Slide guitar synthesizer with gestural control

Jyri Pakarinen
Department of Signal Processing and Acoustics
Helsinki University of Technology
P.O. Box 3000
FI-02015 TKK, Finland
jyri.pakarinen@tkk.fi

Vesa Välimäki
Department of Signal Processing and Acoustics
Helsinki University of Technology
P.O. Box 3000
FI-02015 TKK, Finland
vesa.valimaki@tkk.fi

Tapio Puputti
Helsinki University of Technology
P.O. Box 3000
FI-02015 TKK, Finland
tapio.puputti@tkk.fi

ABSTRACT

This article discusses a virtual slide guitar instrument, recently introduced in [7]. The instrument consists of a novel physics-based synthesis model and a gestural user interface. The synthesis engine uses energy-compensated time-varying digital waveguides. The string algorithm also contains a parametric model for synthesizing the tube-string contact sounds. The real-time virtual slide guitar user interface employs optical gesture recognition, so that the user can play this virtual instrument simply by making slide guitar playing gestures in front of a camera.

Keywords

Sound synthesis, slide guitar, gesture control, physical modeling

1. INTRODUCTION

The term slide- or bottleneck guitar refers to a specific traditional playing technique on a steel-string acoustic or electric guitar. When playing the slide guitar, the musician wears a slide tube on the fretting hand. Instead of pressing the strings against the fretboard, she or he glides the tube on the strings while the picking hand plucks the strings in a regular fashion. This produces a unique, voice-like tone with stepless pitch control. Although the tube is usually slid along all six strings, single-note melodies can be played by plucking just one string and damping the others with the picking hand. The slide tube, usually made of glass or metal, also generates a squeaking sound while moving along on the wound metal strings. In most cases, the slide guitar is tuned into an open tuning (for example the open G tuning: D_2, G_2, D_3, G_3, B_3, and D_4 starting from the thickest string). This allows the user to play simple chords just by sliding the tube into different positions on the guitar neck. The player usually wears the slide tube on the pinky or ring finger, and the other fingers are free to fret the strings normally.

A virtual slide guitar (VSG) [7, 4] is described in this paper. The VSG consists of an infra-red (IR) camera, IR-reflecting slide tube and a ring, a computer running a physics-based string algorithm, and a loudspeaker. The VSG is played by wearing the slide tube on one hand and the ring on the other, and by making guitar-playing gestures in front of the camera. The user's gestures are mapped into synthesis control parameters, and the resulting sound is played back through the loudspeaker in real-time. More information on gestural control of music synthesis can be found e.g. in [8] and [16].

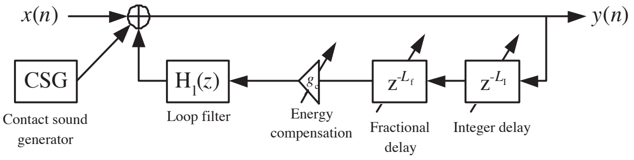

Figure 1: The signal flow diagram of the slide guitar string synthesizer. The energy compensation block compensates for the artificial energy losses due to the time-varying delays. The contact sound generator (see Figure 2) simulates the handling noise due to the sliding tube-string contact.

From the control point of view, the VSG can be seen as a successor of the virtual air guitar (VAG) [1] developed at Helsinki University of Technology a few years ago. The major difference between these gesture-controlled guitar synthesizers is that like in the real slide guitar, the VSG allows a continuous control over the pitch, and also sonifies the contact sounds emanating from the sliding contact between the slide tube and the imaginary string.

The VSG uses digital waveguides [11, 12] for synthesizing the strings. A model-based contact sound generator is added for simulating the friction-based sounds created by the sliding tube-string contact. More information on physics-based sound synthesis methods can be found in [14].

2. STRING SYNTHESIS

A single-delay loop (SDL) digital waveguide (DWG) model [2] with time-varying pitch forms the basis of the slide guitar synthesis engine, as illustrated in Fig. 1. The string model consists of a feedback delay loop with an additional loop filter, an energy scaling coefficient, and a contact sound generator block. The fractional delay filter in Fig. 1 allows for a smooth transition between pitches, and also enables the correct tuning of the string. There are several techniques for implementing fractional delay filters, a thorough tutorial being found in [3]. For the purpose of this work, a fifth-order Lagrange interpolator was found to work sufficiently well. It must be noted that both the integer delay line length and the fractional delay filter are time-varying, i.e. the user controls the total loop delay value and thus also the pitch during run-time.

