# Toward Inverse Control of Physics-Based Sound Synthesis

A. Pfalz[1] and E. Berdahl[1]

[1]Louisiana State University

Long Short-Term Memory networks (LSTMs) can be trained to realize inverse control of physics-based sound synthesizers. Physics-based sound synthesizers simulate the laws of physics to produce output sound according to input gesture signals. When a user's gestures are measured in real time, she or he can use them to control physics-based sound synthesizers, thereby creating simulated virtual instruments.

An intriguing question is how to program a computer to learn to play such physics-based models. This work demonstrates that LSTMs can be trained to accomplish this inverse control task with four physics-based sound synthesizers.

**Keywords:** LSTM, physics-based models, sound synthesis

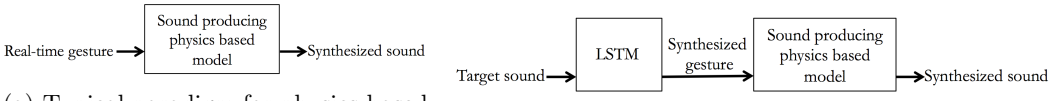
## 1 Introduction

### 1.1 Sound Matching

Controlling the parameters of sound synthesizers in order to realize target sounds has been a challenge for decades. For instance, with the Frequency Modulation (FM) synthesis technique, the connection between the carrier frequencies, modulation frequencies, and modulation indices (particularly as these are automated) and the sound produced is complicated. Accordingly, researchers have applied various techniques for adapting FM parameters to achieve target sounds such as Horner et al. [1993], Garcia [2001], Lai et al. [2006], and Tan and Lim [1996]. Similar work as also been conducted for tuning the parameters of physics-based models; however, most of these works have required very specific optimizations that apply only to certain models (see Sondhi and Resnick [1983] and Riionheimo and Välimäki [2002]).

### 1.2 Inverse Control

However, since humans are able to learn to play musical instruments, it seems very likely that machine learning methods could help address the inverse control problem. For example, deep as well as shallow learning for audio and music generation has been investigated from a number of different perspectives. One approach is generating sequences of notes using MIDI or symbolic notation (as in Waite [2016]). WaveNet is another very promising project among others (van den Oord et al. [2016]).




(a) Typical paradigm for physics-based sound synthesis (forward control).

(b) Proposed paradigm (inverse control).

Figure 1: Paradigms for physics-based sound synthesis.

The advantage of this technique is that it allows the generation of sounds to be automated. If a complex sound is desired, the user needs to spend time practicing to execute a potentially difficult gesture to be able to produce the output audio.

# 2 Approach

## 2.1 Overview

Synth-A-Modeler is an open-source software project for building physics-based sound synthesizers (see Berdahl and Smith III [2012]). Figure 1a shows the typical paradigm for physics-based sound synthesis. In contrast, the present work proposes a new paradigm in which an LSTM learns to realize inverse control of a physics-based sound synthesizer. In other words, the LSTM uses a *target sound* to match the *gesture* that was used to create the *target sound* using a physics-based sound synthesizer. Consequently, the predicted *synthesized gesture* can be applied to a physics-based sound synthesizer to create a *synthesized sound*. Although not studied directly in this introductory work, this paradigm has potential applications denoising audio recordings, toolkits for making foley, sound transformations and sound mappings.

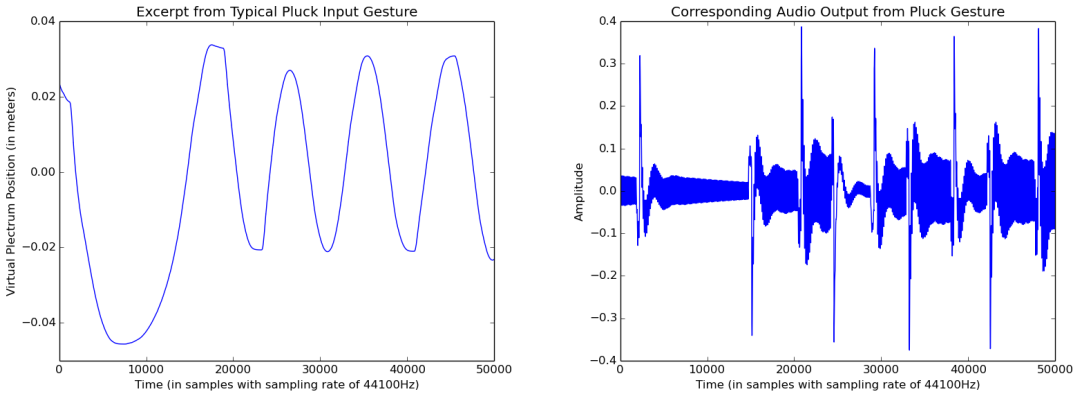
## 2.2 Physics-Based Sound Synthesizers Used in the Project

Each physics-based sound synthesizer used in this work receives a gesture signal that could for example represent the position of a musician's hand in real-time, enabling her or him to play virtual sound synthesizers. These gesture signals are in the range of $-0.05$ m to 0.05 m (see Figure 2a for an example gesture) with an audio sampling rate of 44.1kHz. Depending on the particular sound synthesizer type, the gesture excites the synthesizer in a different way.

