AUDIO-BASED GESTURE EXTRACTION ON THE ESITAR CONTROLLER

Ajay Kapur, George Tzanetakis, Peter F. Driessen

University of Victoria
Victoria, British Columbia, CANADA
ajay@ece.uvic.ca, gtzan@cs.uvic.ca, peter@ece.uvic.ca

ABSTRACT

Using sensors to extract gestural information for control parameters of digital audio effects is common practice. There has also been research using machine learning techniques to classify specific gestures based on audio feature analysis. In this paper, we will describe our experiments in training a computer to map the appropriate audio-based features to look like sensor data, in order to potentially eliminate the need for sensors. Specifically, we will show our experiments using the *ESitar*, a digitally enhanced sensor based controller modeled after the traditional North Indian sitar. We utilize multivariate linear regression to map continuous audio features to continuous gestural data.

1. INTRODUCTION

Using hyper instruments that utilize sensors to extract gestural information for control parameters of digital audio effects is common practice. However, there are many pitfalls in creating sensor-based controller systems. Purchasing microcontrollers and certain sensors can be expensive. The massive tangle of wires interconnecting one unit to the next can get failure-prone. Things that can go wrong include: simple analog circuitry break down, or sensors wearing out right before a performance forcing musicians to carry a soldering iron along with their tuning fork. The biggest problem with hyper-instruments, is that there usually is only one version, and the builder is the only one that can benefit from the data acquired. These problems have motivated our team to attempt to use sensor data so that sensors become obsolete. More specifically we use sensor data to train machine learning models, evaluate their performances and then use the trained acoustic-based models to replace the sensor.

The traditional use of machine learning in audio analysis has been in classification where the output of the system an ordinal value (for example the instrument name). In this work, we explore regression which refers to machine learning systems where the output is a continuous variable. One of the challenges in regression is obtaining data for training. To solve this problem, we use the sensor data to train the model that will replace the sensor. In our experiments, we use audio-based feature extraction with synchronized continuous sensor data to train a "pseudo" sensor using machine learning.

Specifically, we show our experiments using the Electronic Sitar (*ESitar)*, a digitally enhanced sensor based controller modeled after the traditional North Indian sitar.

In this paper the following will be discussed:

- Background on related work using machine learning techniques for extraction of gesture information
- Description of sensor-based gesture extraction using the *ESitar* controller
- Audio-based feature extraction and multivariate feature extraction
- Experiments on audio-based gesture extraction on the *ESitar* controller

2. RELATED WORK

There has been a variety of research using machine learning techniques to classify specific gestures based on audio feature analysis. The extraction of control features from the timbre space of the clarinet is explored in [1]. Deriving gesture data from acoustic analysis of a guitar performance is explored in [2, 3, 4]. An important influence for our research is the concept of indirect acquisition of instrumental gesture described in [4]. Gesture extraction from drums is explored in [5, 6]. All the cited papers fall into two broad categories: (1) methods that rely on signal processing to directly map the sound to gesture parameter and typically work for continuous data, and (2) methods that use machine learning to extract categorical information. The former tend to be sensitive to noise and other uncertainties in the sensor data, and therefore not suitable for use in actual performance. The later although robust to noise don't handle continuous gestural data. In addition, the mapping from sound to gesture in many cases is not straight-forward and machine learning is necessary to obtain a useful mapping. Using regression, our approach deals with continuous gestural data while retaining the robustness achieved by machine learning.

3. DIGITIZING SITAR GESTURES

In this Section we will briefly describe background of traditional sitar technique followed by a description of a sensor based controller which digitizes a sitar players gestures. This brief description is included to inform the presentation of the audio-based gestural analysis described later. More background and details about these topics can be found in [7].

3.1. Sitar and playing technique

The sitar is *Saraswati's* (the Hindu Goddess of Music) 19-stringed, pumpkin shelled, traditional North Indian instrument. Its bulbous gourd (shown in Figure 1), cut flat on the top, is joined to a long necked hollowed concave stem that stretches three feet long and three inches wide. The sitar contains seven strings on the upper bridge, and twelve sympathetic stings below all tuned by tuning pegs. The upper strings include rhythm and drone strings, known as *chikari*. Melodies, which are primarily performed on the

upper-most string and occasionally the second string, induce sympathetic resonances in the twelve strings below. The sitar can have up to 22 moveable frets, tuned to the notes of a *Raga* (the melodic mode, scale, order, and rules of a particular piece of Indian classical music) [8,9,10,11].