NIME08, Genoa, Italy

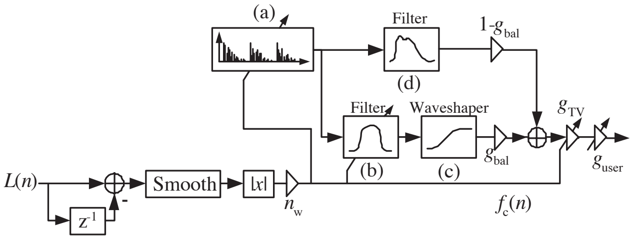

Figure 2: The contact sound generator block. The sliding velocity controlled by the user commands the synthetic contact noise characteristics. The sub-blocks are (a) the noise pulse generator, (b) a resonator creating the first harmonic of the time-varying noise structure, (c) a static nonlinearity generating the upper time-varying harmonics, and (d) an IIR filter simulating the general spectral characteristics of the noise.

The loop filter is a one-pole lowpass filter that simulates the vibrational losses of the string. Different filter parameters are used depending on the length and type of the string, as suggested in [15]. Also, when changing the length of a DWG string during run time, the signal energy is varied [5]. In practice, this can be heard as an unnaturally quick decay of the string sound. A time-varying scaling technique, introduced in [5], was used as a compensation. This results in an additional scaling operation inside the waveguide loop, as illustrated in Fig. 1.

2.1 Contact Sound Synthesis

The handling sounds created by the sliding tube-string contact are very similar to the handling sounds between a sliding finger-string contact. A recent study [6] revealed that these squeaky sounds consist mainly of lowpass-type noise with both static and time-varying harmonic components. The lowpass-cutoff frequency, frequencies of the time-varying harmonics, and the overall magnitude of the contact noise are controlled by the sliding velocity.

For synthesizing the handling sounds, we chose a noise pulse train as the excitation signal. This is based on the assumption that when the tube slides over a single winding, it generates a short, exponentially decaying noise burst. The time interval between the noise pulses is controlled by the sliding velocity; a fast slide results in a temporally dense pulse train, while a slow slide makes the pulses appear further apart. In fact, the contact sound synthesizer can be seen as a periodic impact sound synthesis model rather than a friction model.

The general structure of the contact noise generator block is illustrated in Fig. 2. The input variable $L(n)$ denotes the relative string length, controlled by the distance between the user's hands. Variable n is the time index. Since the contact noise depends on the sliding velocity, a time difference is taken from the input signal. If the control rate of the signal $L(n)$ is different from the sound synthesis sampling rate, as is often the case, a separate smoothing block is required after the differentiator. The smoothing block changes the sampling rate of $L(n)$ to be equal to the sound synthesis sampling rate and uses polynomial interpolation to smooth the control signal. Furthermore, since the contact noise is independent of the direction of the slide (up / down on the string), the absolute value of the control signal is taken. The scaling coefficient n_{w} denotes the number of windings on the string. The signal f_{c} after this scaling can therefore be seen as the noise pulse firing rate.

The basis of the synthetic contact sound for wound strings is produced in the noise pulse train generator (Fig. 2, block (a)). It outputs exponentially decaying noise pulses at the given firing rate. In addition, the type of the string determines the decay time and duration of an individual pulse. For enhancing the harmonic structure of the contact noise on wound strings, the lowest time-varying harmonic is emphasized by filtering the noise pulse train with a second-order resonator (block (b)), where the firing rate controls the resonators center frequency. The higher harmonics are produced by distorting the resonators output with a suitable nonlinear waveshaper (block (c)). A scaled hyperbolic tangent function is used for this. Hence, the number of higher harmonics can be controlled by changing the scaling of this nonlinear function.

A 4th-order IIR filter (block (d)) is used for simulating the static longitudinal string modes and the general spectral shape of the contact noise. As the noise characteristics depend on the tube material and string type, different filter parameters are used for different slide tube and string configurations. In Fig. 2, the scaling coefficient g_{bal} controls the ratio between the time-varying and static contact sound components. Finally, the total amplitude of the synthetic contact noise is controlled by the slide velocity $f_{\mathrm{c}}(n)$, via a scaling coefficient g_{TV}. Parameter g_{user} allows the user to control the overall volume of the contact sound. For plain, i.e. unwound strings, the contact sound synthesis block is simplified by replacing the noise burst generator (block (a) in Fig. 2) with a white noise generator, and by omitting blocks (b), (c), and (d).

