AUDIO TEXTURES

Lie Lu, Stan Li, Liu Wenyin, Hong-Jiang Zhang

Microsoft Research China
No.49 Zhichun Road, Beijing 100080, China
{i-lielu,szli,wyliu,hjzhang}@microsoft.com

Yi Mao

Institute of Artificial Intelligence
Zhejiang University, Hangzhou 310027, China
myb_79@sina.com

ABSTRACT

In this paper, we introduce a new audio medium, called *audio texture*, as a means of synthesizing long audio stream according to a given short example audio clip. The example clip is analyzed, and basic building patterns are extracted. Then an audio stream of arbitrary length is synthesized using a sequence of extracted building patterns. The patterns can be varied in the synthesis process to add variations to the generated sound. Audio textures are useful in applications such as background music, lullabies, game music, and screen saver sounds. A method is proposed for implementing audio textures. Preliminary results of audio textures are provided at our website for evaluation.

1. INTRODUCTION

The size of audio media is an important consideration in applications involving audio. The concerns include the storage needed for the audio data, and the time needed for download and transmission when the Internet is involved. How to make such media objects small in sizes will be critical to the success of the applications.

In many applications, there is a need for a simple sound of arbitrary length, such as lullabies, game music and background music in screen saver. Such sounds are relatively monotonic, simple in structure, and have repeated yet possibly variable sound patterns. A very long simple but not exactly repeating sound would require huge storage.

In this paper, we introduce the idea of a new audio media, called audio texture, as an efficient method for generating such sounds from example clips. Audio texture provides an efficient means of synthesizing continuous, perceptually meaningful, yet non-repetitive audio stream from an example audio clip. It is “perceptually meaningful” in the sense that the synthesized audio stream is perceptually similar to the given example clip. However, an audio texture is not just a simple repetition of the audio patterns contained in the input; variations of the original patterns are fused into it to give a more vivid stream. The audio stream can be of arbitrary length according to the need

The idea of audio texture is inspired by video textures [7], a new type of visual medium. The latter was proposed as a temporal extension of 2D image texture synthesis [1][2], and is researched in the areas of computer vision and graphics. It is natural to generalize the idea to audio data. Audio data as a signal sequence presents self-similarity as a video sequence does. The self-similarity of music and audio has been shown in [3] using a visualization method. So far, audio similarity is studied for audio or music retrieval only [4][5][9] .

We propose an approach for synthesizing audio textures. The key issue here is how to generate, from a short piece of example audio clip, an arbitrarily long audio sequence which bears similarity patterns to the original clip yet presents variations. A two-stage method is proposed for generating audio textures. In the analysis stage, the example clip is analyzed, and segmented into sub-clips by extracting its building patterns or equivalently finding pattern breakpoints. This step is based on the similarity measure between each two frames according to their Mel-frequency cepstral coefficients (MFCCs). In the synthesis stage, the sequence to play the sub-clips or building patterns is decided, and variable effects can be combined into the building patterns to avoid monotony of the synthesized audio stream. Audio texture is thus generated.

The rest of the paper is organized as follows. Section 2 presents an overview of the proposed method for generating audio textures. Section 3 describes algorithms for analyzing audio structure. Section 4 describes the algorithms for synthesis process. Section 5 presents settings for the experiments and provides preliminary results.

2. SYSTEM OVERVIEW

The proposed method for generating an audio texture can be divided into two stages: analysis and synthesis, as shown in Figure 1.

In the analysis stage, feature is extracted to represent the original audio data. The most important feature in our approach is Mel-Frequency Cepstral Coefficients (MFCCs). Then, the structure of the audio clip is analyzed, and the audio clip is segmented into several basic building patterns or sub-clips, where a pattern or sub-clip can be composed of a single frame or multiple frames. Meanwhile, the similarity and transition probability between each two sub-clips are calculated for further synthesis.

In the synthesis stage, we use sub-clip as synthesis unit. We would still keep using frame as synthesis unit, especially when no obvious building patterns are extracted from input audio example. Using different synthesis unit may be more efficient for different kind of audio. Frames can be considered as a special case of sub-clips, so we will only consider sub-clip in the

0-7803-7402-9/02/$17.00 ©2002 IEEE

following sections. A sub-clip sequence is first generated based on the transition probabilities, by deciding which sub-clip should be played after a given sub-clip. Different effects can be introduced by determining different sub-clip sequence or adding different effects to the sub-clips or building patterns. The variations include time scaling and pitch shifting, which can be implemented by synchronous overlap-add (SOLA) method. Once these are done, a perceptually natural audio stream, or an audio texture, is generated.