For example, the sound synthesizer `PluckAResonator.mdl` incorporates a virtual plucking mechanism that activates a single, virtually oscillating resonator. Accordingly, the sound is triggered when the plectrum is pushed beyond the 0 m point (see Figure 2).[1]

For example, as the gesture signal moves up from $-0.05$ m, there will at first be no sound synthesized until the virtual plectrum moves close enough to interact with the virtual resonator (see Figure 2). Then, as the gesture continues moving up, the plectrum "plucks" the virtual resonator causing it to vibrate and produce sound (see Figure 2 near $t = 16000$ samples). Notice that the spikes in amplitude in the synthesized sound signal correspond approximately to zero crossings in the gesture signal.

[1] To model the dynamics of a plectrum more precisely, the sound is actually triggered when the plectrum is pushed a small distance $d$ meters beyond the *difference* between the virtual gesture input signal and the current position of the resonator, where $d$ changes sign with each pluck. The details of how the sound synthesizers work are beyond the scope of this paper. Their dynamical behaviors are complex, nonlinear and nuanced, as is appropriate for modern physics-based sound synthesis.




(a) Gesture signal.

(b) Synthesized sound output.

Figure 2: Gesture signal input and corresponding synthesized sound output for a physics-based model with a plucking excitation. (The horizontal axis is time in samples.)

Different nonlinear excitation mechanisms were employed in the models described below, to verify that the LSTMs could, just like human musicians, learn to produce a wide range of gestures for this application. For example, the synthesizer `TouchSeveralModalResn.mdl` was controlled using a kind of "nonlinear contact link," `ScratchMassLinkChain.mdl` incorporated a bowed-string kind of interaction, and `PluckHarp10.mdl` featured ten individually pluckable virtual strings, each with its own distinct plucking point across the input gesture range.

## 2.3 An LSTM Network for Inverse Control

Long Short Term Memory (LSTM) networks are known for their sequence prediction abilities (see Hochreiter and Schmidhuber [1997]). In this work, it was decided to try using LSTMs for inverse control in order to see if an LSTM could learn to play music using physics-based sound synthesizers.

LSTMs were implemented using the high level TensorFlow API `contrib.rnn`. The models had two or three layers each with 1024 units implemented with `contrib.rnn.MultiRNNCell` wrapper with default settings except where indicated otherwise (Abadi et al. [2015]). The loss function used the mean-squared error to measure the similarity between a synthesized gesture $\hat{y}$ and a gesture $y$ that was used to produce a target sound $x$. The Adam Optimizer was used. Figure 3 shows the process of generating the data and training an LSTM.

In order to capture the most data with a single input, the audio data in $x$ was downsampled from 44100Hz to 2756Hz (a speech-quality sampling rate) before showing it to the LSTM. To obtain gesture signals, six-minute recordings were made with each physics-based sound synthesizer of a human musician performing musical gestures using a single degree-of-freedom haptic device (Berdahl and Kontogeorgakopoulos [2012]). This recording was then broken into segments that were 1024 samples long. Batches of 98 inputs were used per training iteration. The models were trained over 64 epochs. The validation and testing datasets were each 10% of the original six-minute corpus.

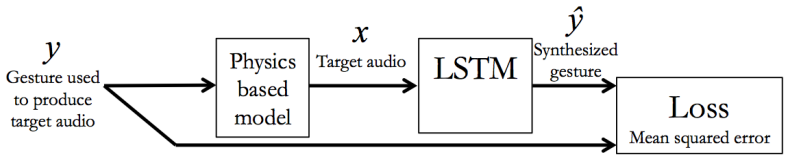



Figure 3: Diagram of how loss is computed.

## 3 Results

Audio examples of results from these experiments are available at `https://cct.lsu.edu/~apfalz/inverse_control.html`. In each of the files, the left channel is the synthesized gesture $\hat{y}$. The right channel is the target audio $x$. All the examples come from the test set.

