Physics-Based Spike-Guided Tools for Sound Design

Kamil Adiloglu, Carlo Drioli, Pietro Polotti, Davide Rocchesso, Stefano Delle Monache

PHYSICS-BASED AND SPIKE-GUIDED TOOLS FOR SOUND DESIGN

Kamil Adiloğlu,
METISS group IRISA-INRIA
Rennes Cedex, France
kamil.adiloglu@irisa.fr

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

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

ABSTRACT

In this paper we present graphical tools and parameters search algorithms for the timbre space exploration and design of complex sounds generated by physical modeling synthesis. The tools are built around a sparse auditory representation of sounds based on Gammatone functions and provide 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 for new sounds. The sonic landmarks can be either prototype synthetic sounds chosen by the user or they can be automatically derived by using clever parameter search and clustering algorithms. The proposed probabilistic method in this paper makes use of the sparse representations to model the distance between sparsely represented sounds. A subsequent optimization model minimizes those distances to estimate the optimal parameters, which generate the landmark sounds on the given auditory landscape.

1. INTRODUCTION

The state-of-the-art sound synthesis techniques offer a high degree of naturalness and expressiveness. Sometimes, however, this comes at a price of a relevant 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 simplified equations of the underlying physical laws. As a result, the parameters involved in the modeling become in principle easy to understand, because they have real counterparts, and easy to control, because our sensorial experience mediates 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, which makes the parameters inaccessible or unfeasible to non specialists and, anyhow, results in complex control mapping issues. 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 parameter tuning of synthesis tools in general. Such tools are of particular interest in the case of physical modeling audio synthesis.

In this paper, we propose a graphical tool equipped with clever parameter optimization algorithms for the timbre space exploration and interactive design of complex sounds by physical modeling synthesis. We exploit a sparse auditory representation of sounds based on Gammatone functions. This representation provides the designer with 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. A supervised probabilistic framework makes use of the sparse representations to model the distance between two sounds. These distances are incorporated in a subsequent probabilistic estimation method based on the expected loss principle to select the optimal parameter values to support the sonic landmark paradigm. The graphical interface incorporates the auditory representation of the landmarks located 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. In a related work, Momeni and Wessel proposed a tool to construct and access multidimensional parameter spaces through 2D (or 3D) interfaces [4]. The interpolation between a set of examples was obtained by a mixture of gaussian kernels whose weight and width could be adjusted. The user could position the example objects in arbitrary positions and visual feedback was implicit in the system. Although similar in some extent, our proposed control layer relies on auditory representations of sonic landmarks, and it is especially suited to exploring wildly non-linear sonic spaces.

We rely on a sound synthesis framework based on a class of physical models for everyday sounds, which includes low level events (impacts and friction) and high-level events (bouncing, breaking, rolling, crumpling). This Physically-based Sound Design Tools package (SDT from now on) has 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))[5, 6].

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 this section, some sound design examples obtained with the tool are also illustrated. The subsequent section introduces the clever parameter search methods. After presenting some optimization results, the paper concludes in Section 5 with final remarks 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 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 mode frequencies, decay factors and gains. 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. Further details on the friction model can be found in [5], 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 of signal 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 [7], a sound signal $x(t)$ can be approximated as a linear combination of K filter functions, the so-called spikes. In this study, we used 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 [8]. Hence, these filter functions have a biological background, which makes them more promising for perceptually relevant tasks. Furthermore, it has been shown that the gammatone filters are highly efficient for natural sounds including the environmental sounds. Even though we have a synthesis model and not natural recordings, the physical modeling paradigm to model environmental sounds (in our case the friction model) encouraged us to use the gammatone filters as the basis of our decomposition. We used the implementation within Slaney's auditory toolbox [9] for the gammatone filterbank, which generates filters with the center frequencies and bandwidths calculated depending on the equivalent rectangular bandwidth (ERB) model [10].

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.

