Modular Physical Models in a Real-Time Interactive Application

Silvin Willemsen, Titas Lasickas, and Stefania Serafin
Multisensory Experience Lab, CREATE,
Aalborg University Copenhagen, Denmark
sil@create.aau.dk

ABSTRACT

Through recent advances in processing power, physical modelling using finite-difference time-domain (FDTD) methods has gained popularity. Many different musical instrument models based on these methods exist, and nearly all are based on the same underlying systems and interactions between them. This paper presents an application where individual resonator modules, such as strings, bars, membranes and plates, can be connected and interacted with in real time. Various excitations, including the bow, hammer and pluck, are implemented as well, allowing for expressive control and a wide sonic palette. Existing and non-existing model configurations can easily be implemented, modified and experimented with, as well as the parameters describing them.

1. INTRODUCTION

Nearly any musical instrument can be subdivided into a resonator and exciter component [1]. Examples of resonator-exciter combinations are the violin and the bow, or the guitar and the pick. Many resonators, such as those mentioned here, can be further subdivided into more basic resonators, i.e., a set of individual strings, a bridge and a wooden body. As many instruments consist of the same basic resonators – simply with different geometries or made from different materials – one can imagine some application that can implement many musical instruments based on the same fundamental resonator components in a modular fashion. Using physical modelling to implement these resonators allows for accurate implementation of the interactions between them.

Modularity in physical modelling sound synthesis is by no means a new concept. The earliest example of a modular system for sound synthesis was due to Cadoz *et al.* [2], where their CORDIS system allowed complex instruments to be created using simple mass-spring systems. Later, Morrison and Adrien created Mosaic [3], a modular environment using modal synthesis [4]. Rabenstein *et al.* presented presented modular physical models with a block-based approach using wave digital filters [5] and digital waveguides [6].

Finite-difference time-domain (FDTD) methods, first used in a musical context by Ruiz [7], Hiller and Ruiz [8, 9] and later by Chaigne [10], also lend themselves to a modularity. In [11], Bilbao presents a modular environment where bars and plates are connected by nonlinear springs, and in [12] Bilbao *et al.* propose a modular environment including higher dimensional systems.

Due to a recent increase in computational power, FDTD methods have gained popularity in real-time applications [13]. A real-time modular environment using strings and lumped objects is presented in [14], and a real-time implementation of connected strings and bars using the FAUST programming language due to Südholt *et al.* in [15].

The main goal of the aforementioned literature on FDTD-based modular environments is to create arbitrary sound-generators based on physical equations of motion, rather than modelling existing instruments. Furthermore, the modular environments able to run in real-time only include systems spatially distributed over a maximum of one dimension. Many instruments require higher-dimensional structures to more accurately reproduce their sound. As done in e.g. [16, 17], the body of stringed instruments can be simplified to a thin plate, which is a system distributed over two dimensions.

This work presents a interactive modular environment in a real-time application where FDTD implementations of strings, bars, membranes and plates can be connected and played by the user. Although it is still possible to create arbitrary configurations to build non-existing instruments, the focus of this contribution is to allow for the possibility of modelling existing instruments. The current work aims to generalise previous work done on musical instruments modelled using FDTD methods (see e.g. [16–20]), such that these can all be easily implemented as 'presets' of a modular application. As many of these instruments consist of the same components, one can avoid from-the-ground-up implementation in the future. This work is part of a larger project, which will see its fruition in virtual reality (VR), where it will be used as a sound engine.

The rest of this paper is structured as follows: Section 2 describes the physical models used in this work, including resonators, exciters and connections between resonators and Section 3 describes how to implement these. Section 4 presents the functionality of the real-time application, Section 5 presents results and discusses these, and concluding remarks appear in Section 6.

2. MODELS

This section describes the continuous-time equations of the systems used in the application. The transverse displacement of a system can be described by state variable $q(\boldsymbol{x},t)$ with time $t \geq 0$ and spatial coordinate $\boldsymbol{x} \in \mathcal{D}$. Here, the dimensions and definition of domain $\mathcal{D}$ depend on the system at hand. The dynamics of a system can then be written using the following general form:

$$\mathcal{L}q = 0, \tag{1}$$

where linear partial differential operator $\mathcal{L}$ describes the dynamics of a model in isolation.

