Real-time Control of Advanced Physical Models using the Sensel Morph

Silvin Willemsen, Nikolaj Andersson and Stefania Serafin
Multisensory Experience Lab,
Aalborg University Copenhagen
Copenhagen, Denmark
{sil, nsa, sts}@create.aau.dk

ABSTRACT

Lorum Ipsum

1. INTRODUCTION

The behaviour of musical instruments can be well defined by partial differential equations (PDEs) [**?**].

Finite-difference schemes (FDSs)

The physical models (PMs) used as a case study in this project are the stiff string and the plate.

On top of all this, we have used the expressive Sensel Morph [**?**] control surface to control the PMs in real-time, something that to the best of the authors' knowledge has not been done before.

This paper is structured as follows: Section **??** describes the PMs used in the implementation and Section **??** shows the FDSs used to digitally implement these models. Furthermore, Section **??** will show how to implement the FDSs, Section **??** shows several different configurations of PMs inspired by real musical instruments and finally Section **??** and Section **??** will discuss and conclude upon the work shown in this paper.

2. MODELS

In this section, the partial differential equations for the damped stiff string and the plate will be presented.

2.1 Stiff string

The state $u = u(x,t)$ describes the transverse displacement of the string. The partial differential equation for the damped stiff string is defined as [**?**]

$$u_{tt} = \gamma^2 u_{xx} - \kappa^2 u_{xxxx} - 2\sigma_0 u_t + 2\sigma_1 u_{txx}, \quad (1)$$

where γ is wave-speed with units of frequency [s^{-1}], κ is a stiffness parameter and $\sigma_0 \geq 0$ and $\sigma_1 \geq 0$ are frequency-dependent and frequency-independent damping respectively. The subscript for u denotes a single derivative with respect to time t or space x respectively.

We can extend Equation (**??**) to a bowed string [**?**]

$$u_{tt} = ... - \delta(x - x_B) F_B \phi(v_{rel}), \quad \text{with} \quad (2)$$
$$v_{rel} = u_t(x_B) - v_B \quad (3)$$

where $F_B = f_B / M_s$ is the excitation function [m/s^2] with bowing force f_B [N] and total string mass M_s [kg]. The relative velocity v_{rel} is defined as the difference between the velocity of the string at bowing point x_B and the bowing velocity v_B [m/s] and ϕ is a friction characteristic, which has been chosen to be [**?**]

$$\phi(v_{rel}) = \sqrt{2a} v_{rel} e^{-a v_{rel}^2 + 1/2}. \quad (4)$$

Furthermore,

$$\delta(x - x_B) = \begin{cases} 1, & \text{if } x = x_B \\ 0, & \text{otherwise,} \end{cases} \quad (5)$$

is referred to as the spatial Dirac delta function, which, when multiplied onto the excitation, applies it only to the point x_B on the string.

2.2 Plate

In the case of a plate, the state $u = u(x,y,t)$ is now defined over two spatial dimensions. The PDE for a damped plate is [**?**]

$$u_{tt} = -\kappa^2 \Delta\Delta u - 2\sigma_0 u_t + 2\sigma_1 \Delta u_t, \quad (6)$$

where κ is again a stiffness parameter. Furthermore, Δ represents the 2D Laplacian.

2.3 Connections

Adding connections between different PMs, further referred to as elements, adds another term to Equation (**??**) or (**??**)

$$u_{tt} = ... + F_\alpha E_\alpha, \quad (7)$$
$$u_{tt} = ... + F_\beta E_\beta, \quad (8)$$

where F_α and F_β are the forces of the connection at connection areas E_α and E_β respectively. If a connection area consists of only one point, E reduces to $\delta(x - x_c)$ where x_c is the point of connection. We use the implementation as presented in [**?**] where the connection between two elements is a non-linear spring. The forces it imposes on the elements it connects - denoted by α and β - are defined as

$$F_\alpha = -\omega_0^2 \eta - \omega_1^4 \eta^3 - 2\sigma_\times \eta_t, \quad (9)$$
$$F_\beta = -M_{\alpha/\beta} F_\alpha, \quad (10)$$

where ω_0 and ω_1 are the linear and non-linear spring coefficients respectively, $\sigma_\times$ is the damping factor, $M_{\alpha/\beta}$ is the mass ratio between the two elements and η is the relative displacement between the connected elements at the point of connection. The subscript t again denotes a derivative with respect to time.

3. FINITE-DIFFERENCE SCHEMES

To be able to digitally implement the continuous equations shown in the previous section, they need to be approximated. The models can be discretised at times $t = nk$, where $n \in \mathbb{N}$ and $k = 1/f_\text{s}$ is the time-step with sample-rate f_s and locations $x = lh$, where $l \in [0, N]$ with N being the total number of points and h is the grid-spacing of the model which is calculated differently for each model (see sub-sections below). The discretised variable u_l^n is $u(x,t)$ at the nth time step and the lth point on the string. The plate is discretised using $x = lh$ where $l \in [0, N_x]$ with N_x being the total horizontal number of points and $y = mh$ where $m \in [0, N_y]$ with N_y being the total vertical number of points. Approximations for the derivatives in the equations found in Section **??** can be found in [**?**].

