Data driven identification and computer animation of bowed string model

Stefania Serafin
Julius O. Smith III
Harvey Thornburg
CCRMA
Department of Music
Stanford University, Stanford, CA
email :serafin@ccrma.stanford.edu

Frederic Mazzella
Arnaud Tellier
Guillaume Thonier
Stanford-NASA National Biocomputation Center,
Computer Science Department
Stanford University, Stanford, CA

Abstract

The goal of this paper is to provide a technique to estimate the input parameters for a waveguide physical model. This work is motivated by the fact that the input parameters strongly influence the quality of the synthetic sound produced by the model. Though we examine the case of a bowed string instrument, this technique can be used to estimate parameters of other self-sustained oscillators. We furthermore describe a computer animation of a violin player whose motion is driven by the estimated parameters.

1 Introduction

One of the characteristics of physical models is that the variation of input parameters over time influences the quality of the synthesis almost as much as the precision of the model itself.

In this paper we propose to estimate the main parameters that a violinist can control with his/her right hand, i.e. the bow force, bow velocity and bow position of a bowed string physical model.

Previous similar work (1) used a two layer neural network in order to reconstruct the input parameters from the time domain waveform. However, different time-domain representations can lead to a perceptually equivalent result. For this reason we prefer to perform the estimation using frequency-domain features, as described below.

As shown in figure 1, recovering the input parameters from the acoustic waveform corresponds to *inverting* the model. Ideally, it would be nice to invert the model analytically, but in the case of the bowed string this is not possible, because it is a nonlinear dynamic system with no known analytical form. Furthermore, the inverse is not necessarily unique, since many combinations of bowing parameters produce nominal Helmholtz motion. For this reason we use a pattern-recognition based approach as described in the following section.

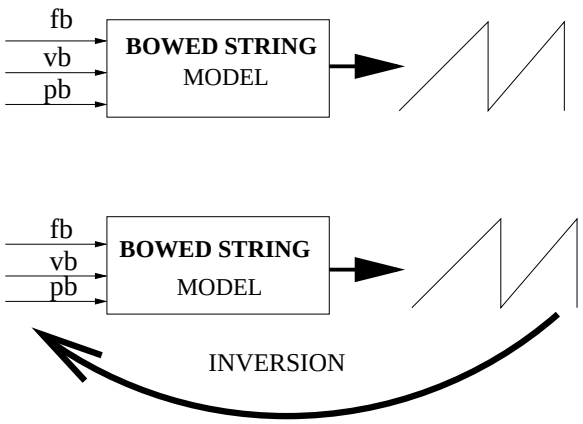

Figure 1: Top: main input parameters of a bowed string physical model whose combination produce the Helmholtz motion. Bottom: inversion of the model, i.e. reconstruction of the input parameters from the acoustic waveform.

2 Pattern recognition based approach

We discretize the input parameter space into 10 values each, for bow force, bow velocity and bow position. Bow force varies between 0.01 and 5 N, bow velocity between 0.02 to 2 m/s while bow position varies between 0.01 and 0.4, where 0 corresponds to the bridge and 1 corresponds to the nut. These ranges contain the playability region of the model as described in (4). This gives 1000 parameter classes of 800 samples each at a fundamental frequency of 147 (cello D string) and at a sample rate of 44100 Hz. Our goal consists of identifying the parameter class from the acoustic waveforms.

3 Feature selection

For each of the time-domain waveforms, we perform linear prediction in order to retain only the important features of the signal's spectral envelope. We retain a vector of 13 LPC coefficients per waveform. Next we transform the LPC feature space using principal component analysis (PCA) (6). PCA is a linear transformation which returns a set of feature vectors where each component is marginally uncorrelated with other components across all feature data irrespective of class. Within each class, however, the individual feature vectors may be correlated. However the correlation is usually much less than in the original feature space, allowing further simplification of the statistical model.

3.1 Estimating the bow position

Dealing with 1000 classes would decrease the performance of our classifier, due to insufficient training data. Therefore, to reduce the number of classes, we decided to estimate the bow position independently, using spectral domain techniques as proposed in (5). In this paper, the plucking point is determined by minimizing the absolute value of the error between the ideal string magnitude spectrum and the sampled-data spectrum. This is a regression rather than classification based approach. After bow position is identified 100 classes remain for the different values of bow force and velocity.