This section presents the partial differential equations (PDEs) of the stiff string and the stiff membrane. These can be used as a general equation for modelling other 1D and 2D systems respectively through a special choice of parameters.

2.1 1D Systems

A commonly used 1D model (see e.g. [14,16]) is the damped stiff string (or string for short). With reference to Eq. (1), consider a string of length L (in m), its transverse displacement described by state variable $q = u(\chi, t)$ (in m), and spatial coordinate χ is defined over domain $\mathcal{D} = [0, L]$. Furthermore, $\mathcal{L} = \mathcal{L}^{(1)}$ is defined as [21]

$$\mathcal{L}^{(1)} = \rho A \partial_t^2 - T\partial_\chi^2 + EI\partial_\chi^4 + 2\sigma_0 \rho A \partial_t - 2\sigma_1 \rho A \partial_t \partial_\chi^2, \tag{2}$$

where ∂_t and ∂_χ denote partial differentiation with respect to time and space respectively. The model is parameterised by material density ρ (in kg/m^3), cross-sectional area $A = \pi r^2$ (in m^2), radius r (in m), tension T (in N), Young's modulus E (in Pa), area moment of inertia $I = \pi r^4/4$ and loss coefficients σ_0 (in s^{-1}) and σ_1 (in m^2/s). If $T = 0$, the model reduces to a damped bar. Note that a circular cross-section is assumed here.

In this work, the boundary conditions are chosen to be simply supported as

$$u = \partial_\chi^2 u = 0, \quad \text{for,} \quad \chi = 0, L. \tag{3}$$

2.2 2D Systems

An often-used 2D system is the thin plate (see e.g. [17,22]). However, to retain generality, and the possibility to model membranes a well, the PDE for a stiff membrane will be presented here.

Consider a rectangular stiff membrane (or membrane for short) with side lengths L_x and L_y (both in m). With reference to Eq. (1) its transverse displacement can be described by $q = w(x, y, t)$ (in m), which is defined for domain $\mathcal{D} = [0, L_x] \times [0, L_y]$, and $\mathcal{L} = \mathcal{L}^{(2)}$ is defined as [23]

$$\mathcal{L}^{(2)} = \rho H \partial_t^2 - T\Delta + D\Delta\Delta + 2\sigma_0 \rho H \partial_t - 2\sigma_1 \rho H \partial_t \Delta, \tag{4}$$

Here, $\Delta = \partial_x^2 + \partial_y^2$ is the Laplacian, and parameters are material density ρ (in kg/m^3), thickness H (in m), tension per unit length T (in N/m), stiffness coefficent $D = EH^3/12(1-\nu^2)$ (in kg $\cdot$ m$^2\cdot$s^{-2}), Young's modulus E (in Pa), dimensionless Poisson's ratio ν, and loss coefficients σ_0 (in s^{-1}) and σ_1 (in m^2/s). If $T = 0$, the model reduces to a thin plate, and if $D = 0$ it reduces to the 2D wave equation, which can be used to model a non-stiff membrane.

For simplicity, boundary conditions are chosen to be clamped, such that

$$w = \mathbf{n} \cdot \nabla w = 0, \tag{5}$$

where ∇ denotes 'the gradient of', and $\mathbf{n}$ is a normal to the plate area at the boundary.

2.3 Connections

One can add connections to the models presented above by extending the general form in Eq. (1). Consider M models q_m indexed by $m \in \mathfrak{M}$, where $\mathfrak{M} = \{1, \ldots, M\}$ and C connections between them. A connection is indexed by $c \in \mathfrak{C}$ with $\mathfrak{C} = \{1, \ldots, C\}$, and is characterised by the indices of the models it connects – $r_c \in \mathfrak{M}$ and $s_c \in \mathfrak{M}$ – and the locations where these models are connected – $\boldsymbol{x}_{r,c} \in \mathcal{D}_{r_c}$ and $\boldsymbol{x}_{s,c} \in \mathcal{D}_{s_c}$. Here, domains $\mathcal{D}_{r_c}$ and $\mathcal{D}_{s_c}$ are the (spatial) domains that models q_{r_c} and q_{s_c} are defined for.