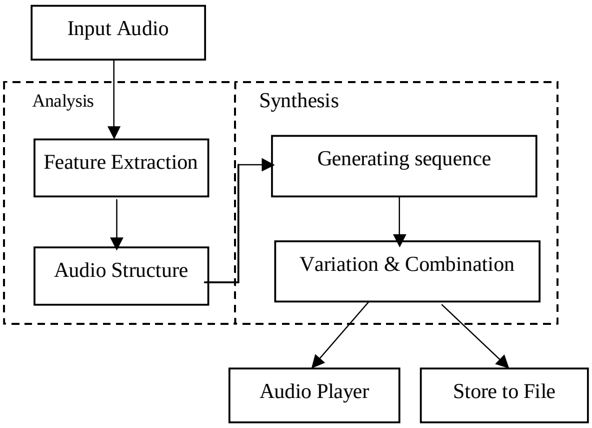

Figure 1. System overview diagram

3. ANALYSIS PROCESS

In this step, the structure of input audio clip is analyzed. It consists of two steps: similarity is first measured between each two frames, and then the audio clip is segmented into sub-clips as basic building blocks.

3.1 Similarity Measure

In order to generate a perceptually natural audio texture, it is necessary to consider the similarity between any two frames and the transition probability from one to another. It will be used to extract the audio structure and segment the original audio into several sub-clips. It is also the basis for synthesis if frame is used as synthesis unit.

Let $\boldsymbol{V}_i$ and $\boldsymbol{V}_j$ be the feature vectors of frames i and j in the MFCC feature space. The similarity measurement is simply based on vector autocorrelation and defined as,

$$S_{ij} = \frac{\boldsymbol{V}_i \bullet \boldsymbol{V}_j}{\|\boldsymbol{V}_i\| \cdot \|\boldsymbol{V}_j\|} \tag{1}$$

The above measure considers the isolated two frames only. In order to give a more comprehensive representation of the similarity, it will be better if their neighboring temporal frames are taken into considerations. Suppose that the previous m and next m frames are considered with weights $[w_{-m},...,w_m]$, the better similarity is developed as follows.

$$S'_{ij} = \sum_{k=-m}^{m} w_k S_{i+k,j+k} \tag{2}$$

This method captures the time dependence of the vectors. To yield a high similarity score, it requires that the two subsequences should be similar. In this way, we are actually matching two sub-clips instead of just two frames.

The transition probability from frame i to frame j depends on the similarity between frames $i+1$ and j. The more similar these two frames are, the higher the transition probability should be. In this principle, the transition probability is related to the similarity by the following exponential function.

$$P_{ij} = A\exp(\frac{S'_{i+1,j}-1}{\sigma}) \tag{3}$$

where A is the normalizing constant such that $\sum_j P_{ij} = 1$, and σ is the scaling parameter.

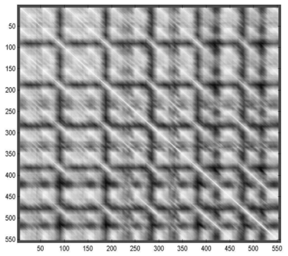

Figure 2. Similarity matrix of an example music clip

Figure 2 shows an example of similarity matrix, using 2D images representing S'_{ij} for all i,j, computed from a piece of music clip. The brightness of a pixel is proportional to the corresponding value. The brighter the pixel is, the larger the similarity is. The transition probability matrix also shows alike characteristics as similarity matrix but has one pixel offset.

3.2 Sub-Clip Extraction

Just as using di-phones in text-to-speech system, we also want to detect some possible building patterns and use them to synthesize texture instead of frames. Building patterns will be got by segmenting the input audio clip into sub-clips at some breakpoints. The segmentation is based on our novelty score at each time slot. Novelty score is used to measure the possibility of a new building pattern appears.

Consider a simple music clip having only two totally different notes, the similarity matrix will be something like following:

$$S = \begin{bmatrix} I & -I \\ -I & I \end{bmatrix} \tag{4}$$

where I is a unit matrix.

The diagonal unit matrix corresponds to the notes which have high self-similarity, while the off-diagonal matrix corresponds to the low cross-similarity between these two notes. When S is correlated by a kernel which looks like S but smaller than S at the diagonal direction, a maximum value will be obtained at the note change boundary. The correlation value at each diagonal point is used as the novelty score at that time. That is, the novelty score at the ith frame can be calculated as:

$$N(i) = \sum_{m=-w/2}^{w/2} \sum_{n=-w/2}^{w/2} K_{m,n} S_{i+m,i+n} \qquad (5)$$

where K is a kernel matrix with $2w+1$ dimension. (4) can be considered as a simple kernel. We could also use 2D window function (such as Hamming) replace the unit matrix in (4) to get a new kernel. This kind kernel can avoid edge effects because it tapers towards zero at the edges.

According to the novelty score, a simple scheme is developed to do sub-clip extraction: The local maxima of the novelty curve are selected as breakpoints. The sub-clip in each two breakpoints can be seen as building pattern.

