AUDITORY REPRESENTATIONS AS LANDMARKS IN THE SOUND DESIGN SPACE

Carlo Drioli
Pietro Polotti
University of Verona
{carlo.drioli,pietro.polotti}
@univr.it

Davide Rocchesso,
Stefano Delle Monache
IUAV University of Venice
roc@iuav.it
stefano.dellemonache@gmail.com

Kamil Adiloğlu,
Robert, Anniés,
Klaus Obermayer
Berlin Institute of Technology
{kamil, robokopp, oby}
@cs.tu-berlin.de

ABSTRACT

A graphical tool for the timbre space exploration and interactive design of complex sounds by physical modeling synthesis is presented. It is built around an auditory representation of sounds based on spike functions and provides the designer with both a graphical and an auditory insight. The auditory representation of a number of reference sounds, located as landmarks in a 2D sound design space, provides the designer with an effective aid to direct his search along the paths that lie in the proximity of the most inspiring landmarks.

1 INTRODUCTION

Sound synthesis techniques are nowadays available which offer a high degree of naturalness and expressiveness. Sometimes, however, this comes at a price of a consistent complexity in both the control of real time performance, and the parametric tuning required in the sound design process. One such case is physical modeling, in which the process that produces the sound is represented by means of more or less simplified equations of the underlying physical laws. As a result, the parameters involved in the modeling should in principle be easy to understand because they have a real counterpart, and should be easy to control because our sensorial experience should mediate the action-reaction patterns to which they relate[1]. Nonetheless, the most accurate and expressive models available today are often described in terms of detailed physical relations and parameters that are not always accessible or understandable to the non specialists. Moreover, the nonlinear nature of the phenomena under observation may sometimes lead to numerical schemes that do not always reflect the behavior of the real systems in the whole parameters space. These considerations motivate the search for new tools to aid the synthesis parametric tuning, and are of particular interest in our opinion in the case of physical modeling audio synthesis.

In this paper we propose a graphical tool for the timbre space exploration and interactive design of complex sounds by physical modeling synthesis that is intended to assist the sound designer. It is aimed at exploiting the auditory representation of sounds based on spike functions and provides the designer with both a graphical and an auditory insight that may be used in place of, or combined with, the set of low-level physical parameters of the models. The graphical tool adopts a sonic landscape paradigm, in which the auditory representation of a number of reference sounds can be located as landmarks in a 2D sound design space, and provides an effective aid to direct the search along the paths that lie in the proximity of the most inspiring sonic landmarks.

The use of terminology and metaphors referring to the environment and landscapes has a rather old tradition in the field of sounds perception, especially when referred to the perception of ecological and everyday sounds. In the late 70's, the Canadian composer R. Murray Schafer introduced the term "soundscape", defined as the auditory equivalent to landscape[2], and Barry Truax published his Handbook for Acoustic Ecology[3]. The term soundscape perception is also used in a scientific context to characterize how inhabitants perceive, experience and appraise their sonic environment.

Our use of the terms "sonic landscape" and "sonic landmark" however refers to a particular organization of sounds in a 2D sonic space. The sound synthesis framework we rely on is based on a class of physical models for everyday sounds, which includes low level events (impacts and friction) and high level ones (bouncing, breaking, rolling, crumpling). This Physically-based Sound Design Tools (SDT from now on) is being developed and supported within a number of EU funded research projects on audio synthesis and sound design (Sounding Objetcs (SOB), Closing the Loop of Sound Evaluation and Design (CLOSED), Natural interactive walking (NIW))[4, 5]. An example of graphical interface for the parametric tuning of friction sounds, in-

SMC 2009, July 23-25, Porto, Portugal

cluded in the SDT and implemented in Max/Msp, is shown in Figure 1. The aim of the tool proposed here is to add to this class of tuning interfaces a perceptual representation output layer (spike functions) and a tool to perform parametric interpolations using reference auditory representations in the perceptual space. The representation of sounds based on the spike functions provides the designer with both a graphical and an auditory insight. The latter point is of extreme interest since it is based on a sort of "auditory representation of sound". The peculiarity of the spike representation is, in fact, to be lossy. This aspect turns to be an advantage, whereas the reconstructed sounds maintain the temporal articulation of the original ones but they are decolored from their original timbre characteristics. Any reconstruction of a sound has a "watery character", so that all the sounds become timbrically similar, even if distinguishable according to their temporal structure. In this way, the sound designer has at disposal a kind of "auditory-radiography" of the sounds that can be compared as sonic archetypes, helping the designer in her/his sound space exploration.