Model q_{r_c} will be placed 'below' q_{s_c} such that the connection force acts positively on the former and negatively on the latter. The general form in Eq. (1) can then be extended to include connections according to

$$\mathcal{L}_m q_m = \sum_{\substack{c\in\mathfrak{C}\\ r_c = m}} \delta(\boldsymbol{x}_m - \boldsymbol{x}_{r,c}) f_c - \sum_{\substack{c\in\mathfrak{C}\\ s_c = m}} \delta(\boldsymbol{x}_m - \boldsymbol{x}_{s,c}) f_c, \tag{6}$$

where $f_c = f_c(t)$ is the force (in N) of the c^{th} connection.

The simplest connection considered in this work is the rigid connection, which assumes that the connected systems have an identical displacement at their respective connection locations. Alternatively, as done in [11, 24], one can use a spring to connect two systems. A connection force due to a nonlinear damped spring is

$$f_{\text{c}} = K_{1,c}\eta_c + K_{3,c}\eta_c^3 + R_c \partial_t \eta_c, \tag{7}$$

with is a linear spring coefficient $K_{1,c}$ (in N/m), nonlinear spring coefficient $K_{3,c}$ (in N/m^3), and damping coefficient R_c (in s^{-1}) of the c^{th} connection. If $K_{3,c} = 0$, Eq. (7) reduces to a linear spring. Furthermore, $\eta_c = \eta_c(t) = q_{s_c}(\boldsymbol{x}_{s,c}, t) - q_{r_c}(\boldsymbol{x}_{r,c}, t)$ is the relative displacement of the two systems at their respective connection locations (in m). Section 3.2 will elaborate on how to calculate the connection forces in all cases.

To illustrate, consider two models ($M = 2$), a string and a membrane, such that $q_1 = u(\chi, t)$ and $q_2 = w(x, y, t)$, and a single connection between them ($C = 1$) at unspecified locations $\chi_{\text{c}} \in \mathcal{D}_{r_1}$ and $(x_{\text{c}}, y_{\text{c}}) \in \mathcal{D}_{s_1}$ respectively. See Figure 1. Placing the string below the membrane, we get that $r_1 = 1$ and $s_1 = 2$, i.e., the 'below' model of connection 1 has index 1, the 'above' model of connection 1 has index 2. Substituting the partial differential operators for the string and membrane found in Eqs. (2) and (4) respectively, Eq. (6) becomes, for the string and the membrane respectively

$$\mathcal{L}^{(1)}u = \delta(\chi - \chi_\mathrm{c})f_1, \tag{8a}$$

$$\mathcal{L}^{(2)}w = -\delta(x - x_\mathrm{c}, y - y_\mathrm{c})f_1. \tag{8b}$$

One can apply a rigid connection to Eq. (8a) according to [24]:

$$u(\chi_\mathrm{c}, t) = w(x_\mathrm{c}, y_\mathrm{c}, t). \tag{9}$$

If instead a spring connection is chosen:

$$f_1 = K_{1,1}\eta_1 + K_{3,1}\eta_1^3 + R_1\partial_t\eta_1 \tag{10}$$

with $\eta_1 = \eta_1(t) = w(x_c, y_c, t) - u(\chi_c, t)$.

2.4 Excitations

In this work, as excitations will only be applied to 1D systems, the state variable of the stiff string, i.e., $u(\chi, t)$, will be used for the presentation of the various excitations. For the string, the general form in Eq. (1) can thus be extended to (ignoring connections for now)

$$\mathcal{L}^{(1)}u = e_\mathrm{e}f_\mathrm{e}, \tag{11}$$

where $e_\mathrm{e} = e_\mathrm{e}(\chi)$ is an excitation distribution and $f_\mathrm{e} = f_\mathrm{e}(t)$ is the externally supplied excitation force (in N).