Audio example 1 shows the simplest of the physics based models. The resultant audio matches the target audio quite closely. There are slight differences in the amplitude and phase on the audio. The authors found the differences are barely noticeable and only when listening on headphones. Audio example 2 shows comparable results with the physics based model that uses a virtual touch for its mode of interaction.

Audio example 3 shows the most impressive results from these experiments. In this example, the LSTM had to learn to consider the frequencies of the target audio it was trying to match. In some informal trials not shown here, the LSTM was shown to be able to predict comparably accurate gestures for simpler physics based models like the one shown in audio example 1 when the LSTM was shown only the RMS level of the target audio, rather than the raw audio itself.

Audio example 4 shows the worst performance. This was expected because the target audio is decidedly more complex than the other inputs. The LSTM was still able to predict a gesture that closely matched the target audio. However, there the gesture contained some extraneous excitations.

The normalized absolute error for the test set was calculated using equation 1.

$$NormalizedAbsoluteError = \frac{\frac{1}{N}\sum_{i=1}^{N}|Y_i - \hat{Y}_i|}{\frac{1}{N}\sum_{i=1}^{N}|\hat{Y}_i|}, \tag{1}$$

where $Y$ is the targets and $\hat{Y}$ is the prediction. The model was able to synthesize the target gesture very accurately for `TouchSeveralModalResn.mdl` and `PluckHarp10.mdl` with 4.12% and 2.20% error respectively. `TouchSeveralModalResn.mdl` and `ScratchMassLinkChain.mdl` performed worse with 22.37% and 16.84% error respectively. The audio that resulted from these gestures still matched the target audio closely though.

## 4 Conclusions and Future Work

The LSTM was able to synthesize gestures for inverse control of physics-based sound synthesis. The synthesized gestures could be used with physics-based models for re-synthesizing audio that, at least in the opinion of the authors, subjectively resembled the target audio quite closely, the authors hope that readers will visit the project web page and listen to the sound examples to judge for themselves. Moreover, the gestures synthesized by the LSTM matched the target gestures. Applications of this technology could include denoising of audio recordings, toolkits for making foley, as well as new kinds of sound transformations and mappings, which can be

achieved by applying the synthesized gesture signals to a diverse range of physics-based sound synthesizers.

More generality might come from training in such a way that the LSTM produces a synthesized gesture that would produce match the target audio but without matching the particular gesture that created the input to LSTM. Measuring the loss against the audio directly rather than against the gestures would remedy this. The physics based models are capable of a wider range of sounds than are demonstrated in the audio inputs presented here. It should be investigated to what extent an LSTM could model both the input gesture and time-varying parameters like fundamental frequency or amplitude. Also larger datasets can be generated randomly instead of only using input from a human.

## References


Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `http://tensorflow.org/`. Software available from tensorflow.org.

Edgar Berdahl and Alexandros Kontogeorgakopoulos. The firefader design: Simple, open-source, and reconfigurable haptics for musicians. In *Proceedings of the 9th Sound and Music Computing Conference*, pages 90–98, 2012.

Edgar Berdahl and J Smith III. An introduction to the synth-a-modeler compiler: Modular and open-source sound synthesis using physical models. In *Proceedings of the Linux Audio Conference*, 2012.

R Garcia. Growing sound synthesizers using evolutionary methods. In *Proceedings ALMMA 2001: Artificial Life Models for Musical Applications Workshop,(ECAL 2001)*, 2001.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8): 1735–1780, November 1997. ISSN 0899-7667.

Andrew Horner, James Beauchamp, and Lippold Haken. Machine tongues XVI: Genetic algorithms and their application to FM matching synthesis. *Computer Music Journal*, 17(4): 17–29, 1993.

Yuyo Lai, Shyh-Kang Jeng, Der-Tzung Liu, and Yo-Chung Liu. Automated optimization of parameters for fm sound synthesis with genetic algorithms. In *International Workshop on Computer Music and Audio Technology*, page 205. Citeseer, 2006.

Janne Riionheimo and Vesa Välimäki. Parameter estimation of a plucked string synthesis model with genetic algorithm. In *ICMC*, 2002.

Man Mohan Sondhi and JR Resnick. The inverse problem for the vocal tract: Numerical methods, acoustical experiments, and speech synthesis. *The Journal of the Acoustical Society of America*, 73(3):985–1002, 1983.

BTG Tan and SM Lim. Automated parameter optimization of double frequency modulation synthesis using the genetic annealing algorithm. *Journal of the audio Engineering Society*, 44(1/2):3–15, 1996.

Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *CoRR abs/1609.03499*, 2016.

Elliot Waite. Generating long-term structure in songs and stories. `https://magenta.tensorflow.org/2016/07/15/lookback-rnn-attention-rnn/`, 2016.