X Congresso Brasileiro de Inteligência Computacional (CBIC'2011), 8 a 11 de Novembro de 2011, Fortaleza, Ceará
© Sociedade Brasileira de Inteligência Computacional (SBIC)

A GENETIC ALGORITHM FOR PARAMETER ADJUSTMENT OF AN ARTIFICIAL REVERBERATION MODEL

Fabrício V. Ambrósio, Heitor S. Lopes, Carlos R. Erig Lima
Electronics Department, Federal University of Technology - Paraná (UTFPR)
alzapua@gmail.com, {hslopes,erig}@utfpr.edu.br

Resumo – Do ponto de vista computacional, a modelagem e simulação de reverberação artificial é uma tarefa bastante difícil que requer grande capacidade computacional. Este trabalho objetiva investigar a aplicabilidade de Algoritmos Genéticos (AG) para ajustar os parâmetros de um modelo de reverberador artificial recursivo. É apresentada uma descrição detalhada da metodologia empregada no problema, bem como os critérios de simplificação para torná-lo computacionalmente viável. A Curva de Decaimento de Energia (CDE) é utilizada para medir a similaridade perceptual entre a resposta do modelo e a resposta ao impulso de uma sala real onde ocorre a reverberação, como parte da função de fitness do AG. Os resultados dos experimentos mostraram que o modelo proposto, juntamente com o AG, foram capazes de gerar uma CDE significativamente próxima da CDE correspondente à região de reverberação tardia da sala real.

Palavras-chave – Algoritmo genético, reverberação, áudio, processamento de sinais.

Abstract – From the computational viewpoint, modeling and simulation of artificial reverberation is a difficult task that requests high computational resources. This work aims at investigating the applicability of a Genetic Algorithm (GA) to adjust the parameters of a recursive artificial reverberation model. A detailed description of the methodology is provided, along with the simplification assumptions so as to make the problem computationally feasible. The Energy Decay Curve (EDC) is used to measure of perceptual similarity between the response of the model and the impulse response of a real room, like part of the fitness function of the GA. The results of experiments showed that the proposed model, adjusted by the the GA, was able to generate an EDC that closely followed the EDC of the late reverb region of the real room.

Keywords – Genetic algorithm, reverberation, audio, signal processing.

1. INTRODUCTION

Artificial reverberation consists in reproducing the acoustical response of a room when stimulated by sounds. The effect of reverberation adds life and sense of space, associated with the architecture and acoustic of concert halls [1]. However, from the computational viewpoint, modeling and simulation of artificial reverberation is a very difficult task, demanding expensive computational effort. The use of recursive filters, introduced by Schroeder [2], provide a low cost solution to generate multiple echoes. After that, several works proposed different solutions to the artificial reverberation problem. In recent literature, Toma et al. [1] present a description of algorithms for some early and late artificial reverberation

A Genetic Algorithm (GA) is a stochastic search method inspired by the mechanics of (Darwinian) natural selection and genetics [3]. GAs have been successfully used in a wide variety of applications in engineering and science [3], [4]. The use of Genetic Algorithms (GA) for searching the parameters of Infinite Impulse Response (IIR) filters was presented by [5]. Another GA approach, was also proposed by [6] as part of experiments with reverberation algorithms. The sparse GA application in artificial reverberation inspires a new work where we can design a linear digital system that is capable of simulating the Impulse Response (IR) of a room. In this case, a GA is used to automatically adjust the parameters of a recursive artificial reverberation model. Only a few works in recent literature use similar approach, such as Jot [7] and Bai [8].

The Impulse Response (IR) of a room can be divided into two regions: early reverb and late reverb. As described in [1], early reverb is related to the first sound reflections and has a typical duration of tens of milliseconds. Late reverb consists in a dense group of reflections that decays exponentially with time. This second region is the main responsible for the spatial impression of the room. An example of true room IR is shown in Fig. 1, and refers to the St. Bernard's Church (Enfield, Nova Scotia) IR (available at: $http://woxnerw.net/05\_01.rar$). The portion of higher amplitude in the beginning of this figure corresponds to the early reverb. The remaining of the waveform corresponds to the late reverb.

A possible implementation of an artificial reverberator is a Finite Impulse Response (FIR) filter, whose coefficients equal the desired IR. This method, however, has a very high computational cost. Consider, for instance, the IR shown on Fig. 1. This signal, sampled at 44.1 KHz, has a total of 107520 samples. Therefore, for a real-time implementation, it would be required to compute an iteration involving 107520 filter taps in 1/44100 seconds. That is, more than 4.741×10^9 computations per second.