4 Classification algorithm

4.1 Description of the algorithm

Let $p(x|C_k)$ denote the conditional distribution of the feature vector x given one of the hundred classes C_k. To better respond to the empirical variation in bowed string waveforms, we model $p(x|C_k)$ as a mixture of n Gaussians:

$$p(x|C_k) = \sum_{j=1}^{n} \rho_j \cdot \frac{1}{2\pi \left|\Sigma_{kj}\right|^{1/2}} e^{\frac{1}{2}(x-\mu_{kj})^T \Sigma_{kj}^{-1}(x-\mu_{kj})}$$

where $\sum_{j=1}^{n} \rho_j = 1$

The *parameters* $\{\rho_j\}_{j=1}^n$, $\{\mu_{kj}\}_{j=1}^n$, and $\{\Sigma_{kj}\}_{j=1}^n$ are unknown. Additionally, we constrain $\Sigma_{kj} = \sigma_{kj} \cdot \mathbf{I}_{2\times 2}$.

To learn the parameters of the density $p(x|C_k)$, we use the Expectation Maximization (EM) algorithm (2). Additionally, we must know the *prior* distribution of each class, $P(C_k)$ which we assume to be uniform. Hence, the joint probability of the feature vector and class is completely specified: $P(x, C_k) = p(x|C_k)P(C_k)$. We develop our classifier to minimize expected *loss*, where the expectation is with respect to $P(x, C_k)$. We incur loss $L(k, j)$ where the correct class is C_j and our classifier returns C_k. According to (6), the optimal classifier is given by:

$$\begin{aligned} k_* &= \operatorname*{argmin}_k \sum_{j=1}^{M} L(k,j) P(x, C_j) \\ \hat{C}(y) &= C_{k_*} \end{aligned}$$

Choices of Loss Matrix

- **Binary loss** With *binary loss*, $L_0(k, j) = \mathbf{1}_{\{\mathbf{j=k}\}}$, the average loss gives the percentage misclassified.

  Binary loss is not realistic for our parameter estimation task because gross errors in estimating input parameters (from faraway classes) are penalized the same as small errors (from neighboring classes). Furthermore, the sensitivity of binary loss increases with the resolution of the class subdivision. To remedy this we describe two alternatives: Euclidean and playability loss.

- **Euclidean loss** The *Euclidian loss* is given as follows. Let $F_{\min}, F_{\max}, V_{\min}, V_{\max}$ denote the ranges as given in (2). Then, define

  $$L_2(k, j) = \left(\frac{F_k - F_j}{F_{\max} - F_{\min}}\right)^2 + \left(\frac{V_k - V_j}{V_{\max} - V_{\min}}\right)^2$$

  In the case of Euclidean loss, we penalize according to a normalized Euclidean distance in force-velocity space, so that large errors are penalized more than small errors. Define the pair $\{F_i, V_i\}$ as the centroid of the region in force-velocity space corresponding to C_i.

- **Playability loss** A drawback of Euclidean loss is, though parameters may be "close" in force-velocity space, the effect on the sound will be appreciable. This is especially true when the true values of force and velocity correspond to Helmholtz motion but the estimated values do not, or vice versa. Let $\mathcal{P} \subset [F_{\min}, F_{\max}] \times [V_{\min}, V_{\max}]$ describe the playability region for Helmholtz motion in force-velocity space. Then we define

  $$\begin{aligned} L_p(k, j) = {} & \mathbf{1}_{(\{\{\mathbf{F_k},\mathbf{V_k}\}\in\mathcal{P}\}\cap\{\{\mathbf{F_j},\mathbf{V_j}\}\in\mathcal{P}^c\})} \\ & + \mathbf{1}_{(\{\{\mathbf{F_k},\mathbf{V_k}\}\in\mathcal{P}^c\}\cap\{\{\mathbf{F_j},\mathbf{V_j}\}\in\mathcal{P}\})} \end{aligned}$$

  Average playability loss approximates the classification error with respect to the playability region.

