Cluster-Weighted Sampling for Synthesis and Cross-Synthesis of Violin Family Instruments

Bernd Schoner, Chuck Cooper*, Neil Gershenfeld
Physics and Media Group
MIT Media Laboratory
20 Ames Street
Cambridge, MA 02139
{schoner,cmc,gersh}@media.mit.edu

Abstract

A digital model of a violin/cello is presented that computes violin sounds in real time given gesture input from a violinist. The model is derived from recorded data using cluster-weighted sampling, a probabilistic inference technique based on kernel density estimation and wavetable synthesis. Novel interface technology is presented that records gesture input of acoustic instruments and also serves as a stand-alone controller for digital sound engines.

1 Introduction

In this paper, we show how to synthesize the sound of violin-family instruments from a player's gestural input. The input-output model describes the mapping between physical input time series and a perceptual parameterization of the output time series. It is trained on a recorded data set consisting of gesture input data of the violinist, such as bow and left hand finger position, along with audio output data. The audio signal is reconstructed from the estimated parameterization and from audio samples stored in memory.

The probabilistic network architecture cluster-weighted modeling (CWM) was developed earlier to model, characterize, and predict input-output time series. For the purpose of synthesizing musical signals, we have extended the framework to cluster-weighted sampling (CWS), which generates sound output directly from the sensor input. CWS uses pre-sampled sounds as a basis for function approximation. In the training step, brief sound segments that best represent a certain playing situation are selected. In the synthesis step, the segment that is most likely given the current input data is used as sound output.

*Present address: Plangent Systems Corporation, 25 Fairfield St., Newton, MA 02460.

Our approach to musical synthesis lies conceptually in between physical modeling and global wavetable synthesis. We do not model the instrument in terms of first principle governing equations, but it is our goal to build a model that, from the point of view of a listener or player, appears to obey the same physical laws as the acoustic instrument. On the other hand, we choose to represent the audio data in terms of sequences of samples.

2 Previous and related work

[SCDG99] showed how to synthesize violin sounds in a data-driven machine learning approach. Given a sinusoidal sound representation, the inference technique CWM was used to describe the mapping from real-time player input to a parameterization of relevant harmonic components of violin sound. The frequency domain parameters were translated into real-time sound through additive synthesis.

CWM is a mixture density estimator for function approximation. The framework has been discussed in detail in [SCDG99, GSM99]. Here we extend the general framework to create cluster-weighted sampling (CWS), a sampling (wavetable) synthesis approach that uses the probabilistic structure of CWM to allocate and parameterize samples.

Wavetable synthesis has become the predominant synthesis technique for commercial digital instruments [Mas98]. [Roa95] and [Mas98] provide extensive reviews of the fundamental algorithms as well as the ad hoc techniques behind wavetable synthesis. The technique works well for instruments with low dimensional control space and without aftertouch, such as the piano. However, the input space for instruments like the violin tends to be too complex to cover all the possible outputs with sampled data. In this paper, we demonstrate how to parameterize the audio efficiently and reconstruct sound from samples and estimated parameters.

We also show how sample sequences can be selected as part of the probabilistic framework.

[PR99] introduced SINOLA, a system that mixes sinusoidal synthesis and wavetable synthesis. This paper presents a similar approach in that we try to capture the perceptually important attack of a bow stroke with appropriate sample sequences while we synthesize the sustained part of a note with the previously presented sinusoidal methodology.

3 Hardware and data collection

The violin sensor hardware used in earlier publications has been replaced and significantly extended [PG97, SCDG99]. A new circuit board was designed that senses

- the bow position relative to the strings. Oscillators (25kHz and 35kHz) mounted on the frog and tip of the bow coupled through a chain of resistors running along the bow into a receiving antenna underneath the strings [PG97]. The position of the bow is proportional to the difference divided by the sum of the two signals.
- the bow position relative to the bridge. A wire running along the bow hair couples a third signal (49kHz) into an antenna mounted on top of the bridge. The received signal strength is proportional to the bow position relative to the bridge.
- the finger pressure between forefinger and bow or alternatively between bow hair and bow tip. The signal is transmitted through a frequency- modulated carrier (68kHz).
- the finger position on each of the four strings. A DC voltage is applied to a resistive ($\approx$ 1 Ohm) stainless steel strip covering the fingerboard. When a string is pressed onto the fingerboard it picks up the voltage at the specific position.
- the string audio signals. Small magnets placed underneath the strings induce a voltage in the vibrating string which is picked up at the ends of the strings.