To overcome this problem, a hybrid system can be used. In such a system, the short duration early reverb region can be implemented using FIR structures. The late reverb simulation, in turn, can be done by means of recursive IIR models. When compared to a method that employs only FIR structures, the computational cost of hybrid systems is considerably lower. Notice,

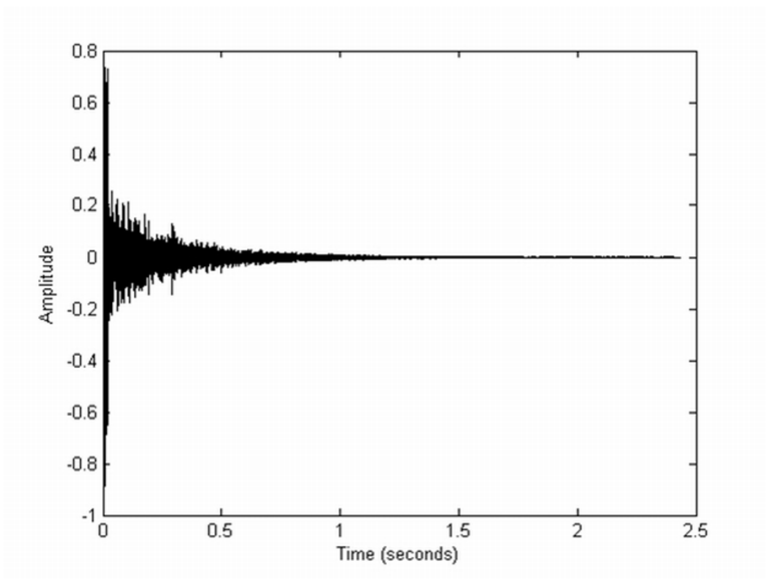


Figura 1: Impulse Response (IR) of a real room. This is the complete response (early reverb + late reverb).

however, that the response of a true room can have thousands of zeros and poles in the audio band [9]. Thus, when considering hybrid systems, one should aim at modeling relevant perceptual characteristics of the true IR rather than reproducing it with high fidelity.

An important fact about recursive models is that there is no definitive method for the adjustment of their parameters. Trial and error is the most employed method. Therefore, the application of a meta-heuristic method, such as GA, presents itself as a possible way of automating this process. Therefore, the objective of this work is to propose a method using genetic algorithms to adjust automatically the parameters of a recursive artificial reverberation model.

2 THE REFERENCE MODEL

In this work, only the late reverb region of the room's IR was considered. To simulate this IR part, a structure known as Feedback-Delay-Network (FDN) was selected. The general FDN model [7] is shown in Fig. 2. The parameters presented in this figure are: $x(z)$: input signal, $y(z)$: output signal, N: order of the FDN, b_1 to b_N: input gain constants, c_1 to c_N: output gain constants, d: constant gain applied to the direct signal path, m_1 to m_N: length (in samples) of the delay lines, $h_1(z)$ to $h_N(z)$: absorbent filters, $t(z)$: output filter, $a_{1,1}$ to $a_{N,N}$: coefficients of the feedback matrix.

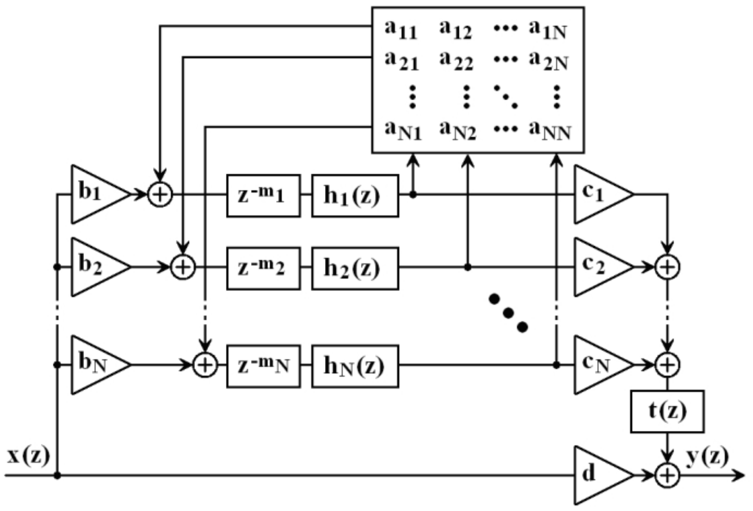


Figura 2: General model of a N^{th} order Feedback-Delay-Network (FDN).