3. REAL-TIME IMPLEMENTATION

Since the user controls the pitch of the VSG in a continuous manner, it is important that there is not a large latency between the user's action and the resulting sound. Thus, a high frame rate (120 fps) infra-red (IR) camera is used for detecting the users hand locations. The camera operates by lighting the target with IR-LEDs and sensing the reflected IR light. A real slide tube coated with IR reflecting fabric is used for detecting the users fretting hand. For recognition of the picking hand, a small ring of IR reflecting fabric is worn on the index finger.

3.1 Technical Details

The implementation works on a 2.66 GHz Intel Pentium 4 CPU with 1 GB of RAM and a SoundMax Integrated Digital Audio soundcard. Both the sound synthesis part and the camera interface operate in the Windows XP environment. The sound synthesis uses PD (Pure Data) [9] version 0.38.4-extended-RC8. The sampling frequency for the synthesis algorithm is 44.1 kHz, except for the string waveguide loop, which runs at 22.05 kHz, as suggested in [13]. A Naturalpoint TrackIR4:PRO USB IR-camera is used for gesture recognition. Its output is a 355 x 290 binary matrix, where the reflected areas are seen as blobs. As a side note, a recent article describing a PD patch for multichannel guitar effects processing can be found in [10].

3.2 Camera API

For the camera API (Application Programming Interface), Naturalpoint's OptiTrack SDK version 1.0.030 was used. The API was modified in the Visual Studio environment to include gesture-recognition features. The added features consist of the distinction between the two blobs (i.e. slide and plucking hand), calculation of the distance between them, recognition of the plucking and pull-off gestures, and transmission of the control data to PD as OSC (Open Sound Control) messages. Also, an algorithm was added to keep track of the virtual string location, i.e. an imaginary line representing the virtual string. This is very similar to the work presented in [1]. The line is drawn through the tube and the averaged location of the plucking hand, so that the virtual string slowly follows the players movements. This prevents the user from drifting away from the virtual string. The API detects the direction of the plucking hand movement, and when the virtual string is crossed, a pluck event and a direction parameter is sent. Also, a minimum velocity limit is defined for the plucking gesture in order to avoid false plucks.

3.3 PD Implementation

When the PD implementation receives an OSC message containing a pluck event, an excitation signal is inserted into each waveguide string. The excitation signal is a short noise burst simulating a string pluck. There is also a slight delay (20 ms) between different string excitations for creating a more realistic strumming feel. The order in which the strings are plucked depends on the plucking direction. Figure 3 illustrates the structure and signaling of the PD patch.

The camera software can be set to show the blob positions on screen in real time. This is not required for playing, but it helps the user to stay in the cameras view. The camera API uses roughly 10% of CPU power without the display and 20-40% with the display turned on. Since PD uses up to 80% of CPU power when playing all six strings, the current VSG implementation can run all six strings in real time without a noticeable drop in performance, provided that the blob tracking display is turned off. Selecting fewer strings, switching the contact sound synthesis off, or dropping the API frame rate to half, the display can be viewed while playing.

3.4 Virtual Slide Guitar

The virtual slide guitar system is illustrated in Fig. 4. The camera API recognizes the playing gestures and sends the plucking and pull-off events, as well as the distance between the hands, to the synthesis control block in PD. The synthesis block consists of the DWG models illustrated in Fig. 1. At its simplest, the VSG is easy to play and needs no calibration. The user simply puts the slide tube and reflecting ring on and starts to play. For more demanding users, the VSG provides extra options, such as altering the tuning of the instrument, selecting the slide tube material, setting the contact sound volume and balance between static and dynamic components, or selecting an output effect (a reverb or a guitar amplifier plugin).

The tube-string contact sound gives the user direct feedback of the slide tube movement, while the pitch of the string serves as a cue for the tube position. Thus, visual feedback is not needed in order to know where the slide tube is situated on the imaginary guitar neck.

4. CONCLUSIONS

This paper discussed a real-time virtual slide guitar synthesizer with camera-based gestural control. Time-varying digital waveguides with energy-compensation are used for simulating the string vibration. The contact noise between the strings and the slide tube is generated with a parametric model. The contact sound synthesizer consists of a noise pulse generator, whose output is fed into a time-varying resonator and a distorting nonlinearity. By controlling the noise pulse firing rate, the resonators center frequency, and the overall dynamics with the sliding velocity, a realistic time-varying harmonic structure is obtained in the resulting synthetic noise. The overall spectral shape of the contact noise is set with a 4th-order IIR filter.