2.4.1 *The Bow*

As done in previous work, see e.g. [16], one can include a bowing interaction by introducing a static friction model. With reference to Eq. (11), the excitation force can be defined using the following friction model [24]

$$f_\mathrm{e} = -f_\mathrm{B}\sqrt{2a_\mathrm{B}}v_\mathrm{rel}e^{-a_\mathrm{B}v_\mathrm{rel}^2+1/2} \tag{12}$$

with externally supplied bow force $f_\mathrm{B} = f_\mathrm{B}(t)$ (in N), dimensionless free parameter a_B and

$$v_\mathrm{rel} = \partial_t u(\chi_\mathrm{B}, t) - v_\mathrm{B} \tag{13}$$

is the relative velocity (in m/s) between the string at externally supplied bowing location $\chi_\mathrm{B} = \chi_\mathrm{B}(t)$ (in m) and the externally supplied bow velocity $v_\mathrm{B} = v_\mathrm{B}(t)$ (in m/s).

Furthermore, the excitation distribution is set to be a single point along the string:

$$e_\mathrm{e} = \delta(\chi - \chi_\mathrm{B}). \tag{14}$$

2.4.2 *Hammer*

Another way of exciting a system is to use a hammer, which can be modelled as a simple mass-spring system with state variable $z = z(t)$. The interaction between the hammer and the string is then modelled as a collision, using the following collision potential [25]:

$$\phi(\eta_\mathrm{e}) = \frac{K_\mathrm{e}}{\alpha_\mathrm{e}+1}[\eta_\mathrm{e}]_+^{\alpha_\mathrm{e}+1}, \tag{15}$$

with collision stiffness $K_\mathrm{e} \geq 0$ (in N/m$^{\alpha_\mathrm{e}}$) and nonlinear collision coefficient $\alpha_\mathrm{e} \geq 1$. Furthermore, $[\cdot] = 0.5(\cdot + |\cdot|)$ describes the 'positive part of' and

$$\eta_\mathrm{e} = \eta_\mathrm{e}(t) = \theta\left(u(\chi_\mathrm{e}, t) - z(t)\right) \tag{16}$$

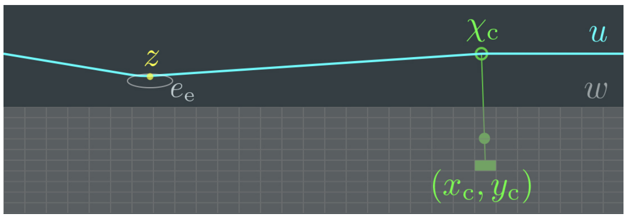

Figure 1. Snapshot of the application including a string and a thin plate with a connection between them. The string is excited using a pluck.

is the relative displacement between the string at collision location χ_e (in m) and the hammer (in m). Here, $\theta = \tau$ if the string should be excited from above, and $\theta = -\tau$ if it should be excited from below, where, $\tau = 1$ if the hammer interaction is triggered, and $\tau = 0$ if not.

Using a change of variables based on energy quadratisation presented in [26], one can rewrite the collision potential as

$$f_\mathrm{e} = -\theta\psi\psi' \tag{17}$$

where $\psi = \psi(\eta) = \sqrt{2\phi}$, and using dots to denote a temporal derivative, $\psi' = \dot{\psi}/\dot{\eta}$. This change of variables ultimately allows for an explicit implementation of the nonlinear collision.

Using the above, the dynamics of the hammer, including the collision with the string, can be described by the following PDE:

$$M_\mathrm{z}\partial_t^2 z = -K_\mathrm{z}\Big(z - (1-\tau)z_\mathrm{off}\Big) + \theta\psi\psi' \tag{18}$$

with mass M_z (in kg), spring constant K_z (in N/m) and $z_\mathrm{off} = z_\mathrm{off}(t)$ is an externally supplied offset (in m) (also see [17]). The offset is used to keep the mass away from the equilibrium (and thus the system it excites) when controlling the application without wanting to excite the system. If the hammer excitation is triggered, and thus $\tau = 1$ the offset will no longer have an effect on the mass and spring force will pull the hammer towards the system, colliding with it in the process. After the collision, $\tau = 0$, and the offset will affect the mass again to avoid continuous collision between the hammer and the syste.