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

For simplicity of graphical representation, we show the spike code of the simplest sound generated by the SDT algorithms, i.e. an impact sound in Figure 1. In this example, a Gammatone filter bank of 256 filters is used for generating the spike code consisting of $K = 32$ spikes. Note that salient areas in the wave of this sound are coded with more spikes than other areas. In another example, we show a real friction sound in Figure 2. Since the friction sound is much longer than the previous example with an impact sound,

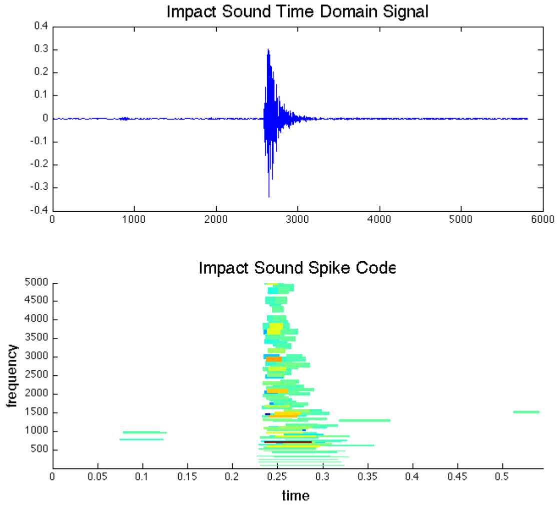


Figure 1: For an impact sound, we display the sound wave and the spike code. This picture indicates how the spike code captures the skeleton of the sound.

we used the same filterbank but used 256spikes to code the friction sound. These two examples indicate that the spike representation captures the signal characteristics properly.

2.2.1. Dissimilarity Between Spike-Coded Sounds

The spike code representation offers a totally new paradigm in observing similarities between sounds. This representation enables to solve the problem of detecting similarities by calculating these similarities without destroying the original structure of the pattern [11]. Basically, the distance between two spikes of different spike codes is computed as step. In the following step, the total distance between the given spike codes is calculated by taking the minimum of the sums of spike distances of all possible pairwise assignments. Intuitively, this method measures the minimal effort to transform one spike code into the other in terms of the single spike distance.

The distance $d_s(s, s')$ between two spikes $s = (t, f, a)$ and $s' = (t', f', a')$ is composed of three individual distances, namely the distance $d_t(t, t')$ between time offsets, $d_f(f, f')$ between center frequencies, and $d_a(a, a')$ between amplitudes:

$$d_s(s, s') = \tau d_t(t, t') + \phi d_f(f, f') + (1 - \tau - \phi) d_a(a, a'), \quad (2)$$

with $\tau, \phi \geq 0, \tau + \phi \leq 1$. Parameters τ and ϕ allow to emphasize either the temporal, spectral, or amplitude aspect, while guaranteeing that the weights sum up to 1.

The temporal distance is calculated linearly, simply by taking the absolute value of the time difference between the time offsets of the corresponding spikes. In order to calculate the frequency distance, we consider logarithmic frequencies, according to the Weber-Fechner law. The amplitude difference is obtained by dividing each amplitude by the maximal amplitude of both sounds, thereby considering the volume differences as well.

Given the spike codes $\mathbf{s} = \{s_1, \ldots, s_K\}$ and $\mathbf{s}' = \{s'_1, \ldots, s'_K\}$ of two sounds and given the distance measure

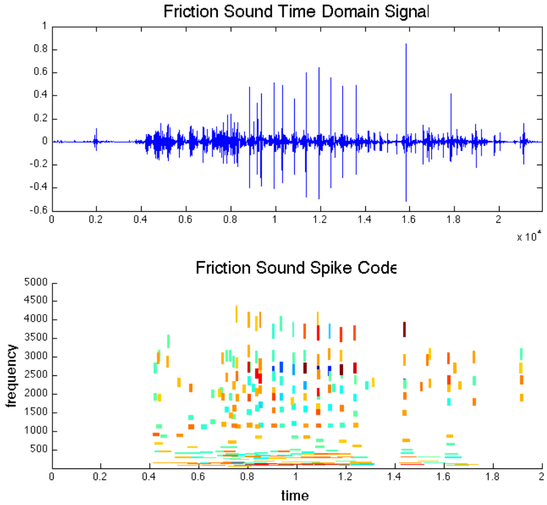


Figure 2: For a real friction sound, we display the sound wave and the spike code. Similar to the impact sound, in this picture we indicate how the spike code captures the skeleton of the friction sound.

$d_s(s_i, s'_j)$ for two spikes $s_i \in \mathbf{s}, s'_j \in \mathbf{s}'$, we can define a dissimilarity $D(\mathbf{s}, \mathbf{s}')$ between two sounds by establishing a bijection between s and s$'$. For a permutation $\mu \in S_K$ we assign to each spike in s exactly one spike in s$'$, so that $s_i \rightarrow s'_{\mu(i)}$. Then we define the dissimilarity between two spike-coded sounds as the normalized sum of the distances between the K corresponding spikes:

$$D(\mathbf{s}, \mathbf{s}') = \frac{1}{K} \sum_{i=1}^{K} d_s(s_i, s'_{\mu(i)}), \quad (3)$$

with μ minimizing the dissimilarity:

$$\mu = \operatorname{argmin}_{\mu \in S_K} D(\mathbf{s}, \mathbf{s}'). \quad (4)$$

In order to minimize μ in Equation (4), we propose a method, which is based on the alignment of the spikes to minimize the total distance. This method stems from the graph theory, and is called the Hungarian algorithm.