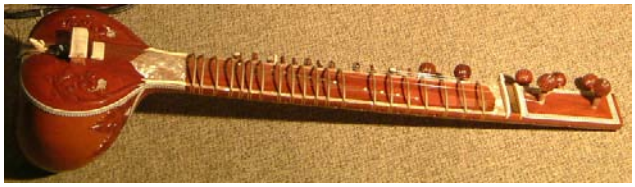

Figure 1: *The traditional North Indian sitar.*

It is important to understand the traditional playing style of the sitar to comprehend how our controller captures its hand gestures. Our controller design has been informed by the needs and constraints of the long tradition and practice of sitar playing. It should be noted that there are two main styles of sitar technique: Ustad Vilayat Khan's system and Pandit Ravi Shankar's system. [10] The *ESitar* is modeled based on the Vilayat Khan system.

Monophonic melodies are performed primarily on the outer main string, and occasionally on the copper string. The sitar player uses his left index finger and middle finger, as shown in Figure 2(a), to press the string to the fret to play the desired *swara* (note). The frets are elliptically curved so the string can be pulled downward, to bend to a higher note. This is how a performer incorporates the use of *shruti* (microtones) which is an essential characteristic of traditional classical Indian music.

On the right index finger, a sitar player wears a ring like plectrum, known as a *mizrab.* The right hand thumb, remains securely on the edge of the *dand* (neck) as shown on Figure 3(a), as the entire right hand gets pulled up and down over the main seven strings, letting the *mizrab* strum the desired melody. An upward stroke is known as *Dha* and a downward stroke is known as *Ra.* [9,10]. The two main gestures we capture using sensors and subsequently try to model using audio-based analysis are: 1) the pitch/fret position and 2) the *mizrab* stroke direction. The corresponding sensors are described in the following Subsection.

3.2. The ESitar Controller

The *ESitar* was built with the goal of capturing a variety of gestural input data. For our experiments, we are interested in gestures that deduce monophonic pitch detection and mizrab pluck direction. A variety of different families of sensor technology and signal processing methods are used, combined with Atmel AVR ATMega16 microcontroller[12], which serves as the brain of the *ESitar*.

3.2.1. Fret Detection

The currently played fret is deduced using an exponentially distributed set of resistors which form a network interconnecting in series each of the frets on the *ESitar* (pictured in Figure 2(b)). When the left hand fingers depress the string to touch a fret (as shown in Figure 2(a)), current flows through the string and the segment of the resistor network between the bottom and the played fret. The voltage drop across the in-circuit segment of the resistor network is digitized by the microcontroller. Using a lookup table it maps each unique value to a corresponding fret number and sends it out as a MIDI message.

(a) (b)

Figure 2: *(a) Traditional fingers playing fret technique; (b) Network of resistors on the frets of the ESitar.*

As mentioned before, the performer may pull the string downward, bending a pitch to a higher note (for example play a *Cb* from the *A* fret). To capture this additional information that is independent of the played fret, we fitted the instrument with a piezo pick-up to be fed into a pitch detector. We chose to implement the pitch detector as a *pure data* [13] external object using an auto-correlation based method [14]. The pitch detection is bounded below by the pitch of the currently played fret and allowed a range of eight semi-tones above to help eliminate octave errors.

3.2.2. Mizrab Pluck Direction

We are able to deduce the direction of a *mizrab* stroke using a force sensing resistor (FSR), which is placed directly under the right hand thumb, as shown in Figure 3. As mentioned before, the thumb never moves from this position while playing, however, the applied force varies based on *mizrab* stroke direction. A *Dha* stroke (upward stroke) produces more pressure on the thumb than a *Ra* stroke (downward stroke). We send a continuous stream of data from the FSR via MIDI, because this data is rhythmically in time and can be used compositionally for more then just deducing pluck direction.

(a) (b)

Figure 3: *(a) Traditional mizrab technique (notice thumb position); (b) FSR sensor used to measure thumb pressure.*

4. AUDIO-BASED ANALYSIS

In this Section we describe how the audio signal is analyzed. For each short time segment of audio data numerical features are calculated. At the same time, sensor data is also captured. These two

steams of data potentially have different sampling rates. In addition, in some cases, the gestural data is not regularly sampled. We have developed tools to align the two streams of data for these cases. Once the features are aligned with the sensor data, we train a "pseudo" sensor using regression .and explore its performance.