Figure 3 show an example of sub-clip or building pattern extraction for a music clip. (a) shows the original music data and (b) shows the corresponding novelty score. The local peak is selected as building pattern boundary. From (b), it could be seen that the local peaks of the novelty curve is basically corresponds to the onsets of the music piece. That is, one note or several notes is extracted as one building pattern.

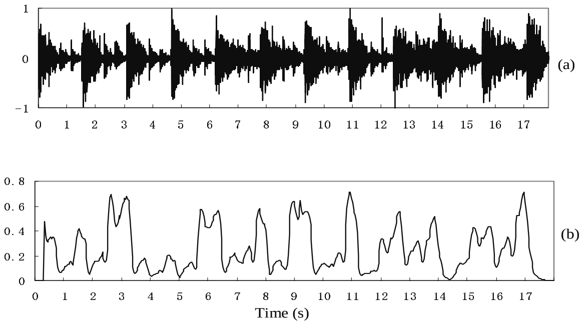

Figure 3 An example of sub-clip and building pattern extraction (a) Digital audio data (b) similarity score curve

Some modification is needed when Equation (2) is used to calculate the similarity between each two sub-clips, because that definition assumes that sub-clips are of equal length. But the segmented sub-clips are usually not of equal length, using this method. In principle, time-warping and dynamic programming method should be used to compute the similarity between each two sub-clips. However, we proposed a simplified method as follows:

Suppose sub-clip i contains M frames and begin from the frame i; sub-clip j contains N frames and begin from the frame j; and $M < N$. The similarity between these two sub-clips can be represented by:

$$S'_{ij} = \sum_{k=1}^{M} w_k S_{i+k,\, j+\left[k\frac{N}{M}\right]} \qquad (6)$$

It will be more reasonable to consider the neighboring sub-clips when the similarity between two sub-clips is measured:

$$S''_{ij} = \sum_{k=-m'}^{m'} w'_k S'_{i+k,j+k} \qquad (7)$$

The transition probability from ith sub-clip to jth sub-clip is determined by $S''_{i+1,j}$, and can be calculated by the similar equation as (3).

4. SYNTHESIS PROCESS

After the transition probability between every two sub-clips has been found, the audio texture can be generated sub-clip by sub-clip. The issues here are: (1) to determine the order in which the sub-clips should be combined and played, and (2) to add effects into the building patterns.

4.1 Determination of Sequence Order

The sub-clip j following sub-clip i may be selected simply by maximum probability P_{ij}. In real applications, this scheme sometimes causes repetition of a small part of the original audio stream, especially at the end of the audio. In order to solve this problem, we select sub-clip j with certain conditions, as defined by the follow equation:

$$j = \max_{j \notin [i-a,i]} \{P_{ij}\} \qquad (8)$$

This means that the next sub-clip of sub-clip i is searched in all sub-clips but those in search window [i-a, i], where a is the size of the window.

To introduce more stochastic characteristics in the generated sub-clip sequence, we could also select any one of sub-clips in the following set as subsequence:

$$j \in \{ j \mid P_{ij} > p_0 \} \qquad (9)$$

where p_0 is a threshold and used to control the number of candidate sub-clips. Larger values of p_0 emphasize the very best transitions while smaller values of p_0 allow for greater variety at the cost of poorer transitions.

In fact, in the sequence order determination, we should also consider the smoothness of amplitude and pitch. It will be very helpful for perceptual ease of the final generated texture.

4.2 Adding Effects

Variations can be introduced to the sub-clips. In our implementation, potential variations are time scaling, pitching shifting and amplitude setting. Different variations can be done by setting different values for the controlling parameters. However, a parameter value for pitch-shifting should be applied to a group of consecutive frames (or a sub-clip) to avoid abrupt changes in pitch. A smoothing is performed on the transitional frames between two groups in order to ensure the pitch transits smoothly.

An interpolation method or TD-PSOLA (Time Domain – Pitch Synchronous OverLap-Add) method [6][8] is used for implementing time scaling and pitch shifting. For interpolation, time scale and pitch scale will be changed simultaneously. If one wants to change time scale and pitch scale independently, TD-PSOLA is a good choice.

4.3 Synthesis by Sequencing and Combining

Basically, an audio texture is generated as a sequence of sub-clips possibly varied. However, there are many different ways of sequencing and combining the sub-clips. For example, for the sound of horse neighing, we can generate a sequence of neighing of a single horse using head to tail sequencing; we also can generate an effect in which several horses are neighing synchronously or asynchronously, by time-overlapping of some textures. Moreover, we can generate an effect of horses running towards and then away from the listener, by using certain variation in pitch and amplitude. In sequencing and combining, TD-SOLA is used again to smooth the break between two concatenated sub-clips.