2.2.2. Graph Matching

Instead of solving the problem of minimizing the assignment μ in Equation 4 directly, we consider the spike codes of two given sounds as two point graphs combined in a bipartite graph. In this constellation, the vertices are the spikes $s_i \in \mathbf{s}, s'_j \in \mathbf{s}'$ of two spike-coded sounds $\mathbf{s}, \mathbf{s}'$, where each sound corresponds to a disjoint subgraph of the bipartite graph, and the weights w_{ij} (similarities) between them are derived from the distance d_s : $w_{ij} = 1 - d_s(s_i, s'_j)$ with $\mu(i) = j$. This consideration converts the problem into a combinatorial one of finding a perfect matching of the weights in a bipartite graph by the Hungarian algorithm [12]. We consider a matching to be a subset of the edges of the given graph containing each vertex only once. In a perfect matching, every vertex within the graph is adjacent to one and only one edge. The Kuhn-Munkres Theorem [12] guarantees the convergence of the algorithm to a perfect matching.

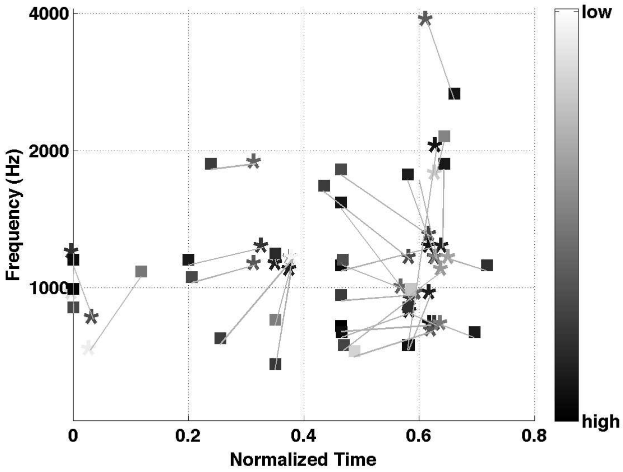


Figure 3: This figure shows how the Hungarian algorithm matches individual spikes of two spike coded pouring liquid sounds to minimise the distance between them. The spikes of the sounds are indicated by a different symbol ('*','■'). The light intensities of these symbols encode the amplitudes of the spikes.

Figure 3 shows an example about the matching results of two spike coded sounds performed by the Hungarian algorithm. This plot indicates the matching results between two sounds of the same sound class, which exemplifies that the distances between spike patterns of same sound classes are in general close to each other.

3. THE SONIC SPACE, SONIC LANDMARKS AND GUI

The sonic landscape is organized as a 2D space, where reference sounds (sonic landmarks) are located. The organization of the reference sounds may rely on perceptual criteria, on multidimensional scaling, they can be 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 choose to generate a new 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. Convenient interpolation schemes for this class of problems are often based on a preliminary triangulation step, aimed at creating a grid of triangles connecting the scatter points together. One possibility, which is used here, is the Delaunay triangulation. A new interpolated point is then computed by first finding the three vertices of the triangle in which the point is confined, and then applying a linear interpolation formula based on the equations of the plane defined by the three vertices.

Figure 4 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 Figure 5. Its main frame represents the 2D sonic space with the sonic landmarks located in pre-defined spots (the figures show a configuration, where the landmarks are at the vertices of a rectangle). The location of the reference sounds can be

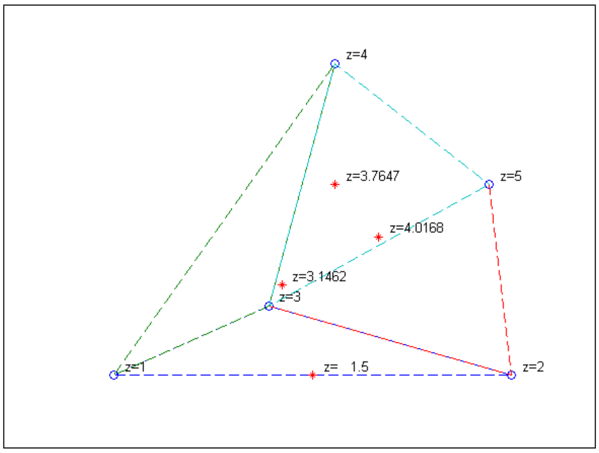


Figure 4: Scatter points interpolation: Delaunay triangulation (dashed lines) on a set of data points (circles), and four interpolated points (stars).

arbitrarily decided and is specified in a configuration file loaded at the beginning of each session. Once the landmark sounds have been loaded and the auditory representation computed, the user can perform a number of actions, including creating a new sound by pointing to a position in the proximity of the sonic landmarks with 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, and checking network connectivity with the Max/MSP or Pd server running the SDT sound synthesis engine.