For data collection, the analog sensor data is recorded along with audio data on a PC data acquisition board (13 channels at 22050 kHz, fig. 1).

For synthesis, a cello interface was developed that is strapped on to the player's body (fig. 1). The *Marching Cello* has the same sensor components as the instrumented acoustic cello. The player bows how or her leg close to the receiving antennas mounted at the end of the fingerboard. The fingerboard consists of four separate strips of stainless steel mounted on a support of transparent plastic. A layer of conductive foil is mounted on top, slightly separated by adhesive material on the edges. When the player pushes on the fingerboard the top layer makes contact with the steel foil. The sensor data is conditioned and digitized in a belly pack and RF transmitted to a receiving unit. The

Figure 1: *Left*: Acoustic violin with mounted sensors. *Right*: Marching Cello

RF receiver in turn connects to a lightweight ethernet board, which sends the data over the local network to a PC running the model.

4 Data-analysis

We extract a spectral representation of a fixed number of strictly harmonic components ($f_s = 22050Hz$, 86 frames/s, window size dependent on the register)[SCDG99] from the audio training data using a short-term Fourier transform. From this we infer the instantaneous pitch and the instantaneous amplitude envelope of the signal.

Sign changes of the bow velocity and the direction of the Helmholtz motion let us find the bow changes as well as the bowing direction. We assign negative (up-bow) and positive (down-bow) signs to harmonic amplitudes and to the instantaneous signal amplitude (fig. 4). This little trick forces the signal to pass through zero when a bow change occurs during synthesis.

The spectral components function as a indicator of timbral properties of a particular piece of sound. Instantaneous amplitude and energy along with the audio data are used for training of the CWS model and for CWS resynthesis.

5 Cluster-weighted sampling

CWM approximates arbitrary functions by allocation of simple output models in a globally nonlinear space. Local models are weighted by Gaussian basis terms (clusters), each of which is represented by an unconditioned probability $p(c_k)$, an input probability distri-

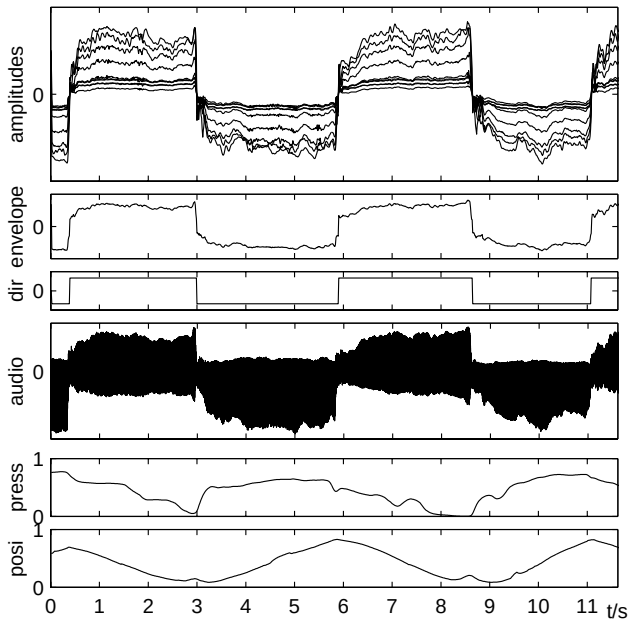

Figure 2: Analyzed cello data. *From the bottom*: bow position relative to the strings, bow pressure, audio signal, extracted bowing direction, envelope, harmonic amplitudes.

bution $p(\mathbf{x}|c_k) = \mathcal{N}(\mathbf{m}_k, \mathbf{P}_{x,k})$, where $\mathbf{x}$ is the feature vector, and $\mathcal{N}$ denotes a normal distribution with mean $\mathbf{m}$ and covariance matrix $\mathbf{P}$, a conditioned output distribution $p(\mathbf{y}|\mathbf{x}, c_k) = \mathcal{N}(\mathbf{f}_k(\mathbf{x}), \mathbf{P}_{y,k})$ and an output model $\mathbf{f}_k$. The deterministic predictor is extracted from the probabilistic description,

$$\hat{\mathbf{y}}(\mathbf{x}) = \langle \mathbf{y}|\mathbf{x} \rangle = \frac{\sum_{k=1}^{K} \mathbf{f}_k(\mathbf{x})\ p(\mathbf{x})|c_k)\ p(c_k)}{\sum_{k=1}^{K} p(\mathbf{x})|c_k)\ p(c_k)} \quad . \quad (1)$$

The model is trained with the Expectation-Maximization algorithm [DLR77, GSM99].