The paper is organized as follows: first, a description of the sound synthesis engine and of the spike-based auditory representation is given in Section 2; Section 3 describes the interpolation sound space and the graphical tool proposed to assist the sound designer; in Section 4, same sound design examples obtained with the tool are illustrated; Section 5 contains the conclusions and future issues.

2 DESIGN AND IMPLEMENTATION OF THE SPIKE-BASED PARAMETER INTERPOLATION TOOL

The interactive interpolation tool presented is organized as a client-server distributed application, in which the client side hosts the user graphical front-end based on the auditory spike sound representation, and the server side hosts the SDT audio synthesis engine. In the following we provide some details on the sound synthesis algorithm used to generate the reference and the new sounds, on the spike analysis framework used to graphically represent the sounds, and the properties of the interpolation in the 2D space.

2.1 The sound synthesis engine

The sound synthesis chosen to illustrate our design tool is a physical modeling implementation of friction sounds synthesis, included in the aforementioned SDT package.

The scope of the SDT is to provide a platform of sound synthesis tools that interaction designers can easily exploit in their sketching activities and that can be run on common real time software such as Max/MSP and Pd. The aim is also to provide the patches with a set of side tools to easily manage projects and that can be of help when working with

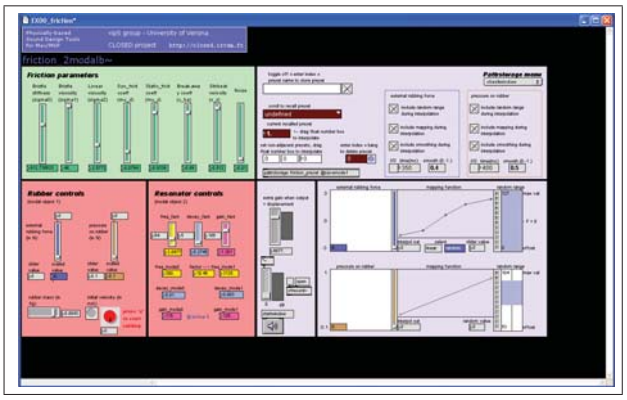

Figure 1. The friction MAX/Pd patch, contained in the SDT synthesis tools package. The user can tune up to 27 synthesis parameters representing the driving forces, the characteristics of the modal resonators, and the characteristics of the nonlinear interaction.

acquisition boards and sensors. Most of the models contained in the SDT, aimed at reproducing sounds from solid objects interaction, are structured as two resonating objects interacting by means of a contact model. The friction model specifically referred to here, relies on a description of the average behavior of a multitude of micro-contacts made by hypothetical bristles extending from each of two sliding surfaces. A modal decomposition is adopted for both interacting objects, leading to a first parametric subset including modes frequency, decay factors and gain. The remaining low-level parameters are related to the interaction mechanisms and to the interaction force specification. To gain an insight of the phenomenological role of the low-level physical parameters of the friction model, and of what a sound designer can be asked to deal with, a description is given in Table 1, and are visible in the SDT tuning interface of Figure 1 . Further details on the friction model can be found in [4], Chap. 8.

Sym.	Physical Description	Phenomenological Description
σ_0	bristle stiffness	affects the evolution of mode lock-in
σ_1	bristle dissipation	affects the sound bandwidth
σ_2	viscous friction	affects the speed of timbre evolution and pitch
σ_3	noise coefficient	affects the perceived surface roughness
μ_d	dynamic friction coeff.	high values reduce the sound bandwidth
μ_s	static friction coeff.	affects the smoothness of sound attack
v_s	Stribeck velocity	affects the smoothness of sound attack
f_N	normal force	high values give rougher and louder sounds

Table 1. A phenomenological guide to the friction model parameters.

2.2 Spike representation

In a classical signal representation approach, overlapping discrete blocks are used. However, in this method, in particular for non-stationary signals, a small shift of a block can cause a large change in the representation, depending on where an acoustic event falls within the block. A sparse, shiftable representation method based on atom-like filter functions can solve the problem. Hence, following [6], a sound signal $x(t)$ can be approximated as a linear combination of K filter functions, the so-called spikes. In this study, we employed the Gammatone functions $\gamma_{f_k,t_k}(t)$ with amplitudes a_k and residual $\epsilon_{K+1}(t)$:

$$x(t) \quad = \quad \sum_{k=1}^{K} a_k \gamma_{f_k,t_k}(t) + \epsilon_{K+1}(t). \qquad (1)$$

A Gammatone function is defined by its center frequency f_k and its filter order. In a Gammatone filter bank, the filter center frequencies are distributed across the frequency axis in proportion to their bandwidth. The shape of the magnitude characteristics of the fourth order Gammatone filter approximates the magnitude characteristics of the human auditory filter in a proper way [7]. Hence, these filter functions have a biological background.