Finally, with reference to Eq. (11), the excitation distribution is set to be a raised cosine with centre location χ_e and excitation width e_w (in m):

$$e_\mathrm{e}(\chi) = \begin{cases} \frac{1-\cos\left(\frac{2\pi(\chi-\chi_\mathrm{e})}{e_\mathrm{w}}+\pi\right)}{2} & \text{if } \chi_\mathrm{e} - \frac{e_\mathrm{w}}{2} \leq \chi \leq \chi_\mathrm{e} + \frac{e_\mathrm{w}}{2} \\ 0, & \text{otherwise.} \end{cases} \tag{19}$$

Note that χ_e and e_w must be chosen such that $\chi_\mathrm{e} - \frac{e_\mathrm{w}}{2} \in \mathcal{D}$ and $\chi_\mathrm{e} + \frac{e_\mathrm{w}}{2} \in \mathcal{D}$.

2.4.3 *Pluck*

The pluck is modelled nearly the same way as the hammer. The main difference is that $\tau = 1$ until the collision force is larger than a certain value, after which it will be set to 0. This will be elaborated on in Section 3.3.

3. DISCRETE TIME

In order to implement the models described in Section 2 using FDTD methods, a spatio-temporal grid needs to be defined.

For all models, time is discretised to $t = nk$ with time index $n = 0, 1, 2\dots$ and time step $k = 1/f_\text{s}$ (in s) where f_s is the sample rate (in Hz). For the stiff string, space is subdivided into N equal intervals of length h_s (in m) according to $\chi = ph$ with spatial index $p \in \{0, \dots, N\}$.

In the case of the stiff membrane, the spatial coordinate is discretised as $(x, y) = (lh, mh)$ where spatial indices $l \in \{0, \dots, N_x\}$ and $m \in \{0, \dots, N_y\}$. Here, N_x and N_y are the number of intervals in the x and y direction respectively. Notice that the same value for grid spacing h is used for both the x and y directions.

Using these definitions, the general state variable $q(\boldsymbol{x}, t)$ can be approximated to grid function $q^n_{\boldsymbol{l}}$, where for the stiff string $\boldsymbol{l} = p$ yielding grid function u^n_p and for the stiff membrane $\boldsymbol{l} = (l, m)$ yielding grid function $z^n_{(l,m)}$.

One of the main concerns when working with FDTD methods is the stability of the scheme. The stability condition for the discrete models can be described in terms of the grid spacing h as [13]

$$h \geq \sqrt{\beta\left(c^2k^2 + 4\sigma_1 k + \sqrt{(c^2k^2 + 4\sigma_1 k)^2 + 16\kappa^2k^2}\right)} \tag{20}$$

where for the string $c^2 = T/\rho A$, $\kappa^2 = EI/\rho A$ and $\beta = 0.5$, and for the membrane $c^2 = T/\rho H$, $\kappa^2 = D/\rho H$ and $\beta = 1$.

3.1 FDTD schemes

The general form in Eq. (1) can be discretised to the following FDTD scheme:

$$\ell q^n_{\boldsymbol{l}} = 0 \tag{21}$$

where ℓ is the discretised version of linear partial differential operator $\mathcal{L}$. The discrete-time definitions of Eqs. (2) and (4) will not be given here for brevity, but can be found in the literature (e.g. [24], [13]). However, for any FDTD scheme, and any definition of ℓ, one can expand (21) to yield an update equation of the form

$$aq^{n+1}_{\boldsymbol{l}} = bq^n_{\boldsymbol{l}} + cq^{n-1}_{\boldsymbol{l}} \tag{22}$$

where a, b and c depend on the system at hand. In the following, we assume that $a = (1 + \sigma_0 k)$ for for any discretised system.