CWS uses multiple output models covering pitch prediction, amplitude prediction and sample selection (fig. 5). Using a simple locally linear estimator, the first output model estimates pitch, given some feature vector input $\mathbf{x}_\mathrm{p}$, which includes the left hand finger position on the fingerboard,

$$\mathrm{p}(t) = \frac{\sum_{k=1}^{K} \mathbf{a}_{k,\mathrm{p}}^T \mathbf{x}_\mathrm{p}(t)\ p(\mathbf{x}(t)|c_k)\ p(c_k)}{\sum_{k=1}^{K} p(\mathbf{x}(t)|c_k)\ p(c_k)} \quad . \quad (2)$$

The vector of coefficients $\mathbf{a}_{k,\mathrm{p}}$ is of dimension $D_{x_\mathrm{p}}+1$, with $x_0 = 1$ (the constant term of the linear estimator).

As in (2), the second output model predicts the instantaneous scaling factor $v(t)$ (volume) given the feature vector $\mathbf{x}_\mathrm{v}$, and the output model $\mathbf{f}_k(\mathbf{x}_\mathrm{v}) = \mathbf{a}_{k,\mathrm{v}}^T \mathbf{x}_\mathrm{v}$.

The third expert is a pointer into sample space. During training, the sequence is picked to represent a cluster that has the highest posterior probability with respect to this cluster. We assume a sequence to be perceptually represented by its instantaneous spectral characteristics, packaged in the vector $\mathbf{y}$. $\mathbf{y}$ can be interpreted as a pointer into timbre space.

$$\mathcal{S}_k = \mathrm{argmax}_{\mathcal{S}}\ p(\mathbf{y}_\mathcal{S}, \mathbf{x}_\mathcal{S} \mid c_k) \quad (3)$$

to represent cluster c_k.

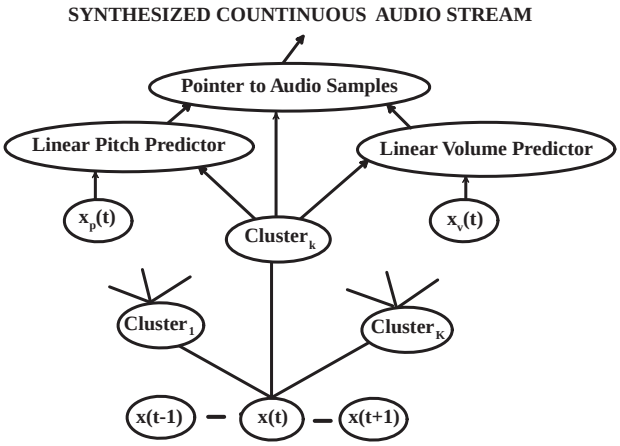

Figure 3: Network architecture: clusters with three output models.

For synthesis, this idea is inverted. The cluster c_k which most likely generated the input vector $\mathbf{x}_s$ takes over and its sequence of samples is used for replay.

$$\mathcal{S} = \mathcal{S}_{c_k}|c_k = \mathrm{argmax}_{c_k}\ p(\mathbf{x} \mid c_k)\ p(c_k) \quad (4)$$

The selected samples are re-sampled with respect to the predicted target pitch using

$$\begin{aligned} \hat{s}(t) &= \sum_{n=-N}^{n=N} s(n \cdot T_s)\ h_s(t - n \cdot T_s) \quad (5) \\ h_s &= \min\{F_s/F_s'\} \cdot \mathrm{sinc}(\min\{F_s, F_s'\} \cdot t) \\ \mathrm{sinc}(x) &= \frac{\sin(x)}{x} \quad , \end{aligned}$$

where F_s is the sampling frequency of the sequence, F_s' is the target sampling frequency, and N is the number of filter coefficients. If the signal is down-sampled, i.e. pitch is increased, the signal needs to be low-pass filtered with respect to the new Nyquist frequency. This is taken care of by the term $\min\{F_s/F_s'\}$ in equ. (5).

Likewise we rescale the sequences with respect to the predicted target volume, which is as simple as

$$s[n] = s_r[n]\frac{v_t[n]}{v_r[n]} \quad (6)$$

where s is the final output, s_r is the prerecorded sample, v_t is the instantaneous target volume and v_r is the instantaneous prerecorded volume.

