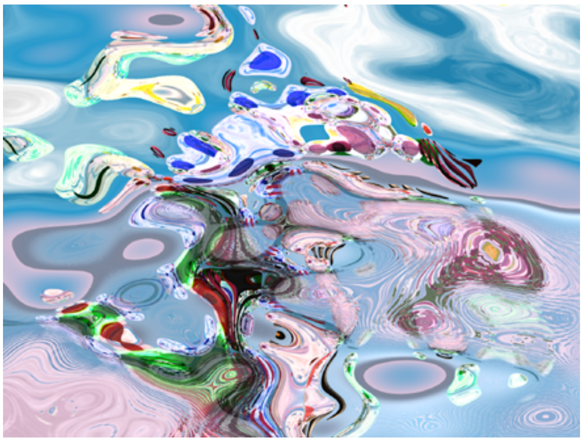

Figure 15. 3D rendering (of the POVRay source file) of the message: SphericalSound Message Sentences sonifications and 3D rendering (second algorithm).

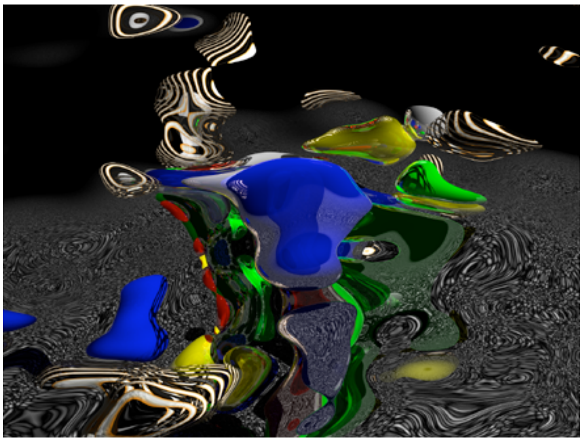

Figure 16. 3D rendering (of the POVRay source file) of the message: SphericalSound Message Sentences sonifications and 3D rendering (third algorithm).

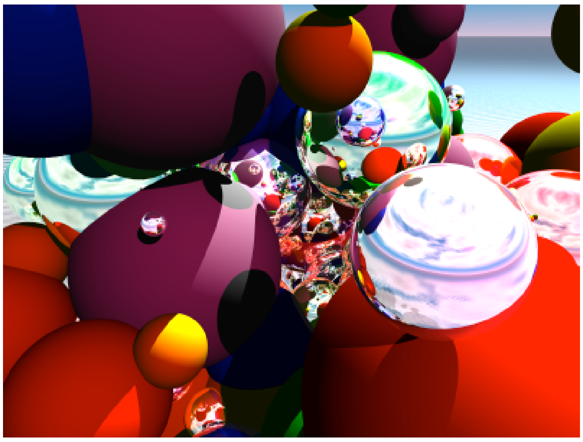

Figure 17. 3D rendering (of the POVRay source file) of the message: SphericalSound Message Sentences sonifications and 3D rendering (fourth algorithm).

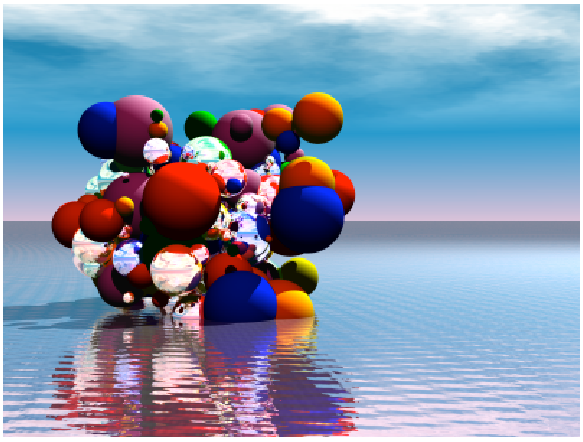

Figure 18. Another example of 3D rendering (of the POVRay source file) using the first algorithm, the source was a T.S. Elliot poetry.

pings, holistic sound design, and data-specific sonification, National Science Foundation

[4] jMusic (Java API for MIDI notation) website: `http://jmusic.ci.qut.edu.au/`

[5] The INFN GRID project, and the European DATAGRID project: http://www.infn.it/grid, `http://www.datagrid.org`

[6] `http://www.jyu.fi/musica/miditoolbox/`

[7] The GRID INFN Laboratory for Dissemination Activities (GILDA): https://gilda.ct.infn.it

[8] Malinowski, *Audification/Sonification*, `http://www.musanim.com/mam/samindex.html`

[9] Tripp, A.C, *Geological Sonification: Using Auditory Signals to Represent Geological Data*, proc. of the AAPG annual meeting 2003.

[10] Gregorian Chant notation `http://interletras.com/canticum/`

[11] Persistence of Vision RayTracer (POVRay), `http://www.povray.org`

[12] Vicinanza, D. (2004), *First results about sound synthesis and sonification within GRID*, in proc. of the UCM 2004 International Conference, Caserta, Italy, 2004

[13] Vicinanza, D., ”Making music with GRID - Sonification of ETNA Volcano Seismograms”, poster presented at the ”50 years of CERN” celebrations, Geneva, Switzerland, October 16, 2004