3.2 Connections

To apply the effect of connections to FDTD schemes, interpolation and spreading operators must be introduced. As in Section 2.3, consider M models where the grid function of the mth model is $q^n_{m,\boldsymbol{l}_m}$. One can define (zeroth-order) interpolation operator

$$I_{m,\boldsymbol{l}_m}(\boldsymbol{x}_m) = \begin{cases} 1 & \text{if } \boldsymbol{l}_m = \lfloor \boldsymbol{x}_m/h_m \rfloor \\ 0, & \text{otherwise} \end{cases} \tag{23}$$

and can be applied to grid function $q^n_{m,\boldsymbol{l}_m}$, to to obtain the state of one grid point of the system. A spreading operator can then be defined as

$$J_{m,\boldsymbol{l}_m}(\boldsymbol{x}_m) = \frac{1}{(h_m)^\varepsilon} I_{m,\boldsymbol{l}_m}(\boldsymbol{x}_m), \tag{24}$$

where ε is the number of spatial dimensions the system is defined over ($\varepsilon = 1$ for 1D systems, $\varepsilon = 2$ for 2D systems). Equation (24) can then be used to localise a force onto a specific location along a system. The general form in Eq. (6) can be discretised to

$$\ell_m q^n_{m,\boldsymbol{l}_m} = \sum_{\substack{c\in\mathfrak{C}\\ r_c = m}} J_{m,\boldsymbol{l}_m}(\boldsymbol{x}_{r,c}) f^n_c - \sum_{\substack{c\in\mathfrak{C}\\ s_c = m}} J_{m,\boldsymbol{l}_m}(\boldsymbol{x}_{s,c}) f^n_c, \tag{25}$$

Assuming that none of the connections overlap [1] one can then explicitly solve for the forces f^n_c. One can express the relative distance between two models (q_{s_c} and q_{r_c}) connected by connection c as

$$\eta^n_c = I_{s_c,\boldsymbol{l}_{s_c}}(\boldsymbol{x}_{s,c}) q^n_{s_c,\boldsymbol{l}_{s_c}} - I_{r_c,\boldsymbol{l}_{r_c}}(\boldsymbol{x}_{r,c}) q^n_{r_c,\boldsymbol{l}_{r_c}}. \tag{26}$$

If a rigid connection is chosen, the force of connection c can be solved for according to

$$f^n_c = \frac{\eta^\star_c}{\frac{k^2}{a_{r_c}\mathcal{M}_{r_c}} + \frac{k^2}{a_{s_c}\mathcal{M}_{s_c}}}, \tag{27}$$

where $\mathcal{M}_m$ is the effective mass of a single grid point (in kg) of model q_m, i.e., $\mathcal{M} = \rho A h$ for 1D systems, and $\mathcal{M} = \rho H h^2$ for 2D systems. Furthermore,

$$q^\star_{m,\boldsymbol{l}_m} = \frac{b_m q^n_{m,\boldsymbol{l}_m} + c_m q^{n-1}_{m,\boldsymbol{l}_m}}{a_m} \tag{28}$$

is the state of model q_m at the next time step in isolation, i.e., without the connection forces (obtained by rewriting Eq. (22)), and can be appropriately used in Eq. (26) to obtain $\eta^\star_c$.

Introducing the centred difference and centred averaging operator

$$\delta_{t\cdot}\eta^n = \frac{1}{2k}\left(\eta^{n+1} - \eta^{n-1}\right) \approxeq \eta, \tag{29}$$

$$\mu_{t\cdot}\eta^n = \frac{1}{2}\left(\eta^{n+1} + \eta^{n-1}\right) \approxeq \eta, \tag{30}$$

the spring in Eq. (7) can be discretised to

$$f^n_\text{c} = K_{1,\text{c}}\mu_{t\cdot}\eta^n_\text{c} + K_{3,\text{c}}(\eta^n_\text{c})^2\mu_{t\cdot}\eta^n_\text{c} + R_\text{c}\delta_{t\cdot}\eta^n_\text{c}, \tag{31}$$