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 audio file is generated; finally, the GUI loads the audio file, generates the new auditory representation, and plots the result in the 2D sonic space.

The results of a sound design experiment with six reference sounds (the sonic landmarks) organized in the 2D space are shown in Figure 6 on the left. 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 6 on the right.

Perceptually, the results match the expectations, and the dynamic and spectral characteristics are recognized as actually deriving from the characteristics of the neighboring landmarks [1].

[1] The audio files corresponding to the sound landmarks and to the interpolated sounds are available for download at http://mordente.sci.univr.it/∼carlodrioli/Dafx10/Experiments.htm

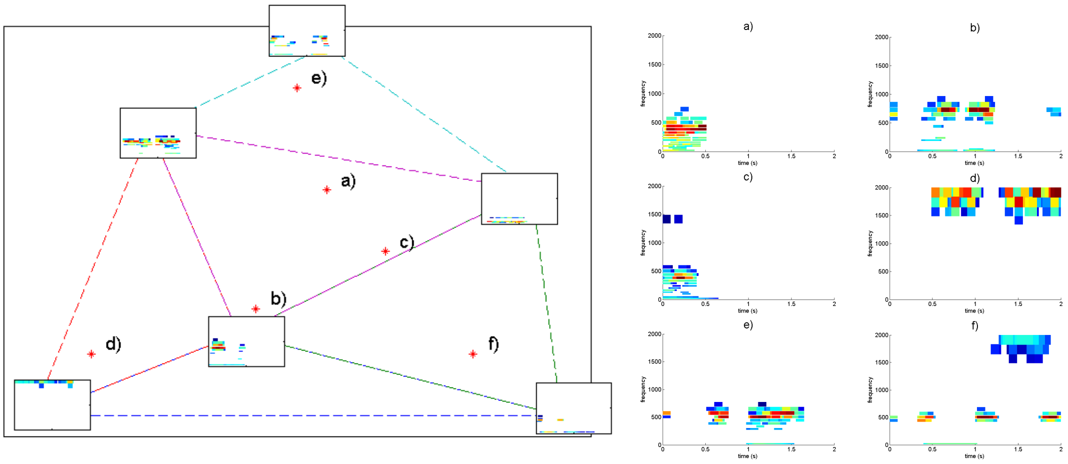


Figure 6: Interpolation example: the new sounds a), b) and c) are generated by interpolation of points in the same triangle; the new sounds d), e) and f) are located in three different triangles.

4. PARAMETERS SEARCH ALGORITHMS

The sonic landmarks are the representatives of a physical sound synthesis model. They not only show the diversity of the sounds, which can be generated by this model, but can also be used as starting points to generate new sounds, which are mainly interpolations of several of them (See Section 3).

Therefore defining the sonic landmarks constitute a very useful step in the sound design process using a synthesis model. As mentioned earlier (Section 3), they can be real sounds or results of some statistical methods like the MDS or clustering. For the landmarks, defined by using such methods, we do not know the input parameter values of the physical sound synthesis model, which would generate them. The ad-hoc approach the sound designers generally utilize is the trial and error method. This generally yields sub-optimal solutions and takes a long time. In this paper, we propose a Bayesian framework for the parameter optimization of a physical sound model problem for a given sonic landmark for which the input parameter values are unknown. From this point on, we call the landmarks as reference sounds. For this purpose, we make use of the spike code and the corresponding distance measure presented in Section 2.2. The reference sound and each candidate sound are coded by using the spike code. Subsequently, the distance between the spike codes of these two sounds is calculated. We use Gaussian Processes [13] [14] to model this distance function. We combine this model with a clever optimization method to select the next candidate sound, which is expected to improve the quality of the candidate sounds by minimizing the distance function modeled by the Gaussian process. In each iteration, the real distance between the reference and candidate is calculated. By using this distance, the Gaussian process is trained once again with the real distance values calculated so far. The newly trained Gaussian process is incorporated in the optimization method to select the next candidate sound. This procedure is repeated as long as the distance between the reference and the candidate is above a certain threshold value.