In the absence of the absorbent and output filters, equations 1 and 2 completely define the FDN [1, 9, 10], where $s_i(n), 1 \leq$

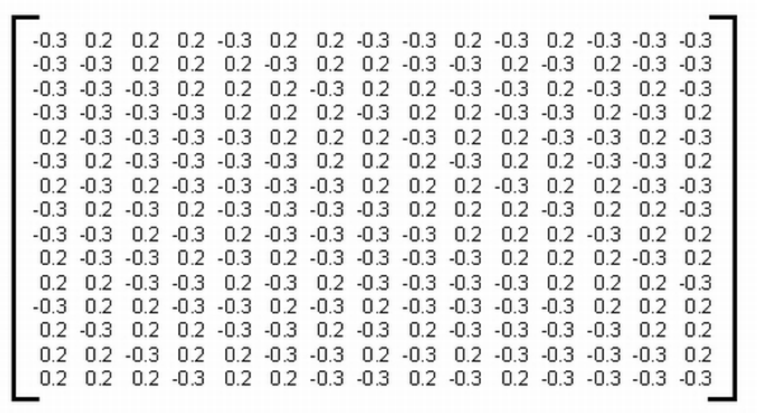
$i \leq N$, are the delay line outputs at sample time n:

$$y(n) = \sum_{i=1}^{N} c_i \cdot s_i(n) + d \cdot x(n) \tag{1}$$

$$s_i(n + m_i) = \sum_{j=1}^{N} a_{i,j} \cdot s_j(n) + b_i \cdot x(n) \tag{2}$$

The coefficients of the feedback matrix are directly related to the stability of the system. When $h_i(z)$ and $t(z)$ are not present, it is possible to choose the coefficients so that all poles are precisely located on the unity circle of the z-plane [1,7,9,10]. Consequently, the system becomes lossless and lays in the edge of stability, resulting in an output with infinite reverberation time. This is an important point about the FDN, since it implies that the structure itself is independent of the decaying characteristics of the generated waveform. Having such a lossless structure in hands, the absorbent filters $h_i(z)$ can then be inserted in order to shape the decay of the output. In general, each one of these filters are composed by a low-pass filter and a gain factor smaller than one. The output filter $t(z)$, in turn, could then be used to provide some kind of output compensation. In [7], for instance, it is employed as an output tone corrector.

In this work, a 15^{th} order FDN was selected. Its feedback matrix, obtained in [10], is shown in Fig. 3. Notice that this is a circulant matrix, that is, one row is equal to the above row circularly shifted one position to the right. With this matrix, the system is unconditionally stable if the gains of the absorbent filters are smaller than one. Hence, if this last condition holds, the remaining parameters can be freely chosen. It is worth to mention that a high order FDN was selected to effectively allow the model to simulate, perceptually, the IR of a real room. It is noticed that in [9], the use of filters above 8^{th} order is recommended for satisfactory results.



$$\begin{bmatrix}
-0.3 & 0.2 & 0.2 & 0.2 & -0.3 & 0.2 & 0.2 & -0.3 & -0.3 & 0.2 & -0.3 & 0.2 & -0.3 & -0.3 & -0.3 \\
-0.3 & -0.3 & 0.2 & 0.2 & 0.2 & -0.3 & 0.2 & 0.2 & -0.3 & -0.3 & 0.2 & -0.3 & 0.2 & -0.3 & -0.3 \\
-0.3 & -0.3 & -0.3 & 0.2 & 0.2 & 0.2 & -0.3 & 0.2 & 0.2 & -0.3 & -0.3 & 0.2 & -0.3 & 0.2 & -0.3 \\
-0.3 & -0.3 & -0.3 & -0.3 & 0.2 & 0.2 & 0.2 & -0.3 & 0.2 & 0.2 & -0.3 & -0.3 & 0.2 & -0.3 & 0.2 \\
0.2 & -0.3 & -0.3 & -0.3 & -0.3 & 0.2 & 0.2 & 0.2 & -0.3 & 0.2 & 0.2 & -0.3 & -0.3 & 0.2 & -0.3 \\
-0.3 & 0.2 & -0.3 & -0.3 & -0.3 & -0.3 & 0.2 & 0.2 & 0.2 & -0.3 & 0.2 & 0.2 & -0.3 & -0.3 & 0.2 \\
0.2 & -0.3 & 0.2 & -0.3 & -0.3 & -0.3 & -0.3 & 0.2 & 0.2 & 0.2 & -0.3 & 0.2 & 0.2 & -0.3 & -0.3 \\
-0.3 & 0.2 & -0.3 & 0.2 & -0.3 & -0.3 & -0.3 & -0.3 & 0.2 & 0.2 & 0.2 & -0.3 & 0.2 & 0.2 & -0.3 \\
-0.3 & -0.3 & 0.2 & -0.3 & 0.2 & -0.3 & -0.3 & -0.3 & -0.3 & 0.2 & 0.2 & 0.2 & -0.3 & 0.2 & 0.2 \\
0.2 & -0.3 & -0.3 & 0.2 & -0.3 & 0.2 & -0.3 & -0.3 & -0.3 & -0.3 & 0.2 & 0.2 & 0.2 & -0.3 & 0.2 \\
0.2 & 0.2 & -0.3 & -0.3 & 0.2 & -0.3 & 0.2 & -0.3 & -0.3 & -0.3 & -0.3 & 0.2 & 0.2 & 0.2 & -0.3 \\
-0.3 & 0.2 & 0.2 & -0.3 & -0.3 & 0.2 & -0.3 & 0.2 & -0.3 & -0.3 & -0.3 & -0.3 & 0.2 & 0.2 & 0.2 \\
0.2 & -0.3 & 0.2 & 0.2 & -0.3 & -0.3 & 0.2 & -0.3 & 0.2 & -0.3 & -0.3 & -0.3 & -0.3 & 0.2 & 0.2 \\
0.2 & 0.2 & -0.3 & 0.2 & 0.2 & -0.3 & -0.3 & 0.2 & -0.3 & 0.2 & -0.3 & -0.3 & -0.3 & -0.3 & 0.2 \\
0.2 & 0.2 & 0.2 & -0.3 & 0.2 & 0.2 & -0.3 & -0.3 & 0.2 & -0.3 & 0.2 & -0.3 & -0.3 & -0.3 & -0.3
\end{bmatrix}$$