5. EXPERIMENTS

A set of audio textures are generated using the audio texture algorithms presented in this paper. The original audio clips are all 2-15 seconds long, sampled at the rate of 8KHz or 32KHz, mono channel, and encoded by 16bit per sample. Experiments are performed on sounds such as horse neighing, rooster crowing, thunder, explosion, raining, stream, ripple and simple music. Some examples are shown below:

Horse neighing. The input audio is 2 seconds long. It contains just one neigh of a single horse. Eight sub-clips are extracted. These sub-clips correspond to the start, the end and several vibrations in the middle of a neigh. The length and pitch variations for each sub-clip are set randomly. The length of sub-clip sequence is chosen from a certain range. Different neighing of horse can be generated by adjusting these parameters. Then two textures are synthesized. The first one is done by combining the different neighs as a temporal sequence, generating a sound that a horse is neighing continuously. Another one is done by combining different neighs with some time-overlapping, give a sound in which a group of horses are neighing synchronously and asynchronously.

Stream. This example is used to show how to generate textures by using individual frame when no obvious building pattern is found. The input audio is about 11 seconds long. It is divided into 25ms frame with 12.5ms overlapping. The texture is generated by sequencing these frames instead of sub-clips. The length of texture is set randomly. Variations on time-scaling and pitch-shifting are set for each one-second texture. To prevent the pitch from changing too dramatically, smoothing is performed. The generated texture is stream of infinite length, with some variations in stream speed and amplitude.

***Simple music*.** This example shows how this algorithm works on simple music, since music is always more complex than other audio types. This clip just has some simple rhythms, not as complex as the traditional songs. It is about 12 seconds long and is segmented into several building patterns. No effects are added to any building pattern. Final texture is synthesized by sequencing sub-clips. Results show the algorithm works well.

Some preliminary experiment results are presented on the website: http://research.microsoft.com/~szli/AudioTextures. The interested reader may want to compare the original sound and the synthesized audio texture.

6. CONCLUSION

In this paper, we have introduced a new audio media, called audio textures. An audio texture is an audio sequence of arbitrary length generated from a short clip of audio example. It consists of consecutively connected patterns that are perceptually similar to those contained in the example clip but present variations. A method has been proposed for the extraction of the basic patterns from the original clip, the making of variations of the basic patterns and the connection of the variable patterns into a long sequence. Algorithm has been presented to implement the idea.

There are many potential applications for audio textures such as lullabies, game music, background sounds and other effects. Another potential application is that it is a good choice for audio compression. We also hope the new concept could inspire you on your research work in audio field.

The Audio texture technique can be improved in several aspects in the future work. In analysis step, we just use a correlation to measure the similarity between each two frames or sub-clips. It will be more useful if we could find a perceptual similarity measurement. In the synthesis step, we decide sub-clip sequence based on local similarity, how to control the global perception of generated texture is still a difficult task. Other features, such as amplitude and pitch, will be helpful for audio texture generation. In experiments, it would be better if more effective evaluation, such as perception testing, could be used to evaluate our algorithm. We would also extend our work to more traditional music. Thus, more powerful signal processing methods are needed.

7. REFERENCES

[1] J.S. de Bonet. "Multi-resolution sampling procedure for analysis and synthesis of texture images". *SIGGRAPH'97*. pp. 361-368, 1997.

[2] A.A. Efros, T.K Leung. "Texture Synthesis by Non-parametric Sampling". In *Proc. IEEE International Conference on Computer Vision*, 1999.

[3] J. Foote. "Visualizing Music and Audio using Self-Similarity". In *Proc. ACM Multimedia '99*, pp. 77-80, Orlando, Florida, November 1999.

[4] J. Foote. "Content-based retrieval of music and audio". In C. C. J. Kuo et al., editors, Multimedia Storage and Archiving Systems II, *Proc. SPIE*, volume 3229, pages 138-147, 1997.

[5] S.Z. Li. "Content-based Classification and Retrieval of Audio Using the Nearest Feature Line Method". *IEEE Transactions on Speech and Audio Processing*, 8(5):619-625. September, 2000.

[6] E. Moulines, F. Charpentier. "Pitch-Synchronous Waveform Processing Techniques for Text-To-Speech Synthesis using Diphones", *Speech Comm.*, Vol.9, pp453-467, 1990.

[7] Schodl, R. Szeliski, D.H. Salesin, I. Essa. "Video Textures". *Computer Graphics Proceedings, Annual Conference Series*, pages 33-42, Proc. SIGGRAPH 2000, July 2000. ACM SIGGRAPH

[8] H. Valbret, E. Moulines, J. P. Tubach. "Voice Transformation using PSLOA Technique". In *Proc. ICASSP-92*. 1992.

[9] E. Wold, T. Blum, and J. Wheaton. "Content-based Classification, Search and Retrieval of Audio". *IEEE Multimedia*, 3(3), pp.27-36, 1996.