4.1. Gaussian Processes

A Gaussian process (GP) is a stochastic process containing any finite number of random variables having a joint Gaussian distribution. A GP is defined by its mean $m(x_i)$ and covariance functions $k(x_i, x_j)$. So, we write $f(x_i) \sim GP(m(x_i), k(x_i, x_j))$.

We use a zero mean prior GP with squared exponential covariance function given by

$$k(x_i, x_j) = \sigma_f^2 exp\{-\frac{1}{2}(x_i - x_j)^T M^{-1}(x_i - x_j)\}, \quad (5)$$

where σ_f^2 is the variance of the function we model and M is a diagonal matrix of the length scales of the Gaussian components.

In real world applications, it is generally not possible to obtain the real function values $f(x)$ but their noisy versions $y = f(x)+\epsilon$. Pursuing this idea, we assume that our observations are noisy. We assume additive noise ϵ with zero mean and variance represented by σ_n^2. Therefore our final covariance function is written as

$$cov(x_i, x_j) = k(x_i, x_j) + \sigma_n^2 \delta_{ij}, \quad (6)$$

where δ_{ij} is the Kronecker delta, which is one if $i = j$, otherwise zero.

The goal of modeling the function f by using the GPs is to be able to make predictions about the data points, that have not been observed yet by simply incorporating the data points observed so far.

In order to be able to make predictions about the function values of the unobserved data points, we need to make use of the observed data points and get to the posterior distribution.

The function values corresponding to the data points can be sampled from the joint distribution by evaluating the posterior mean and covariance functions. Hence, for each new data point, the posterior distribution is re-calculated and the function value for the new data point is sampled from this distribution. However, we do not know the optimal values for the so-called hyper parameters of the posterior distribution.

The hyper parameters of the posterior distribution of the function values are given by $\theta = \{\sigma_f, M_{ii}, \sigma_n\}$, where M_{ii} represents the length scales of the Gaussian components and $i = 1 \ldots D$ (See Equation 5). In order to determine the optimal hyper parameter values, we compute the probability of the data given the hyper parameters, which we call the marginal likelihood.

Therefore, after each new data point, the marginal likelihood is maximized to model the function values regarding the new data points in a better way. Subsequently, the posterior distribution is utilized to estimate the function value of the new data point. Note that our aim is not to estimate the function value as close as possible to the real function value, but to find the data point iteratively, which minimizes this value. Therefore, we need a clever strategy to select the next data point. However, it is still unclear how we select the new data points. For this purpose, we define the concept of expected improvement using the GP we described here.

4.2. The Expected Improvement Function

The GP we defined in the previous section models the distance function in a proper way. However, we do not want to model the distance function at first place, but rather try to find its minimum with respect to a given reference. In particular, if the evaluation of the function is a time consuming operation, a clever sampling method is needed. Therefore, we need a selection criterion to sample the data space cleverly to come closer to this minimum as fast as possible. Osborne et. al. [15] suggest a probabilistic method for this sampling problem, in particular for the minimization of such time consuming functions.

Imagine that we have only one iteration left to improve the quality of our estimation. In our model, the best estimation is the closest sound generated by the physical sound model given the reference sound. Suppose that $(\mathbf{X_0}, \mathbf{f_0})$ are the data points and their corresponding function values evaluated so far. The best estimation among these function values is defined as $\eta = \min \mathbf{f_0}$. Hence, the evaluation of the last data point $(\mathbf{x}, f)$ could improve the quality of our estimation in the following manner:

$$\lambda(f) = \begin{cases} f; & f < \eta \\ \eta; & f \geq \eta \end{cases} \tag{7}$$

Hence, the expected improvement [15] is simply $\min(f, \eta)$ either the previous minimum or the function value of the last data point. Given our GP over the function, which we try to minimize, we can define the total expected improvement given a data point x as follows:

$$\Lambda(\mathbf{x}|\theta) = \int \lambda(f) p(f|\mathbf{x}, \theta) df, \tag{8}$$

where we use the predictive distribution of the function values. After some elementary calculations, the total expected improvement function becomes the following form:

$$\Lambda(\mathbf{x}|\theta) = \eta + (\mu - \eta)\Phi(\eta; \mu, C) - C\mathcal{N}(\eta; \mu, C), \tag{9}$$

where Φ and $\mathcal{N}$ are the cumulative probability distribution function (cdf) and the probability density function (pdf) of the function values respectively. The mean μ and the covariance C are the mean and the covariance of the posterior GP defined in the previous section.