Each spike $s_k = (t_k, f_k, a_k)$ is composed of the temporal offset t_k, the center frequency f_k of the corresponding Gammatone filter and the amplitude a_k.

For a pre-determined number of spikes K, the optimal coefficients a_k, f_k, t_k, in terms of a minimal residual signal $\epsilon(t)$ have to be found. We employ matching pursuit [8, 6] to iteratively determine the spikes s_k and to minimize the residual. The algorithm is initialized by setting the residual

$$\epsilon_1(t) := x(t). \qquad (2)$$

Then for $1 \leq k \leq K$, the optimal time offset t_k and center frequency f_k are determined iteratively so that the Gammatone filter $\gamma_{f_k,t_k}(t)$ maximally correlates with the signal $\epsilon_k(t)$:

$$(f_k, t_k) = \text{argmax}_{f,t^*} < \epsilon_k(t), \gamma_{m_{f,t^*}}(t) > . \qquad (3)$$

In order to perform the scalar product shown in Equation 3 between the filters and the signal, the filters should have certain window lengths in time domain. In his auditory toolbox, Slaney [9] used fixed window lengths for each filter in the filterbank. However, the energy levels decrease much slower for the low frequency filters than for the high frequency filters. Therefore, we adapted the window lengths considering the center frequencies. We calculated the positions within each filter, where the total energy in the time envelope falls to its thousandth. We used these position values in time as the window lengths to calculate the scalar products. Hence, the amplitude of the k^{th} spike is defined to be the scalar product between the corresponding filter, with its window length, and the residual signal, as follows:

$$a_k := < \epsilon_k(t), \gamma_{f_k,t_k}(t) > . \qquad (4)$$

Then we update the residual by

$$\epsilon_{k+1}(t) = \epsilon_k(t) - a_k \gamma_{f_k,t_k}(t) \qquad (5)$$

Finally, for $k = K$ we yield Equation 1.

By varying K within Equation 1, the SNR values of the spike code corresponding to the sparsity of the representation can be changed. Increasing K increases the SNR of the spike code. Small K (high sparsity) decreases the SNR.

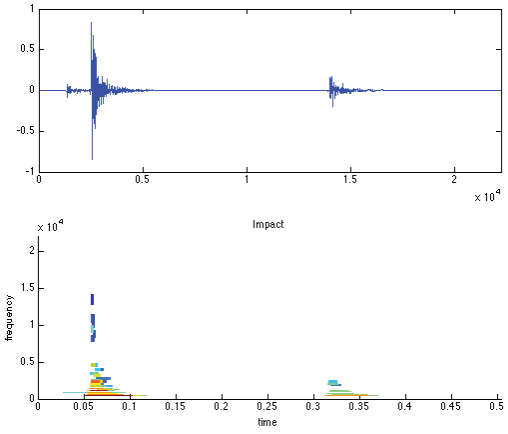

Figure 2. For an impact sound, we display the sound wave and the spike code. This picture clearly indicates how the spike code captures the skeleton of the sound.

Figure 2 shows the wave form and the spike code representation of an *impact* sound. In this example, a Gammatone filter bank of $M = 256$ filters is used for generating the spike code consisting of $K = 32$ spikes. Note that salient areas in the wave of this sound is coded with more spikes than other areas. This indicates that the spike representation captures the signal characteristics properly.

3 THE SONIC SPACE, SONIC LANDMARKS AND GUI

The sonic landscape is organized as a 2D space in which reference sounds (sonic landmarks) are located. The organization of the reference sounds may rely on perceptual criteria, derived on a statistical basis (e.g., clustering), or may be arbitrarily decided by the user.