3.1 Stiff String

Equation (**??**) can be approximated using

$$\begin{aligned}\delta_{tt}u_l^n =&\gamma^2\delta_{xx}u_l^n - \kappa^2\delta_{xx}\delta_{xx}u_l^n - 2\sigma_0\delta_{t\cdot}u_l^n \\ &+ 2\sigma_1\delta_{t-}\delta_{xx}u_l^n - \delta_{x_\text{B}}F_\text{B}\phi(v_\text{rel}),\end{aligned} \tag{11}$$

where $\delta_{x_\text{B}} = \delta(x - x_\text{B})$ (see Equation (**??**)) and

$$v_\text{rel} = \delta_{t\cdot}u(x_\text{B}) - v_\text{B}. \tag{12}$$

For our implementation, clamped boundary conditions were used, defined as:

$$u = u_x = 0 \quad \text{where} \quad l = \{0, N\}. \tag{13}$$

For stability reasons, the grid-spacing needs to abide the following condition

$$h \geq \sqrt{\frac{\gamma^2k^2 + 4\sigma_1k + \sqrt{(\gamma^2k^2 + 4\sigma_1k)^2 + 16\kappa^2k^2}}{2}}. \tag{14}$$

The closer h is to this limit, the higher the quality of the implementation. The number of points N can then be calculated using

$$N = h^{-1}. \tag{15}$$

3.2 Plate

Equation (**??**) can be approximated using

$$\begin{aligned}\delta_{tt}u_{l,m}^n =& -\kappa^2\delta_\Delta\delta_\Delta u_{l,m}^n - 2\sigma_0\delta_{t\cdot}u_{l,m}^n \\ &+ 2\sigma_1\delta_{t-}\delta_\Delta u_{l,m}^n\end{aligned} \tag{16}$$

As in the case of the stiff string, we chose to use clamped boundary conditions:

$$\begin{aligned}u = u_x = 0 \quad &\text{where} \quad l = \{0, N_x\} \quad \text{and} \\ u = u_y = 0 \quad &\text{where} \quad m = \{0, N_y\}\end{aligned} \tag{17}$$

3.3 Connections

$$F_\alpha = -\omega_0^2\mu_{t\cdot}\eta - \omega_1^4\eta^2\mu_{t\cdot}\eta - 2\sigma_\times\delta_{t\cdot}\eta, \tag{18}$$

$$F_\beta = -M_{\alpha/\beta}F_\alpha, \tag{19}$$

The relative displacement η between α and β can be calculated as

$$\eta^n = h_\alpha u_{\alpha,x_\alpha}^n - h_\beta u_{\beta,x_\beta}^n, \tag{20}$$

which, in other words, is the difference between the state of element α at connection point x_α and the state of element β at connection point x_β scaled by their respective grid-spacings h_α and h_β.

4. IMPLEMENTATION

In this section, we will present how to implement the FDSs presented in Section **??** and elaborate more on the parameters used. The values for most parameters have been arbitrarily chosen and can - as long as they satisfy the conditions - be changed. We used C++ along with the JUCE framework for implementing the PMs and connections in real-time. The main hardware used for testing was a MacBook Pro with a 2.2 GHz Intel Core i7 processor.

Note: In this paper we have used the simple case of a single point for bowing, excitation and connections. These can be extended to a bowing area, excitation area and area of connection. For more information on this, we would like to refer the reader to [?].

The most important thing is to optimise the FDSs as these will be ran at sampling rate. We make sure to only have one multiplication per location of u per time-step (u_l^n, u_{l+1}^{n-1} etc) (**SEE APPENDIX DAFX PAPER**)

4.1 String

The wave-speed of the string is proportional to the fundamental frequency of the stiff string according to

$$\gamma = 2f_0. \tag{21}$$

This can be altered in real-time to control the pitch. The stiffness parameters have been chosen to be $\sigma_0 = 0.1$ and $\sigma_1 = 0.005$.

The stiffness can be calculated using

$$\kappa = \frac{\sqrt{B}\gamma}{\pi}, \tag{22}$$

where $B = 0.001$ is the inharmonicity coefficient [m^{-2}].

The output is retrieved at an arbitrary point l.

As can be seen from Equation (**??**) the solution for v_rel is implicit. It is thus necessary to use an iterative root-finding method such as Newton-Raphson [source].

4.2 Plate

We chose the number of points to be $N_x = 20$ and $N_y = 10$ as this was found to be a great speed/quality tradeoff.

In the case of the plate, we set the number of horizontal and vertical points and calculate grid spacing h from that using

$$h = \frac{\sqrt{N_x/N_y}}{N_x}. \tag{23}$$

4.3 Connections

4.4 Tuning

See [**?**] section 7.6.