Note that the expected improvement function given in Equation 9 decreases firstly when the mean value μ becomes lower than the current minimum η, and secondly when the covariance C becomes larger. The first case implies exploitation and the second case exploration. In other words, the expected improvement function exploits the region, where the current minimum is as well as it explores other regions with high covariance. Consequently, this function defines a balance between these two concepts, which is also known as the exploitation / exploration dilemma.

The data point for which the expected improvement function is the lowest is the next data point for the next evaluation. Hence, we need to minimize this function. Fortunately, the expected improvement function given in Equation 9 is easy to evaluate and differentiable infinitely many times. Therefore, there are many methods to minimize this function with respect to the data point x by incorporating its gradient and Hessian. Here we used Matlab optimization toolbox to minimize the expected improvement function. Figure 7 shows an iteration of the expected improvement scenario. As one can easily see, the expected improvement function (line with the diamonds at the bottom of the figure) is lower for the regions, where the covariance of the GP is large as well as for the regions, where the current minimum is. So the expected improvement function indeed balances the exploration of the unvisited regions with the exploitation of the observed regions, which potentially contain the global minimum.

4.3. Tests and Results

The parameter optimization model has been tested with the synthesized sounds at first, which were synthesized by the same physical model. For this purpose, we made use of the preset parameter values. In the physical modeling paradigm, different parameter combinations can yield similar results. This means that there are many local minima for a given reference sound. Therefore, we wanted to see whether the optimization model can reveal the parameter proximity by using the spike based similarity measure. The experiments performed show that the model can indeed approximate the parameter values, which have been used for generating the reference sound.

The physical model we wanted to optimize is a highly complex model with a large quantity of input parameters. These parameters not only control the spectral content of the generated sounds but also their temporal content. Furthermore, because of the stochastic nature of the model, the temporal content is not static, but it changes randomly in a certain range without diverging from the group of sounds which the model is supposed to generate. In other words, the physical model generates not exactly the same sound for the same parameters. Since the spike based similarity measure captures the temporal similarity as well as the spectral similarity, these random changes in the temporal content of the generated sounds were a problem. Therefore, we restricted the temporal diversity of the sounds at first and concentrated ourselves on the spectral content. Including the parameters controlling the temporal behaviour of the sound into the optimization procedure could be a possibility. Unfortunately, these parameters are not scalars, but mainly mappings between two functions. We simply kept the temporal pattern of the sounds to be generated by the model constant. By doing this, we made the model generate very similar sounds for the same parameter values, but not exactly the same sounds.

Since the physical model has a large number of input parameters, we wanted to increase the complexity of the optimization model gradually. Thus, we did not cover the parameter domain completely but started with a smaller number of parameters to optimize. For these tests, the other parameters were kept constant.

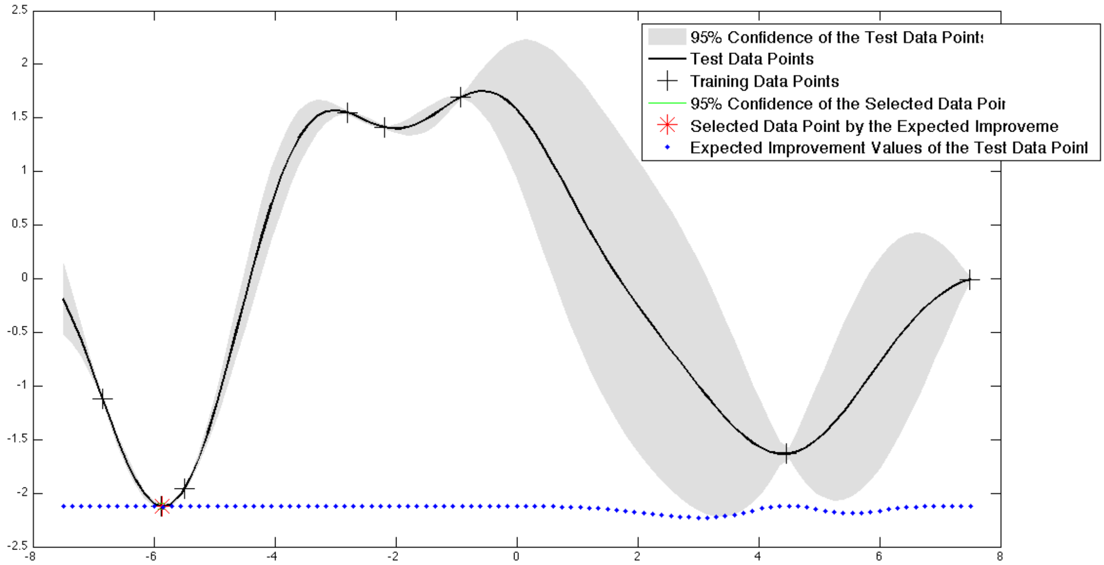


Figure 7: One iteration performed using the expected improvement function is shown. The line with diamonds shows the expected improvement values of the test data points. The asterisk on this line is the data point selected by minimizing the expected improvement function. The crosses are the training data points observed so far. The solid line is the test data points, which are evaluated with respect to the current GP estimation of the function to be modeled. The shaded area around this line is the confidence interval of the GP.