The sonic landmarks in the sonic space form a 2D scatter points set. The sound designer may chose to generate a new

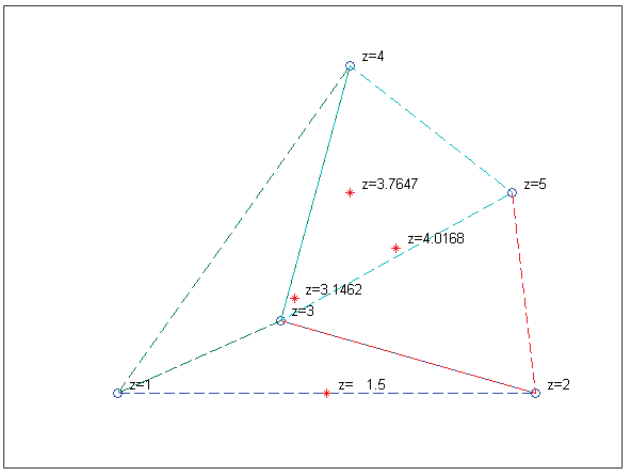

Figure 3. Scatter points interpolation: Delaunay triangulation (dashed lines) on a set of data points (circles), and four interpolated points (stars).

sound in the vicinity of a set of sonic landmarks inspired by their sonic properties. Depending on the coordinates of the new position indicated by the user, the new set of synthesis parameters is generated through an interpolation scheme accounting for the neighboring sonic landmarks. Consider a set of landmark points

$$\{(x_1, y_1, \mathbf{z}_1), (x_2, y_2, \mathbf{z}_2), \ldots, (x_n, y_n, \mathbf{z}_n)\}$$

where (x_i, y_i) is the position in the 2D space, and $\mathbf{z}_i$ is the value of the synthesis parameters vector. A convenient interpolation scheme for this class of problems is based on the following steps (for sake of simplicity, we consider a scalar parameter, z):

- Compute a grid of triangles connecting the scatter points together. One possibility, which is used here, is the Delaunay triangulation.
- For a new interpolated point, find the three vertices (x_{i1}, y_{i1}, z_{i1}), (x_{i2}, y_{i2}, z_{i2}) and (x_{i3}, y_{i3}, z_{i3}) defining the triangle in which the point is confined and compute the coefficients A, B, C, D for the equation of the plane defined by the triangle $Ax + By + Cz + D = 0$.
- Finally, compute the interpolated value through linear interpolation: $f(x, y) = -\frac{A}{C}x - \frac{B}{C}y - \frac{D}{C}$

Figure 3 gives an example of such interpolation for a small set of data points (circles) and a scalar z parameter. The interpolated data are depicted as stars.

A graphical user front-end (GUI) was implemented in Matlab and is shown in Figures 4 and 5. Its main frame represents the 2D sonic space with the sonic landmarks located in pre-defined spots (the figures show a configurations in which the landmarks were disposed as vertices of a rectangle). The location of the reference sounds can be arbitrarily decided and is specified in a configuration file loaded ad the beginning of each session. Once the landmarks sounds have been loaded and the auditory representation computed, the user can perform a number of actions, including:

- Creating a new sound by choosing with the mouse a position in the proximity of the sonic landmarks whit desirable characteristics
- Listening to landmarks and new sounds by clicking on the spike plot
- Deleting newly generated sounds which are not of interest for the user
- Saving the parametric setting of a new sound as presets in the XML-based format used by the SDT GUI
- Checking network connectivity with the Max/Msp or Pd server running the SDT sound synthesis engine

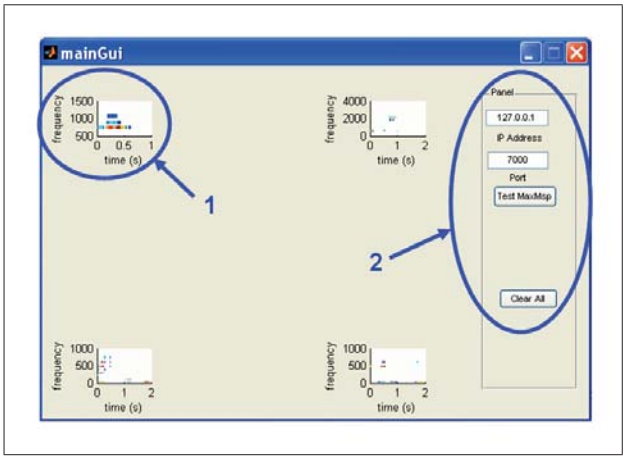

Figure 4. Matlab GUI. Spike representation of four sounds (sonic landmarks) representative of the entire sound space generated by means of the SDT friction model. By clicking in the graphical area around a spike representation it is possible to listen to the corresponding sound (see the area highlighted by circle 1). Circle 2 shows the control panel for the communication with the SDT model in Max/MSP

The spike-based GUI is connected through an OSC network layer to a Max/Msp or Pd server running the SDT sound synthesis engine. When the user confirms the position of a new sound to be generated, the GUI starts the communication with the SDT server, proceeding through the following steps: first, the new interpolated parameters are sent to the server, which updates the controller values; when done, the SDT synthesis engine is started and an auidio file is generated; finally, the GUI loads the audio file, generates the

new auditory representation, and plot the result in the 2D sonic space.