Figura 3: Feedback matrix for a 15^{th} order FDN.

Due to the complexity of the IR of a real room, it is impossible to do a point-by-point approximation of the related time domain signal by a recursive artificial reverberation model. Therefore, a measure of perceptual similarity between the response of the model and the IR of a real room (only the late reverb) is necessary. Among the audio related measures presented in the literature, the selected one was the Energy Decay Curve (EDC), defined in equation 3:

$$EDC(t) = \int_{\tau=t}^{\infty} h^2(\tau)\, d\tau \tag{3}$$

where $h(t)$ is the IR [1,7]. The EDC measures the amount of energy that remains in the IR after time t. Hence, the comparison between the model's IR and the room's IR is performed using the EDC calculated for both. The details involved in this procedure will be described later in session 4.

3 GENETIC ALGORITHMS

A Genetic Algorithm (GA) is a stochastic metaheuristic from the area of evolutionary computation. A GA is a nature-inspired approach mainly used for optimization problems [4]. It is based on the Darwinian principle of the survival of the fittest.

Usually, an optimization problem has the following elements: (i) an objective function to be optimized (either a gain function to be maximized or a cost function to be minimized); (ii) a set of variables related to the parameters of the problem; (iii) a set of constraints that limit the possible range of the variables. Frequently, for real-world problems, the number of variables of the problem can be rather large (leading to a huge search space) and/or too constrained. Also, the objective function may be subjected do noise, dynamic variation, stochasticity or discontinuity. These factors precludes the use of mathematical methods of optimization, thus suggesting the use of metaheuristics such as GA.

When applying GAs for solving a real-world problem, the first issue to be modeled is how a possible solution for the problem is represented [3]. In a GA, the relevant variables of an optimization problem are encoded with a predefined alphabet (usually binary, integer or real numbers). GAs manipulate encoded variables instead of the variables themselves, that is why it is necessary to encode/decode such values (see section 4.3). Each variable is encoded with a limited space (for instance, a given number of binary digits), also known as a gene. A group of genes represent a chromosome. Here comes up the biological metaphor behind GAs. One or more chromosomes represent an individual, as in Biology, and a set of individuals represent a population. However, in the case of GA, an individual represents a possible (encoded) solution for an optimization problem. The encoded information is the genotype, and the corresponding set of (decoded) variables having meaning to the real-world is the phenotype. For the case of binary encoding of the chromosome, the induced search space is given by 2^{num_bits}.

Once defined the representation of a solution, it has to be evaluated. This is done by a fitness function that measures how good a solution is for the problem in hand. The fitness function encloses not only the objective function, but also, the decoding of an individual and other procedures (such as scale adjustment and constraint violation checking). For most problems, the evaluation of the fitness function is the most complex and computationally expensive part of the GA (this will be discussed later in section 4.4).

A AGA, by itself, is simple and intuitive, because it uses a human problem-solving principle of trial and successive refinement. A GA can be summarized in the following steps: (1) Randomly generate solutions for the problem; (2) Evaluate the quality of solutions; (3) If the best solution in hand is satisfactory, stop search; (4) Select probabilistically the best solutions according its quality; (5) Construct new solutions using parts of the solutions selected in the previous step; (6) Stochastically modify small parts of current solutions; (7) Return to step 2.

It is in step 4 where the selection of the fittest principle is evident, because the individuals with the highest quality will have the highest probability to be selected. In steps 5 and 6, another biological metaphor appears. New individuals will appear in the population as a recombination of parts of other individuals. Later, these new individuals can be slightly modified. These procedures are accomplished by means of the genetic operators of crossover and mutation, mimicking the corresponding biological phenomena. During each loop of the above pseudo-code, a complete generation of individuals (solutions) is generated and evaluated. Hopefully, the average quality of the population increases.

A GA has a number of running parameters that affect directly its performance, and have to be adjusted by the user (see section 4.5). Amongst them, the probabilities of application of crossover and mutation operators are responsible for modulating the intensity of local and global search, respectively. The selection method is also important, since it is directly responsible for maintaining diversity in the population throughout generations. Also, a fitness scaling procedure is usually implemented so as to avoid large dispersion in the fitness (quality) of evaluated individuals. This is done to avoid excessive differences in fitness among individuals, thus avoiding too high selective pressure towards the best individuals.