Hence, we started with optimizing one parameter only, namely the frequency. Afterwards, we performed 3D optimization with the parameters frequency, gain and delay. Finally, we performed six dimensional optimization with the additional parameters bristle stiffness, bristle viscosity and surface roughness. Several results of these optimization experiments are shown in Table 2.

Real Values	Estimated Values
65-12-105	66-59-105
65-12-105	68-64-56
50-54.5-50-36.6-65.8-0	50.5-83.2-38.4-23.9-66.5-42.9
50-54.5-50-36.6-65.8-0	49.5-45.5-63.3-7.7-77.9-0
50-54.5-50-36.6-65.8-0	46.8-55.3-55.7-28.8-71.4-0
100-61.5-100-64.2-89.4-83.7	100-66.6-100-66.6-100-100
100-61.5-100-64.2-89.4-83.7	100-53.5-100- 71.3-100-100
0.8-95-0-50-11.5-50	0-100-0-100-26.4-55.7
34.15-96.75-70-50-52.5-76	0-100-100-46.5-100-0

Table 2: This table shows three and six dimensional optimization results. The column on the left hand side shows the parameter values of the given sonic landmark. The second column on the right hand side shows the estimated parameter values.

In the results of the three dimensional parameters, the order of the parameters is given as frequency, decay, gain. For the six dimensional results, the first three parameters are like in the three dimensional results. The last three parameters are bristle stiffness, bristle viscosity, surface roughness. All the parameters shown in Table 2 are normalized to the interval $[0 : 128]$. As one can easily see, experiments repeated with the same reference sound yield similar results.

For the last results in Table 2, we show the progress in Figure 8 by plotting the spike codes of the sounds suggested by the expected improvement function. The fifth sound is the optimal approximation found by the method. The sixth sound next to the offered optimal is the sound landmark for which we run the optimization. As one can see, the similarity to the sound landmark increases gradually form iteration to iteration. However, please recall that the similarity of the suggested sounds does not become larger after each iteration, because the algorithm balances the exploration with the exploitation. While exploring some unobserved regions, the suggested sound can be less similar to the sound landmark than the most similar sound detected so far. However in the end, the detected optimal is likely to be the global optimal because of this reason.

In the near future, these results will be extended with real world sounds. Testing the optimization model against some preset values still shows the capability of the model to some extent. However testing it against real world sounds is mandatory to observe the real performance of the model and to see its pros and cons completely. For a real sound, there will not be an exact match of the parameters. Therefore, the evaluation of the model for a real world sound should be done in another way. Human judgements can be used like in a psychoacoustic experiment. The SNR between the real world sound and the estimated sound can give an objective criterion about the quality of the estimated sound.

In order to test the model, the temporal pattern of the real world sound should be mapped to the temporal parameters of the model. This should be done by determining the temporal pattern of the real world sound by means of proper methods as envelope-follower, pitch trackers and so on beforehand and train the model again only for the spectral content.

When the framework will be complete, it will be possible for the designer to pick up an environmental sound or to produce an imitation that could be translated into a sonic landmark, automatically chosen as the closest synthetic realization that can be produced by the given sound model.

5. CONCLUSIONS

The initial results show that the visualization platform in combination with a probabilistic framework helps the sound designer to automatically navigate through the soundscape and to rapidly find the desired sound.

The experiments we performed so far contain preset sounds of the physical model itself. Nonetheless, these results give an idea about the capabilities of the optimization framework we propose in this paper. Furthermore, in the near future, the model will be tested against real world sounds to see the simulation and optimization capacity of the model in much wider sonic contexts.

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), FP6-NEST-PATH no. 29085, and NIW (Natural Interactive Walking), FP7-ICT-2007 FET Open no. 222107.