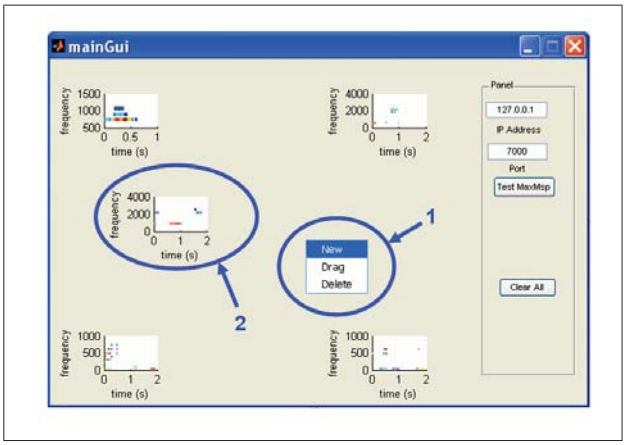

Figure 5. Creation of new sounds. By clicking in any part of the Matlab GUI it is possible to start the computation of a new sound, whose SDT synthesis parameters are calculated according to the graphical position with respect to the four sonic landmarks (see circle 1). The interpolation of the parameters of the four sonic landmarks for the creation of a new sound is linear. The new parameter set is stored as an SDT preset in xml format and acquired by the SDT model in Max/MSP. Circle 2 highlights the Spike representation of a new sound ready generated to be played in Max/MSP.

4 INTERPOLATION EXAMPLES

This section presents the result of a sound design experiment in which six reference sounds (the sonic landmarks) where organized in the 2D space as shown in Figure 6 (the spike representation of the reference sounds, and the Delaunay triangulation generated with this configuration, are shown). Six new sounds were generated by choosing six interpolation positions (depicted as red stars). The synthesis sounds resulting from the interpolated parameters, converted into the spike-based representation, are represented in Figure 7.

Perceptually, the results are in good agreement with the user's expectations, and the dynamical and spectral characteristics are recognized as actually deriving from the characteristics of the neighboring landmarks. The audio files corresponding to the sound landmarks and to the interpolated sounds are available for download at the link provided in the footnote [1] .

[1] http://mordente.sci.univr.it/∼carlodrioli/SoMuCo09/Experiments.htm

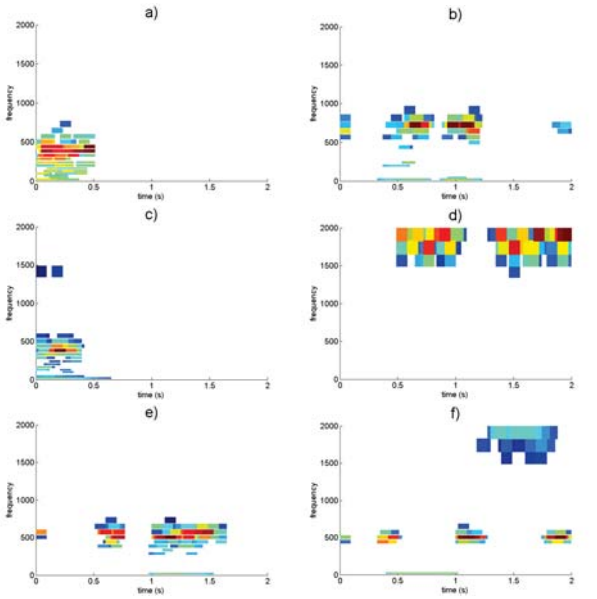

Figure 7. Interpolation example: new sounds (a), b) and c)) are generated by interpolation of points in the same triangle; the new sounds d), e) and f) are located in three different triangles

5 CONCLUSIONS

These results show that the visualization platform helps the sound designer to easily navigate through the soundscape and to find the desired sound fastly. In a following step, this visualization platform will be extended to perform a similar navigation within the soundscape towards a pre-defined (eg. recorded) sound with certain characteristics automatically. A probabilistic method will be defined to model the distribution of the input parameters of the sound model. Given the pre-defined sound, this method will automatically find the optimal input parameters for the sound model, which will produce the closest possible sound to the given sound, in terms of a spike distance. Hence, a distance measure will be defined to measure the distance between the spike code of the pre-defined sound and the spike code of the current sound produced by the sound model.

6 ACKNOWLEDGEMENTS

This work is part of the research carried out within two EU funded project: CLOSED (Closing the Loop of Sound Evaluation and Design) and NIW (Natural Interactive Walking).