GAs do not guarantee to find the optimal solution for a given problem. Instead, a good quality solution can be found in a reasonable processing time, an acceptable trade-off for hard optimization problems.

4 IMPLEMENTATION

Preliminary tests with GAs were done using all parameters presented in Fig. 2 for the model chosen. These tests indicated that simplifications of the model are necessary, as well as the inclusion of knowledge about the problem in the GA. These modifications, presented below, were carried out and the optimization process became definitely easier.

4.1 Impulse Response of the Room

The IR used in this implementation is the one shown in Fig. 1. Since the file available was stereo, only one channel was used (the right one). The file was recorded using a sampling frequency of 44.1 KHz and has a total of 107520 samples, resulting in a signal of 2.44 seconds. As mentioned before, only the late reverb region was considered in this work. For this purpose, the first 1184 samples corresponding to the early reverb were removed. Samples after 88190 are considered background noise (after the impulse response) and were also removed. Therefore, a total of 87006 samples were effectively used in this work, which corresponds to around 81% of the original signal.

4.2 Absorbent Filters

A total of 15 absorbent filters take part of the FDN model (see Fig. 2). Filters $h_1(z)$ to $h_{15}(z)$ were, each one, designed as a composition of a first order low-pass filter (lpf_i) and a constant gain smaller than one. The poles were designated p_1 to p_{15} and the gain constants k_1 to k_{15}. The transfer function of the low-pass filters was obtained in [11], and is defined by equation 4. The cutoff frequency, in Hertz, can be computed by equation 5, in which f_s is the sampling frequency.

$$lpf_i(z) = \frac{1}{2} \cdot \left(1 + \frac{z^{-1} + p_i}{1 + p_i z^{-1}}\right) \tag{4}$$

$$fc_i(z) = \arctan\left(\frac{p_i + 1}{1 - p_i} \cdot \frac{f_s}{\pi}\right) \tag{5}$$

4.3 Chromosome Encoding

As mentioned before, some simplifications were introduced in the selected 15^{th} order FDN to facilitate the optimization process. Therefore, some of the parameters shown in Fig. 2 were actually not encoded due to the simplifying considerations done, as follows.

1. Gain constants b_1 to b_{15} and c_1 to c_{15}: preliminary tests indicated that these constants could be set to 1. Therefore, they were eliminated from the model.
2. Gain constant d: when only the late reverb region is considered, there is no reason to model the direct signal path. Hence, the gain constant d was set to zero.
3. Delay lengths m_1 to m_{15}: the genotypic value (gm_i) of each of these constants was encoded as an 11-bit unsigned integer. Equation 6 was used to obtain the phenotypic values (m_i) from their corresponding genotypic values, allowing a spam in the range $[1, 2048]$ delay samples.
$$m_i = gm_i + 1 \tag{6}$$
4. Poles p_1 to p_{15}: the genotypic value (gp_i) of each pole (see section 4.2) was encoded as a 12-bit unsigned integer. To obtain the phenotypic values (p_i) from their respective genotypic values, equation 7 was used. Therefore, the poles can vary roughly in the range $[-0.999512, 1]$.
$$p_i = \frac{gp_i - 2047}{2048} \tag{7}$$
5. Gain constants k_1 to k_{15}: these constants (see section 4.2) were not encoded. Notice that the decay characteristics of the model's IR depend on the parameters k_i, m_i and p_i. Preliminary tests have shown that the GAs alone would not be able to adequately balance these parameters, especially k_i and m_i. To solve this problem, all k_i were forced to depend on mi according to the following equation:
$$k_i = c^{m_i} \tag{8}$$
The parameter effectively encoded was the constant c. The genotypic value (gc) of this constant was encoded as a 12-bit unsigned integer. Equation 9 was used to obtain the phenotypic value (c) from its genotypic value.
$$c = gc \cdot \frac{8 \times 10^{-5}}{4095} + 0.9999 \tag{9}$$
Hence, constant c was defined in the range $[0.9999, 0.99998]$. This range, defined empirically, is relatively narrow. However, we observed that very small variations in c cause significant changes in the response of the model, thus justifying the number of bits employed.
6. Output filter $t(z)$: this filter was actually implemented as a constant, named t. Since the gain constants b_i and c_i were set to one, the purpose of t was to allow an output gain adjustment. Toward this end, the room's IR and the model's IR were normalized in the range $[-1, 1]$ before the EDC of each one was computed. Then, from the resulting normalization factors, the value of t was obtained by equation 10, where nf_M corresponds to the normalization factor of the model and nf_R to the normalization factor of the room.
$$t = \frac{nf_M}{nf_R} \tag{10}$$
Notice that, during the execution of the GA, the computed value of t for each individual was stored in its utility field. The utility field is a feature of GALOPPS (see documentation in [12]) that allows additional information associated with an individual to be stored separately from its chromosome. Therefore, after a run of the GA, such additional information can be extracted from the utility field of the elected individual. Since the value of t was computed and stored as described above, there was no reason to encode it.