7. REFERENCES

[1] Cumhur Erkut, Vesa Välimäki, Matti Karjalainen, and Henri Penttinen, ," in *Physics-based Sound Synthesis*, Pietro Polotti and Davide Rocchesso, Eds., vol. Sound to Sense, Sense to Sound. A State of the Art in Sound and Music Computing, chapter 8, pp. 303–343. Logos Verlag, Berlin, 2008.

[2] Vermont: Destiny Books Rochester, Ed., *The Soundscape: Our Sonic Environment and the Tuning of the World.* Rochester, Vermont: Destiny Books, 1994.

[3] Berry Truax, *Handbook for Acoustic Ecology*, ARC Publications, 1978.

[4] A. Momeni and D. Wessel, "Characterizing and controlling musical material intuitively with graphical models," in *Proceedings of the New Interfaces for Musical Expression Conference*, Montreal, Canada, 2003.

[5] Davide Rocchesso and Federico Fontana, Eds., *The Sounding Object*. Mondo Estremo, 2003.

[6] Stefano Delle Monache et Al., "Sound design tool: Algorithms for ecologically-founded sound synthesis," *submitted to ACM SIGCHI Designing Interactive Systems DIS2010 Conference*, 2010.

[7] E. Smith and M. S. Lewicki, "Efficient coding of time-relative structure using spikes," *Neural Computation*, vol. 17, pp. 19–45, 2006.

[8] R. D. Patterson and J. Holdsworth, "A functional model of neural activity patterns and auditory images," *Advances in Speech, Hearing and Language Processing*, vol. 3, pp. 547–563, 1996.

[9] M. Slaney, "A matlab toolbox for auditory modeling work," *Interval Research Corporation*, 1998.

[10] B. R. Glasberg and B. C. J. Moore, "Derivation of auditory filter shapes from notched-noise data," *Hearing Research*, vol. 47, pp. 103–138, 1990.

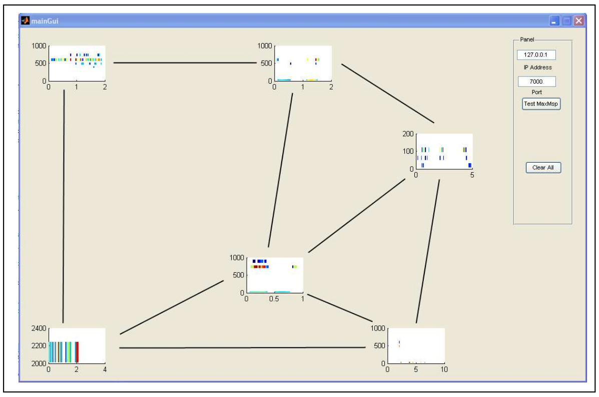
[11] K. Adiloglu, R. Annies, E. Wahlen, H. Purwins, and K. Obermayer, "A graphical representation and dissimilarity measure for basic everyday sound events," *IEEE Transactions on Audio, Speech and Language Processing*, 2010, submitted.

[12] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, pp. 83–97, 1955.

[13] C. E. Rasmussen, "Gaussian processes in machine learning," in *Advanced Lectures on Machine Learning, Revised Lectures*, O. Bousquet, U. von Luxburg, and G. Rätsch, Eds., pp. 63–71. Springer Verlag, Heidelberg, Germany, 2004.

[14] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*, MIT Press, Cambridge, MA, USA, 2006.

[15] M. A. Osborne, R. Garnett, and S. J. Roberts, "Gaussian processes for global optimization," in *3rd International Conference on Learning and Intelligent Optimization (LION3)*, Trento, Italy, 2009.



127.0.0.1
IP Address

7000
Port

Figure 5: 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. On the left, a control panel is provided for the communication with the SDT model in Max/MSP

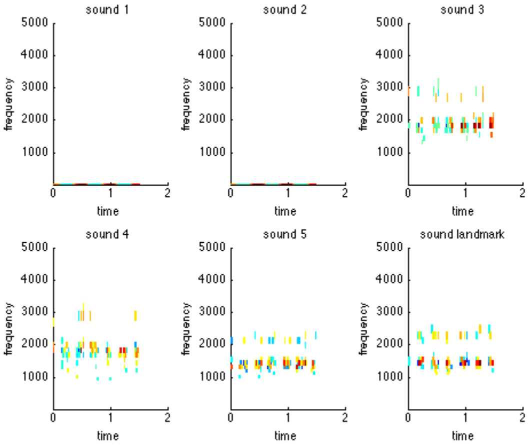


Figure 8: This figure shows the progress of one optimization experiment. The spike codes of the shown sounds 1-5 are the suggested sounds from the expected improvement function in the consecutive iterations. The last sound is the sound landmark.