4.2 Classification results

Figure 2 shows the results of the classification algorithm in the case of binary loss versus number of Gaussians. We obtain about 75% correct classifications in the case of 13 LPC coefficients.

All the average loss figures were obtained running the classification algorithm using 100 independent test waveforms.

In order to test the influence of the number of features in the classification, we tested the percentage of success versus PCA dimension. The results show how the classification is completely unreliable when less than 4 features are used, while the results using between 5 and 13 features are rather comparable.

Figures 3,4, and 5 show the results of the classification algorithm in case of, respectively, *Euclidean, Playability*, and *Combined* loss, again plotted versus PCA dimension. The combined loss function is probably the most useful for parameter identification, as it requires parameter estimates to be accurate *and* gives a special penalty for misclassifying waveforms with respect to the playability region. To evaluate the combined loss we add the normalized quadratic and playability losses. The normalization is done so that the theoretical maximum loss in either case is 0.5.

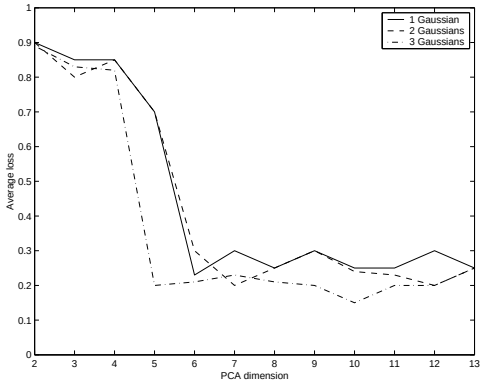

Figure 2: Percentage of success of the classification algorithm in the case of binary loss for different dimensions of the feature space.

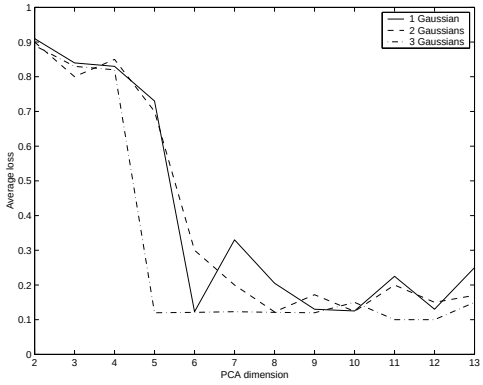

Figure 3: Average quadratic loss for different dimensions of the feature space.

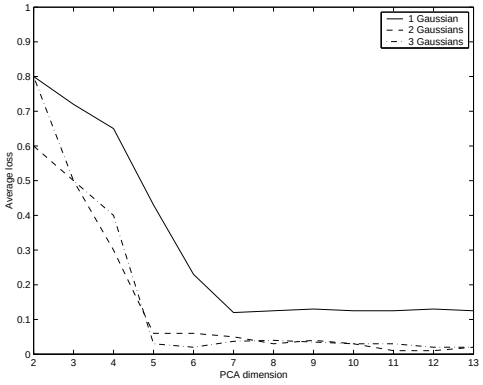

Figure 4: Average 'playability' loss for different dimensions of the feature space.

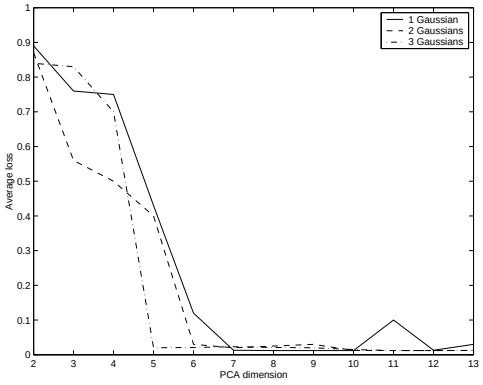

Figure 5: Averaged combined Euclidean and 'playability' loss for different dimensions of the feature space.

5 Application

We merged the parameters estimation technique with a computer animation framework to create an animation of a violinist which runs in real time on a standard PC. The simulation is accurate enough to realistically reproduce the expressive motion of the player.