An important detail is the sequencing of pieces of audio, when cluster sequences have to be looped or transitions occur from one cluster/sequence to another. We

choose to match samples by minimizing the least square error between the old and the new samples. Given $s_1[n]$ and $s_2[n]$ we find the correct offset T in between these two sequences through

$$T = \text{argmax}_T \sum_{n=1}^{N} s_1[n] \cdot s_2[n+T] \tag{7}$$

where N is the length of the correlation window, which should exceed the period of the sequence. Additionally we fade out s_1 and fade in s_2 using a Hamming window overlap-add as in

$$\begin{aligned} s[N+n] &= s_1[N_1 - N_f + n] \cdot (1 - A[n]) + s_2[n] \cdot A[n] \\ A[n] &= \sin^2\left(\frac{\pi n}{2N_f}\right) \end{aligned} \tag{8}$$

N is the starting point of the cross-fade, N_1 is the length of the ending sequence and N_f is the length of the fading ([Mas98]).

6 Experimental results

In an off-line approach we build a model from a training data set (fig. 4) and synthesize violin sound from out-off-sample input sensor data. Audio examples are available from [www].

In addition to the pure sampling approach, we mix sinusoidal and wavetable synthesis. The strength of the sinusoidal representation is its flexibility and its robustness with respect to new player input. The disadvantage of this approach is that non-harmonic sound components are not represented. Hence the sinusoidal representation and prediction handles sustained sound very well, but fails in dealing with transitional parts of the sound. The wavetable approach reproduces the full sound, including the noise components, but this comes at the cost of less flexibility and more undesired artifacts.

We preserve the noisy and characteristic violin attack, without compromising the flexible sinusoidal modeling of the sustain part of the note, by replaying stored samples immediately after the bow change, having the sinusoidal method stand in for the sustained part of a note. Whenever the synthesis program detects a bow change, it starts a recorded note onset sequence which is pitch adjusted and scaled with respect to the estimated parameters. After about half a second, the sample sequence fades out and the sinusoidal model takes over.

A video clip of real-time synthesis with input data generated by the Marching Cello is available from [www]. We calibrate the raw sensor data through a further layer of simple CWM units with respect to experimental calibration data. Therefore the interfaces are interchangeable; that is, the Marching Cello can be used to generate violin sounds and the violin interface can be used to generate cello sounds. Both interfaces could be considered universal musical controllers rather than devices that emulate a single specific instrument.

7 Conclusions and Future Work

We showed how to extend the notion of global sampling into a flexible sampling engine that allows for continuous control of the sound material. The method has been seamlessly integrated with the probabilistic framework of CWM, which greatly simplifies the allocation and selection of samples.

It has also been shown how the benefits and weaknesses of sampling and sinusoidal synthesis can be traded off with each other and can be combined to produce more flexible and complete models.

Future work will stress the notion of cross-synthesis across instrument families. The probabilistic nature of the model allows for easy and clean integration of different kinds of data. Hence the violin/cello interface can be used to drive very different instrument models with similar aftertouch control, for example, a trombone.

8 Acknowledgments

The authors would like to thank Diana Young and Mary Farbood for helping with the data collection and for performing the model. This work was made possible by the Media Lab's Things That Think consortium.

References

[DLR77] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum Likelihood From Incomplete Data via the EM Algorithm. *J. R. Statist. Soc. B*, 39:1–38, 1977.

[GSM99] N. Gershenfeld, B. Schoner, and E. Metois. Cluster-weighted modeling for time series analysis. *Nature*, 379:329–332, 1999.

[Mas98] Dana C. Massie. Wavetable sampling synthesis. In Mark Kahrs and Karlheinz Brandenburg, editors, *Applications of Digital Signal Processing to Audio and Acoustics*, pages 311–341. Kluwer Academic Publishers, 1998.

[PG97] Joseph A. Paradiso and Neil Gershenfeld. Musical applications of electric field sensing. *Computer Music Journal*, 21(2):69–89, 1997.

[PR99] Geoffrey Peeters and Xavier Rodet. SINOLA: A New Analysis/Synthesis Method using Spectrum Peak Shape Distortion, Phase and Reassigned Spectrum. In *Proc. ICMCs*, Beijing, 1999.

[Roa95] Curtis Roads. *The computer music tutorial.* MIT Press, 1995.

[SCDG99] B. Schoner, C. Cooper, C. Douglas, and N. Gershenfeld. Data-driven modeling of acoustical instruments. *Journal for New Music Research*, 28(2):417–466, 1999.

[www] www.media.mit.edu/physics/publications/papers/ICMC00/.