Each individual, representing a possible solution to the problem, was composed by a single binary chromosome, with a number of genes. Taking into account the above mentioned simplifying considerations, the chromosome of our implementation was organized as a string of 357 bits, corresponding to the following variables: $gc|gm_1|\ldots|gm_{15}|gp_1|\ldots|gp_{15}$, where the symbol "|"is a concatenation operator. According to this encoding, the number of possible solutions is 2^{357}, a rather large search space even for an efficient optimization method like GA.

4.4 Fitness Function

For each individual, the fitness function was calculated by means of the 10 steps presented below. It should be noted that the EDC for the normalized IR of the real room (only the late reverb) is considered to have already been computed and stored before the beginning of this algorithm. It is also important to mention that a fitness weighting function, named w, was employed. It was empirically defined as a decreasing ramp from 1 to 10^{-6}. Its length is necessarily equal to the length of the EDCs.

S1: Extract the genotypic values of the parameters from the chromosome and obtain their corresponding phenotypic values.

S2: Designate as $dmin$ the minimum between the smallest delay line length of the model and 1184. This last number corresponds to the amount of removed samples (early reverb) from the room's IR.

S3: Use the phenotypic values obtained in step $S1$ to adjust the parameters of the model.

S4: Generate the IR of the model using an impulse as input. The length of this IR must be the sum of $dmin$ and the length of the room's late reverb.

S5: Advance the model's IR $dmin$ samples in time. In other words, shift the model's IR $dmin$ samples towards time zero.

S6: Truncate the obtained IR so that its length matches the length of the room's late reverb.

S7: Normalize the resulting IR in the range $[-1, 1]$.

S8: Compute the constant t and store its value in the utility field of the individual (see section 4.3).

S9: Compute the model's EDC using the normalized IR.

S10: Compute the weighted mean absolute error (E_{WMA}) between the model's EDC and the room's EDC, using equation 11:

$$E_{WMA} = \frac{\sum_{n=1}^{L} w(n) \cdot |EDC_M(n) - EDC_R(n)|}{\sum_{n=1}^{L} w(n)} \tag{11}$$

where EDC_M, EDC_R, L and w correspond, respectively, to the model's EDC, room's EDC, length of the room's late reverb and the weighting function mentioned in the beginning of this section.

After executing the steps above, it is possible to compute the fitness of the individual by using equation 12, where the constant K was empirically defined as 112:

$$fitness = \frac{K - E_{WMA}}{K} \tag{12}$$

It should be noted that the adjustment of the model's parameters was treated as a minimization problem by using equation 11 as the objective function. Nevertheless, the GAs were originally conceived to deal with maximization problems. Therefore, equation 12 is, in fact, responsible for converting the minimization criterion (equation 11) into a maximization criterion. It is important to recall that the fitness values calculated by equation 12 are already normalized in the range $[0, 1]$. Therefore, the perfect match between the model's EDC and the room's EDC would be represented by a fitness equal to 1.

The normalization of the model's IR and the room's IR (late reverb region) plays an important role in the evaluation of the fitness function. Preliminary tests have revealed that, without the normalization, the GA has a tendency to select individuals whose IRs have high amplitude peaks that are incoherent with the overall waveform. This unwanted behavior happens because these peaks are very sparse and may have little influence on the fitness of the individuals. To circumvent this problem, the proposed normalization was employed. With such procedure, the GA naturally assigns a low fitness value for any individual that exhibits the above mentioned behavior.

4.5 Control Parameters of the GA

The source code of the implementation was written in C language, and was based on the public-domain software GALOPPS (Genetic Algorithm Optimized for Portability and Parallelism) [12]. After a number of preliminary experiments, and based on the current literature [3], [4], the main parameters of the GA were set as follows: population size = 300 individuals; number of generations = 500; selection method = stochastic tournament with 9 individuals (3% of the population size); crossover = one-point crossover with probability of 80%; mutation: simple single-bit mutation with probability of 1%; fitness scaling: linear scaling with a ratio of best fitness to average fitness equal to 1.5; elitism: not used; initialization = based on an uniform distribution; encoding = binary; chromosome length = 357.

Since the GA is an stochastic method, the overall behavior of the algorithm for a given problem can only be characterized considering the average of several independent runs. Therefore, in this work, a total of 100 runs were performed using the same running parameters for the GA, but with different initial random seeds.