Using OpenGL to implement the animation allows us to create a real-time graphical interface driven by the audio physical model. The keyframed animations are built in 3DSMax, and imported in the OpenGL interface. The controllers' activity then triggers special animations. The scene is mainly made of a 12000 polygon mesh of a human skeleton, a 1400 polygon mesh of a violin and a 110 polygon mesh of a bow.

In order to animate the character and to use the inverse kinematics implementations incorporated in 3DSMax, each bone of the skeleton is attached to a wire architecture generated by Character Studio's biped plugin. Each individual phalanx needs to be precisely linked to the biped structure in order to enable the skeleton to perform the movements necessary to play the violin.We build independent animations for

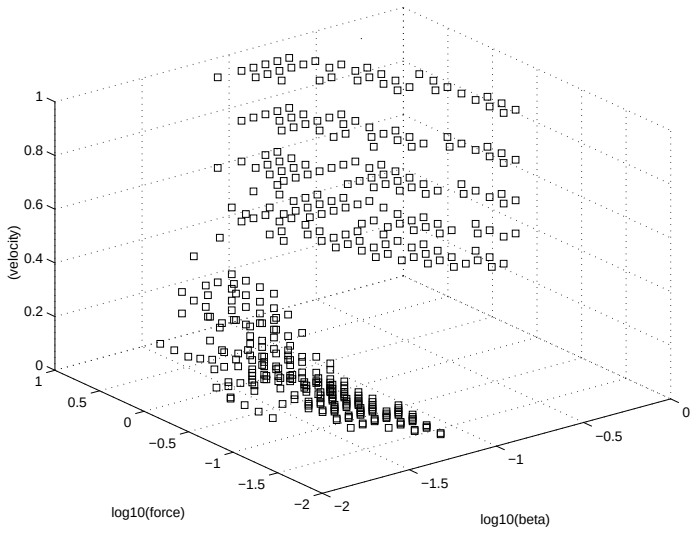

Figure 6: The squares show the combinations of input parameters that allow to achieve Helmholtz motion (in logarithmic scale for bow position beta and bow force). This is the playable region of the model.

independent movements: one animation per bow stroke, one animation per string played, and so on. Once loaded in the OpenGL program, the biped structure allows interpolations between the different animations, so that the skeleton can respond to several triggered events at the same time (e.g: play on the G string and move to the 3rd position).

6 Merging sound synthesis and animation

A two-thread implementation is required for operation in real-time. One thread handles sound synthesis; the other handles the animation

The animation thread also handles the controllers' activity and changes variables that the sound thread reads to produce the music. Data continually flows through the controllers' thread and the sound thread retrieves its information as quickly as possible. The sound thread is actually given a higher priority than the controllers thread.

7 Conclusions

In this paper, we proposed a classification-based technique to estimate the input parameters for bowed-string waveforms synthesized using the physical model of (4). Our technique is able to reconstruct the bow position, bow force and bow velocity inputs given recorded steady-state waveforms. These inputs can be used to control the animation of a virtual violin player.

Figure 7: *Virtual violinist playing the A string.*

8 Acknowledgments

We would like to thank Prof. Christoph Bregler for allowing us to use the facili ties of the Computer Graphics Laboratory at Stanford University.

References

[1] Casey, M.A. "Understanding Musical Sounds with Forward Models and Physical Models" Connection Science, 6, pp. 355-371, 1994.

[2] Dempster, A.P., Laird, N.M., and and Rubin, D.B. Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society B 39, 1-38, 1977.

[3] Smith, J. O., "Physical modeling using digital waveguides," *Computer Music Journal, vol. 16, n. 4, pp. 74-91, 1992.*

[4] Serafin, S., Smith, J.O., Woodhouse, J., An investigation of the impact of torsion waves and friction characteristics on the playability of virtual bowed string instruments", IEEE WASPAA, Mohonk, NY, Oct. 1999

[5] Traube, C., and Smith, J. O., "Estimating the plucking point on a guitar string," *Proc. COST G-6 conference on Digital Audio Effects, (DAFX-00), Verona, Italy, Dec. 2000.*

[6] Bishop, C. "Neural Networks for Pattern Recognition," *Clarendon Press, Oxford, 1995.*