4.5 Damping finger

Another way to generate different pitches is to add damping to the model at several points that acts as a fretting finger. If the string is fretted st single location $x_\mathrm{f} \in [0, 1]$ and $l_\mathrm{f} = \mathrm{floor}(x_\mathrm{f}/h)$ we use

$$u_l^n = \begin{cases} 0, & \text{if} \quad l = l_\mathrm{f} - 1 \vee l = l_\mathrm{f} \\ (1 - \alpha_\mathrm{f}^\epsilon) u_l^n & \text{if} \quad l = l_\mathrm{f} \end{cases} \tag{24}$$

where $\alpha_\mathrm{f} = x_\mathrm{f}/h - l_\mathrm{f}$ describes the fractional location of x_f between two grid points. We found that the effect of damping between grid points does not linearly scale to pitch. We thus added $\epsilon = 7$ as a heuristic value to more properly map finger position to pitch.

4.6 Sensel Morph

The Sensel Morph (or further referred to as Sensel) is an expressive touch controller

4.6.1 Mapping strategies

Something about the different prototype mappings, and the "final" mapping

5. INSTRUMENTS

In this section, several configurations of strings, plates and connections that are inspired by real-life instruments will be presented. We subdivide string-elements into three types: bowed, plucked and sympathetic strings. All strings will be connected to one plate which simulates an instrument body. The user can control the plate stiffness and increase it to generate more interaction between the strings. Furthermore, the user can change the output-level of each element-type using. Apart from these parameters, which are controlled by the mouse, the instruments are fully controlled by two Sensels. The instruments we have chosen as our inspiration are the sitar, the hammered dulcimer and the hurdy gurdy. Videos and sound examples can be found through the following link:...

5.1 Bowed Sitar

The sitar is an originally Indian string instrument that has both fretted strings and sympathetic strings. Instead of picking the fretted strings, we extended the model to bow these strings. Our implementation consists of 2 bowed strings, 13 sympathetic strings and 5 plucked strings (as it is also possible to strum the sympathetic strings). One Sensel is vertically subdivided into two sections, one for each bowed string, The first finger registered by the Sensel is mapped to the bow and the second is mapped to a fretting finger on the string controlling pitch. The frets are not implemented as such (the pitch is continuous), but they are visualised for reference. The horizontal position of the first finger is mapped to the bowing position on the string, the vertical velocity to the bow velocity with a maximum of $v_\mathrm{B} = 0.2$ m/s and the finger force is linked to the excitation function with a maximum of $F_\mathrm{B} = 100$ m/s^2. The other Sensel is subdivided into 5 sections mapped to the plucked strings.

5.2 Hammered Dulcimer

The hammered dulcimer is an instrument that can be seen as an 'open piano' where the musician has the hammers in their hand. Just like the piano, the strings are grouped in pairs or triples that are played simultaneously. In our implementation, we have 20 pairs of plucked strings. The excitation (which are plucked strings with a longer excitation length) One of each pair is connected to the plate which slightly detunes it

5.3 Hurdy Gurdy

The hurdy gurdy is an instrument that consists of bowed and sympathetic strings. The bowing happens through a rosined wheel attached to a crank and bows these strings as the crank is turned. It is possible to change the pitch of a few bowed strings - the melody strings - using buttons that press tangent pins on the strings at different positions. The other strings, referred to as drone strings, are mostly tuned lower than the melody strings and provide the base notes of the instrument. The musician can place the bowed strings on rests that keep the wheel from interacting with it.

Our implementation consists of 5 bowed strings subdivided into 2 drone strings tuned to A2, E3 and 3 melody strings tuned to A3, E4 and A4 and 13 sympathetic strings.

The Sensel is vertically subdivided into 5 rows that control whether the strings are placed on the wheel. The bowing velocity is mapped to the average pressure of the fingers. The pitch (in the model this is the wave-speed γ) of the melody-strings are changed by a midi controller.

6. DISCUSSION

Body should not be plate.

7. CONCLUSION AND FUTURE WORK

Speed up more: AVX vector / GPU / multithreading such as [**?**]

Model instrument bodies

Acknowledgments

We would like to thank...

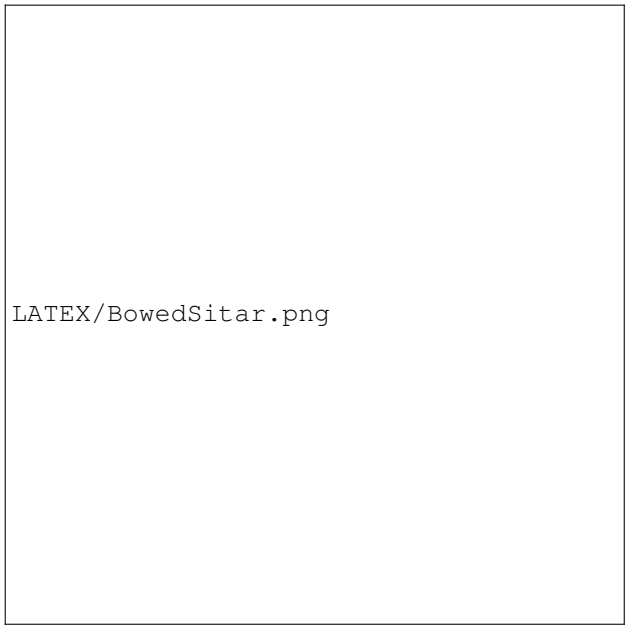

Figure 1. The bowed sitar application.