4.1. Audio-Based feature extraction

The feature set used for the experiments described in this paper is based on standard features used in isolated tone musical instrument classification, music and audio recognition. It consists of 4 features computed based on the Short Time Fourier Transform (STFT) magnitude of the incoming audio signal. It consists of the Spectral Centroid (defined as the first moment of the magnitude spectrum), Rolloff and Flux as well as RMS energy. More details about these features can be found in [15]. The features are calculated using a short time analysis window with duration 10-40 milliseconds. In addition, the means and variances of the features over a larger texture window (0.2-1.0 seconds) are computed resulting in a feature set with 8 dimensions. The large texture window captures the dynamic nature of spectral information over time and it was a necessary addition to achieve any results in mapping features to gestures. Ideally the size of the analysis and texture windows should correspond as closely as possible to the nature time resolution of the gesture we want to map. In our experiments we have looked at how these parameters affect the desired output. In addition, the range of values we explored was determined empirically by inspecting the data acquired by the sensors.

4.2. Audio-Based Pitch Extraction

The pitch of the melody string (without the presence of drones) is extracted directly from the audio signals using the method described in [16]. This method is an engineering simplification of a perceptually-based pitch detector and works by slitting the signal into two frequency bands (above and bellow 1000Hz), applying envelope extraction on the high-frequency band followed by enhanced autocorrelation (a method for reducing the effect of harmonic peaks in pitch estimation). Figure 4 shows a graph of a simple ascending diatonic scale calculated directly from audio analysis.

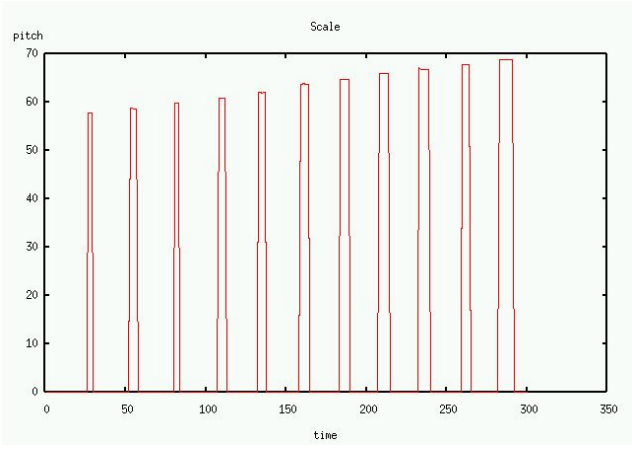

Figure 4: *Graph of Audio-Based Pitch extraction on an ascending diatonic scale without drone strings being played.*

The audio-based pitch extraction is similar to many existing systems that do not utilize machine learning therefore it will not be further discussed. Currently the audio-based pitch extraction works only if the drone strings are not audible. We are planning to explore a machine learning approach to pitch extraction when the drone strings are sounding in the future.

The interaction between sensors and audio-based analysis can go both ways. For example we used the audio-based pitch extractor to debug and calibrate the fret-sensor. Then the fret sensor can be used as ground truth for machine learning of the pitch in the presence of drone strings. We believe that this bootstrapping process can be very handy in the design and development of gestural music interfaces in general.

4.3. Regression Analysis

Regression refers to the prediction of real-valued outputs from real-valued inputs. Multivariate regression refers to predicting a single real-valued output from multiple real-valued inputs. A classic example is predicting the height of a person using their measure weight and age. There are a variety of methods proposed in the machine learning [17] literature for regression. For the experiments described in this paper, we use linear regression where the output is formed as a linear combination of the inputs with an additional constant factor. Linear regression is fast to compute and therefore useful for doing repetitive experiments for exploring the parameter. We also employ a more powerful back propagation neural network [18] that can deal with non-linear combinations of the input data. The neural network is slower to train but provides better regression performance. Finally, the M5 prime decision tree based regression algorithm was also used [19]. The performance of regression is measured by a correlation coefficient which ranges from 0.0 to 1.0 where 1.0 indicates a perfect fit. In the case of gestural control, there is significant amount of noise and the sensor data doesn't necessarily reflect directly the gesture to be captured. Therefore, the correlation coefficient can mainly be used as a relative performance measure between different algorithms rather than an absolute indication of audio-based gestural capturing.

5. AUDIO-BASED GESTURE EXTRACTION

5.1. Data Collection

In order to conduct the experiments the following tools were used to record audio and sensor data. Audio files were recorded with DigiDesign's ProTools Digi 002 Console using a piezo pickup (shown in Figure 4) placed directly on the sitar's *tabli*. Midi data was piped through *pure data* [13] (`http://pure-data.sourceforge.net/`) where it was filtered and sent to a custom built Midi Logger program which recorded time stamps and all midi signals. Feature extraction of the audio signals was performed using *Marsyas* [20] (`http://marsyas.sourceforge.net`). The sampling rate of the audio files and the sensor data were not the same. The audio data was sampled at 44100 Hz and then downsampled for processing to 22050 Hz. Also the sensor data was not regularly sampled. Tools were developed to align the data for use with *Weka* [21] (`http://www.cs.waikato.ac.nz/ml/weka/`), a tool for data mining with a collection of various machine learning algorithms.