The slide guitar synthesizer is operated using an optical gesture recognition user interface, similarly as suggested in [1]. However, instead of a web-camera, a high-speed infrared video camera is used for attaining a lower latency between the users gesture and the resulting sound. This IR-based camera system could also be used for gestural control of other latency-critical real-time applications. The real-time virtual slide guitar model has been realized in PD. A video file showing the virtual slide guitar in action can be found on the Internet: `http://youtube.com/watch?v=eCPFYKq5zTk`.

5. ACKNOWLEDGMENTS

This work has been supported by the GETA graduate school, the Cost287-ConGAS action, EU FP7 SAME project, and the Emil Aaltonen Foundation.

6. REFERENCES

[1] M. Karjalainen, T. Mäki-Patola, A. Kanerva, and A. Huovilainen. Virtual air guitar. *J. Audio Eng. Soc.*, 54(10):964–980, Oct. 2006.

[2] M. Karjalainen, V. Välimäki, and T. Tolonen. Plucked-string models: From the Karplus-Strong algorithm to digital waveguides and beyond. *Computer Music J.*, 22(3):17–32, 1998.

[3] T. I. Laakso, V. Välimäki, M. Karjalainen, and U. K. Laine. Splitting the unit delay - tools for fractional delay filter design. *IEEE Signal Proc. Mag.*, 13(1):30–60, 1996.

[4] J. Pakarinen. *Modeling of Nonlinear and Time-Varying Phenomena in the Guitar.* PhD thesis, Helsinki University of Technology, 2008. Available on-line at `http://lib.tkk.fi/Diss/2008/isbn9789512292431/` (checked Apr. 14, 2008).

[5] J. Pakarinen, M. Karjalainen, V. Välimäki, and S. Bilbao. Energy behavior in time-varying fractional delay filters for physical modeling of musical instruments. In *Proc. Intl. Conf. on Acoustics, Speech, and Signal Proc.*, volume 3, pages 1–4, Philadelphia, PA, USA, Mar. 19-23 2005.

[6] J. Pakarinen, H. Penttinen, and B. Bank. Analysis of handling noises on wound string. *J. Acoust. Soc. Am.*, 122(6):EL197–EL202, Dec. 2007.

[7] J. Pakarinen, T. Puputti, and V. Välimäki. Virtual slide guitar. *Computer Music J.*, 32(3), 2008. Accepted for publication.

[8] J. Paradiso and N. Gershenfeld. Musical Applications of Electric Field Sensing. *Computer Music J.*, 21(2), 1997.

[9] M. Puckette. Pure data. In *Proc. Intl. Computer Music Conf.*, pages 269–272, 1996.

[10] M. Puckette. Patch for guitar. In *Proc. PureData Convention 07*, Aug. 21-26 2007. Available on-line at `http://artengine.ca/~catalogue-pd/19-Puckette.pdf` (checked Apr. 9, 2008).

[11] J. O. Smith. Physical modeling using digital waveguides. *Computer Music J.*, 16(4):74–87, Winter 1992.

[12] J. O. Smith. *Physical Audio Signal Processing.* 2004. Aug. 2004 Draft, http://ccrma.stanford.edu/~jos/pasp/.

[13] V. Välimäki, J. Huopaniemi, M. Karjalainen, and Z. Janosy. Physical modeling of plucked string