5 RESULTS

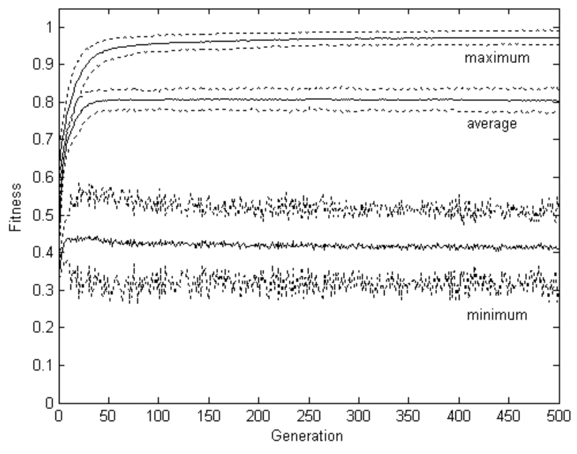
For the execution, we used two 3 GHz dual-core desktop computers with 2 GB of RAM and Microsoft Windows XP operating system. A total of 100 independent runs were done and the computational time per run was around 2.2 hours. Considering the computational resources used, the overall processing time for all the experiments was around 55 hours.

Fig. 4(a) shows the overall behavior of the GA throughout generations. In this figure, the solid lines correspond to the average maximum fitness per generation, average fitness per generation and minimum fitness per generation. Each point of these curves is the average of the 100 runs of the algorithm. The dashed lines presented in Fig. 4(a) delimit a confidence interval of 95% (2 standard deviations) around each solid line.

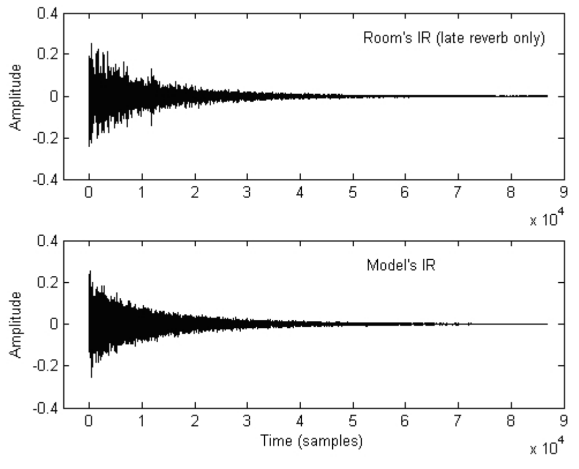
The average best fitness (and standard deviation) for the set of runs was equal to 0.98078 $\pm$ 0.00523. This value was computed by first selecting the best fitted individual of each run and then taking the average of their fitness values. The associated standard deviation represents roughly 0.5% of the computed average. Considering a confidence interval of 95% (2 standard deviations), the fitness of the best individual of a run can be described by 0.98078 $\pm$ 0.01046. In other words, the fitness of the best individual generated by the proposed GA in a given run is 95% likely to be included in that interval, thus suggesting that GA behaves consistently.

The individual chosen as the solution to the adjustment of the model (that is, the best-of-all-runs) achieved a fitness of 0.98904 and was found in generation 428 of its run. The model's IR corresponding to this individual is shown in Fig. 4(b) together with the late reverb of the room's IR. Their respective EDCs are displayed in Fig. 5(a). Notice that, for better visualization the room's EDC is shifted from its original position (although the model's EDC is in its original position). Such shift, performed in both time and amplitude, makes possible to compare the shape of the curves, and see that they are very similar.

The magnitude spectra of the room's IR and of the model's IR are shown in Fig. 5(b). This figure also includes a zoomed version of the magnitude spectrum of the model's IR. This zoom was done also to allow a better comparison between the spectra of the room's IR and of the model's IR.



(a)



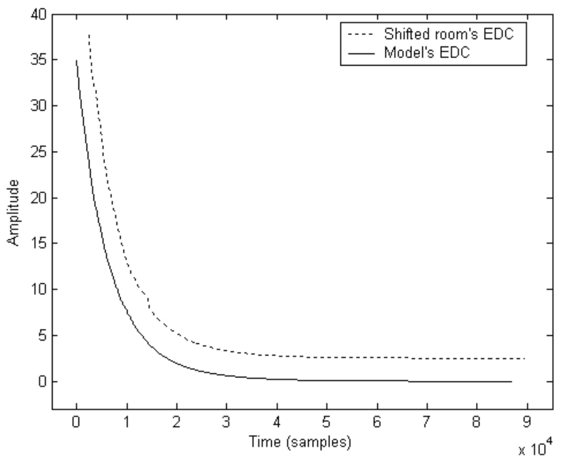
(b)

Figura 4: (a) Overall behavior of the GA. The solid lines correspond to the maximum fitness per generation (upper plot), average fitness per generation (middle plot) and minimum fitness per generation (bottom plot) averaged over the 100 runs. The dashed lines delimit a confidence interval of 95% (2 standard deviations) around each solid line. (b) Late reverb of the room's IR (upper plot) and model's IR for the best individual (bottom plot).