For the experiments, excerpts of a *jhala* portion of a *raga* were performed on the *ESitar*. *Jhala* is a portion of sitar performance characterized by the constant repetition of pitches, including the drone, creating a driving rhythm. [10] Because of the rhythmic nature of this type of playing we chose to explore the signals of the thumb sensor to get an inclination of *mizrab* pluck direction using audio-based feature analysis and regressive machine learning algorithms.

5.2. Experiments using Regression Analysis

Our first experiment was to analyze the effect of the analysis window size used for audio based feature extraction. Table 1 shows the results from this experiment. Take note that the texture size remained constant at 0.5 seconds and linear regression was used. The correlation coefficient for random inputs is 0.14. It is apparent based on the table that an analysis window of length 256 (which corresponds to 10 milliseconds) achieves the best results. It can also be seen that the results are significantly better than chance. We used this window size for the next experiment.

Analysis Window Size (samples at 22.5 KHz)	128	256	512
Correlation Coefficient	0.2795	0.3226	0.2414

Table 1: *Effect of analysis window size.*

The next experiment explored the effect of texture window size and choice of regression method. Table 2 shows the results from this experiment. The rows correspond to regression methods and the columns correspond to texture window sizes expressed in number of analysis windows. For example, 40 corresponds to 40 windows of 256 samples at 22050 Hz sampling rate which is approximately 0.5 seconds. To avoid overfitting we use a percentage split where the first 50% of the audio and gesture data recording is used to train the regression algorithm which is subsequently used to predict the second half of recorded data.

	10	***20***	***30***	***40***
Random Input	0.14	0.14	0.14	0.14
Linear Regression	0.28	0.33	0.28	0.27
Neural Network	0.27	0.45	0.37	0.43
M5' Regression Method	0.28	0.39	0.37	0.37

Table 2: *Effect of texture window size (columns) and regression method (rows).*

It is evident from the table and Figure 5 that the best choice of texture window size is 20 which corresponds to 0.25 seconds. In addition, the best regression performance was obtained using the back propogation neural network. Another interesting observation is that the relation of inputs to outputs is non-linear as can be seen from the performance of the neural network and M5' regression algorithm compared to the linear regression.

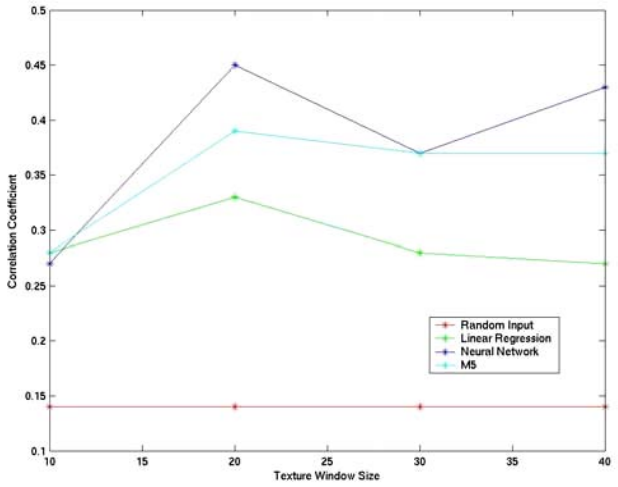

Figure 5: *Graph showing the effect of texture window size and regression method.*

6. CONCLUSIONS AND FUTURE WORK

In this paper, we propose the use of machine learning methods and more specifically regression for replacing sensors with analysis of the audio input data. A proof-of-concept experiment to verify this idea was conducted using the ESitar controller. Our preliminary results indicate that regression can be used to predict non-trivial continuous control data such as the thumb sensor of the ESitar without using sensors. Previous methods either relied on complex signal processing design and were not robust to noise or only dealt with classification into discrete labels. Our approach handles gracefully both problems with minimal user involvement as the training is performed using the sensor we are trying to replace without requiring any human annotation. We show how the choice of regression method and analysis parameters affects the results for a particular recording. There is a lot of work to be done in exploring how this approach can be used and this is only the beginning.