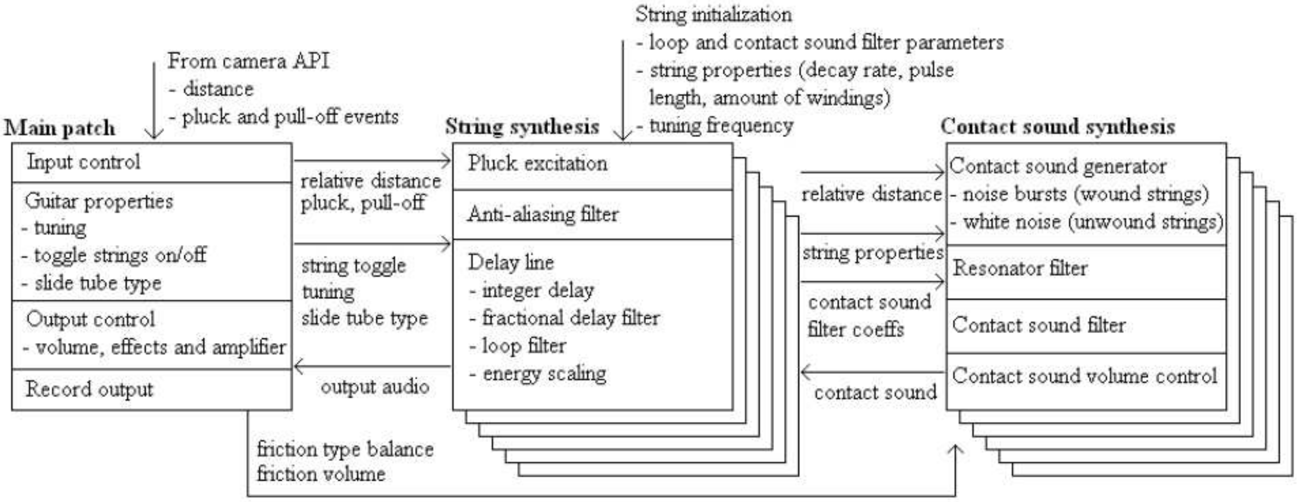

Figure 3: Structure and signaling of the PD patch.

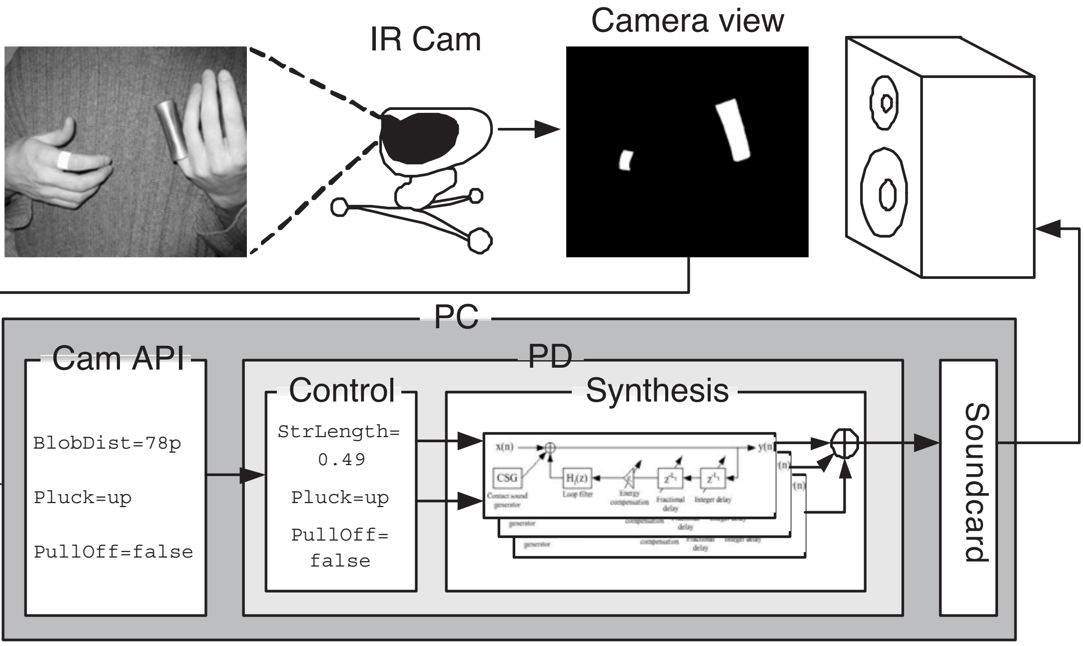

Figure 4: The complete components of the virtual slide guitar.

instruments with application to real-time sound synthesis. *J. Audio Eng. Soc.*, 44(5):331–353, 1996.

[14] V. Välimäki, J. Pakarinen, C. Erkut, and M. Karjalainen. Discrete-time modelling of musical instruments. *Reports on Progress in Physics*, 69(1):1–78, Jan. 2006.

[15] V. Välimäki and T. Tolonen. Development and calibration of a guitar synthesizer. *J. Audio Eng. Soc.*, 46(9):766–778, 1998.

[16] M. Wanderley and P. Depalle. Gestural control of sound synthesis. *Proc. IEEE*, 92(4):632–644, 2004.