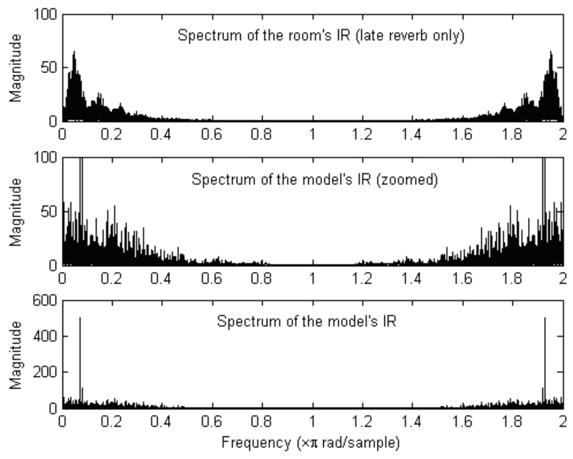
6 DISCUSSION AND CONCLUSIONS

Observing the results presented in the former section, it can be noticed that the model chosen, adjusted by the proposed GA methodology, was able to generate an EDC that closely followed the EDC of the late reverb region of a real room. In other words, the energy decay of the model's IR became very similar to the energy decay of the late reverb of the room's IR. Furthermore, the proposed GA proved to have a robust behavior. Notice, from Fig. 4(a), that the confidence intervals for the maximum and average fitness per generation delimit a very tight area around their respective mean values. This fact implies that the GA has a consistent behavior independently of the initial state of the pseudo random number generator. The same can be said about the fitness of the best individual that the GA generated in a certain run, which is 95% likely to be included in the interval 0.98078 $\pm$ 0.01046 (see section 5). Observe that the mean of this interval corresponds to a high value of fitness and the associated deviation is only $\pm$1% of the mean.

It can be noticed, however, that the spectra of the room's IR (only the late reverb) and of the model's IR have significant differences (see Fig. 5(b)). This fact indicates that the model, in fact, does not sounds like the real room. Qualitative audio tests have confirmed that the obtained model sounds metallic and somewhat unnatural. As a consequence, the resulting model cannot be used in practice right away. Nevertheless, it should be emphasized that the GA did very well what it was designed to do. That is, the EDC was employed as the measure of similarity and the obtained model was highly fit with respect to this criterion. These results suggest, therefore, that other measures of similarity besides the EDCs have to be devised, in order to produce a model that sounds like the real room. Considering the obtained results, a reasonable suggestion for future work is to combine both time and frequency domain information in the optimization criterion of the GA.



(a)



(b)

Figura 5: (a) EDC of the model in its original position (solid line) and EDC of the room's late reverb shifted 0.25×10^4 samples in time and 2.5 in amplitude (dashed line). (b) Magnitude spectrum of the late reverb of the room's IR (upper plot), magnitude spectrum of the model's IR (bottom plot) and a zoomed view of the spectrum of the model's IR (middle plot). The zoom was introduced to allow a better comparison between the spectrum of the room's IR and the spectrum of the model's IR.

Referências

[1] N. Toma, M. Topa, V. Popescu and E. Szopos. "Comparative performance analysis of artificial reverberation algorithms". In *Proc. of IEEE International Conference on Automation, Quality and Testing*, pp. 138–142, 2006.

[2] M. R. S. Schroeder and B. F. Logan. ""Colorless"Artificial Reverberation". *IRE Transactions on Audio*, vol. 9.

[3] D. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Reading, Massachusetts, 1989.

[4] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer, New York, 1998.

[5] G. Y. Liu Yongmei and M. Chunjing. "Design and Simulation of Artificial Reverberator Based on Simulink". In *Fifth IEEE International Symposium on Embedded Computing*, pp. 212–215, 2008.

[6] N. Collins. "Experiments with a New Customizable Interactive Evolution Framework". *Organised Sound*, vol. 7.

[7] J. Jot. "An analysis/synthesis approach to real-time artificial reverberation". In *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 2, pp. 221–224, 1992.

[8] M. R. Bai and G. Bai. "Optimal Design and Synthesis of Reverberators with a Fuzzy User Interface for Spatial Audio". *Journal of the Audio Engineering Society*, vol. 53.

[9] D. Rocchesso and J. Smith. "Circulant and elliptic feedback delay networks for artificial reverberation". *IEEE Trans. on Speech and Audio Processing*, vol. 5.

[10] D. Rocchesso. "Maximally diffusive yet efficient feedback delay networks for artificial reverberation". *IEEE Signal Processing Letters*, vol. 4.

[11] U. Zolzer. *DAFX - Digital Audio Effects*. John Wiley & Sons, New-York, 2002.

[12] E. Goodman. "GALOPPS 3.2.4 - the Genetic ALgorithm Optimized for Portability and Parallelism System". Technical report, Genetic Algorithms Research and Applications Group, Michigan State University, East Lansing, MI, 2002.