There are many directions for future work. We are exploring the use of additional audio-based features such as Mel-Frequency Cepstral Coefficients (MFCC) and Linear Prediction Coefficients (LPC). We are also gathering more recording to use for analysis. Creating tools for further processing the gesture data to reduce the noise and outliers is another direction for future research. We would also like to try and predict other types of gestural data such as fret position. Another eventual goal is to use this technique to transcribe the sitar and other Indian music. We are also interested in using this method for other instruments such as the tabla and snare drum.

7. ACKNOWLEDGEMENTS

Many thanks to Scott Wilson, Ari Lazier, Phil Davidson, Michael Gurevich, Bill Verplank, and Perry Cook for their contribution in building the *ESitar*. Another thanks to Phil Davidson for his Midi Logger program used to record sensor data and Tiffany Jenkins for her help with data acquisition.

8. REFERENCES

[1] E. B. Egozy, "Deriving Musical Control Features from a Real-Time Timbre Analysis of the Clarinet," *Masters Thesis,* Department of Electrical Engineering and Computer Science. Massachusetts Institute of Technology, 1995.

[2] N. Orio, "The timbre space of the classical guitar and its relationship with plucking techniques," *Proc. of Int. Computer Music Conf. (ICMC-99),* Bejing, China, 1999.

[3] C. Traube, J. O. Smith, "Estimating the plucking point on a guitar string," *Proc. of Digital Audio Effects Conf. (DAFX-00)*, Verona, Italy, 2000.

[4] C., Traube, P. Depalle, and M. Wanderly, "Indirect Acquisition of Instrumental Gesture Based on Signal, Physical and Perceptual Information," *Proc. of 2003 Conf. on New Instruments for Musical Expression (NIME-03),* Montreal, Canada, May 2003.

[5] F. Gouyon, P. Herrera, "Exploration of techniques for automatic labeling of audio drum tracks instruments," *Proc. of MOSART: Workshop on Current Directions in Computer Music*, 2001.

[6] J. Silpanpaa, "Drum stroke recognition," Technical report, Tampere University of Technology. 2000.

[7] A. Kapur, A. Lazier, P. Davidson, S. Wilson, and P. R. Cook, "The Electronic Sitar Controller," *Proc. of 2004 Conf. on New Instruments for Musical Expression (NIME-04),* Hamamatsu, Japan, June 2004.

[8] R. R. Menon, *Discovering Indian Music*. Mumbai, India: Somaiya Publications PVT. LTD, 1974.

[9] R. A. Vir, *Learn to play on Sitar*. New Delhi, India: Punjab Publications, 1998.

[10] S. Bagchee, *NAD: Understanding Raga Music.* Mumbai, India: Ceshwar Business Publications Inc., 1998.

[11] S. Sharma, *Comparative Study of Evolution of Music in India & the West*. New Delhi, India: Pratibha Prakashan, 1997.

[12] S. Wilson, M. Gurevich, B. Verplank, and P. Stang, "Microcontrollers in Music Education - Reflections on our Switch to the Atmel AVR Platform," *Proc. of 2003 Conf. on New Instruments for Musical Expression (NIME-03)*, Montreal, Canada, May 2003.

[13] M. Puckette, "Pure Data: Another Integrated Computer Music Environment," *Proc. of Second Intercollege Computer Music Concerts,* Tachikawa, Japan, 1996, pp. 37–41.

[14] U. Zölzer (ed.), *DAFX: Digital Audio Effects*, John Wiley & Sons, Ltd. England, pp. 336–370, 2002.

[15] G. Tzanetakis, P. R. Cook. "Musical Genre Classification of Audio Signals," *IEEE Trans. on Speech and Audio Processing,* vol. 10, no. 5, July 2002.

[16] T. Tolonen, M. M. Karjalainen, "A computationally efficient multipitch analysis model," *IEEE Trans. Speech Audio Processing*, vol. 8, pp. 708–716, Nov. 2000.

[17] T. Mitchelle, *Machine Learning.* Singapore: McGraw Hill, 1997.

[18] S. Y. Kung, *Digital Neural Network.* Englewood Cliffs, NJ: PTR Prentice Hall, 1993.

[19] J. R. Quinlan, "Learning with {C}ontinuous {C}lasses," *5th Australian Joint Conference on Artificial Intelligence,* 1992, pp. 343-348.

[20] George Tzanetakis, Perry Cook, "MARSYAS: A Framework for Audio Analysis," *Organized Sound, Cambridge University Press,* vol. 4, no. 3, 2000.

[21] H. Ian, E. Frank, and M. Kaufmann, *Data Mining: Practical machine learning tools with Java implementations*. San Francisco, 2000.