which can be shown to be inherently stable [11, 24]. The connection force can then be solved for according to

$$f^n_c = \frac{\eta^\star_c + \frac{\gamma_{c,-}}{\gamma_{c,+}}\eta^{n-1}_c}{\frac{1}{\gamma_{c,+}} + \frac{k^2}{a_{r_c}\mathcal{M}_{r_c}} + \frac{k^2}{a_{s_c}\mathcal{M}_{s_c}}}, \tag{32}$$

where

$$\gamma_{c,\pm} = \frac{K_{1,c}}{2} + \frac{K_{3,c}(\eta^n_c)^2}{2} \pm \frac{R_c}{2k}. \tag{33}$$

Notice that the rigid connection in Eq. (27) is a special case of Eq. (32) where $\eta^{n-1}_c = 0$ and $\gamma_{c,+} \to \infty$. See [13, Ch. 11] for a derivation.

[1] Overlapping connections are possible [11] and will be briefly discussed in Section 5.

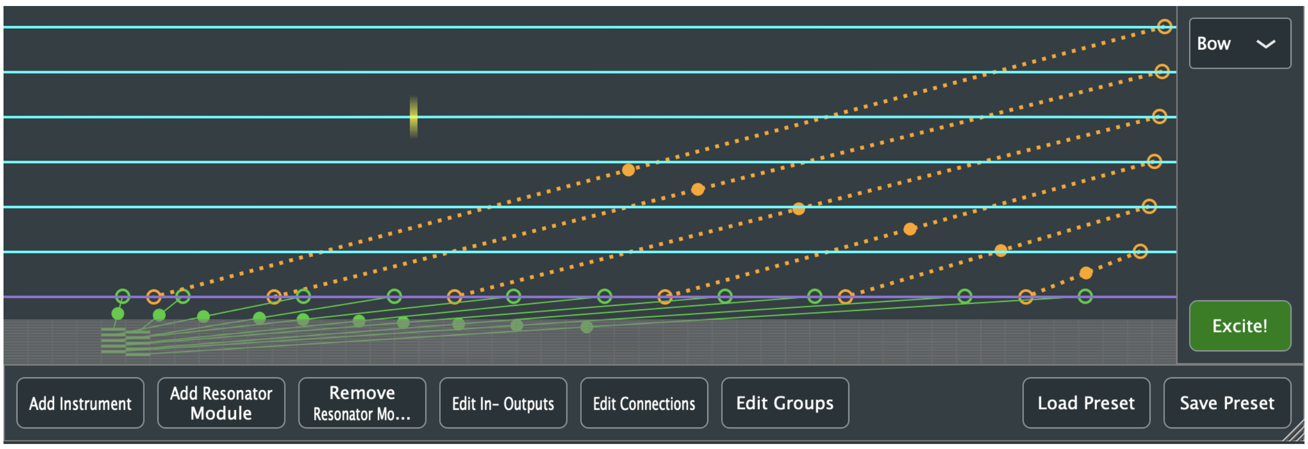

Figure 2. The graphical user interface (GUI) of the application. Here the guitar preset is loaded with 6 strings, 1 bar, 1 thin plate and several connections between them.

3.3 Excitations

This section discretises the excitations found in Section 2.4.

3.3.1 The Bow

The discretisation of the friction model in Eq. (12) as well as its implementation using an iterative solver such as Newton-Raphson is well-covered in the literature (see e.g. [13, Ch. 8]). The spatial Dirac delta function in Eq. (14) is discretised using cubic interpolator $J_{p,3}$ defined as

$$J_{p,3}(\chi_i) = \frac{1}{h}\begin{cases} -\alpha_i(\alpha_i-1)(\alpha_i-2)/6, & p = p_i - 1, \\ (\alpha_i-1)(\alpha_i+1)(\alpha_i-2)/2, & p = p_i, \\ -\alpha_i(\alpha_i+1)(\alpha_i-2)/2, & p = p_i+1, \\ \alpha_i(\alpha_i+1)(\alpha_i-1)/6, & p = p_i+2, \\ 0, & \text{otherwise.} \end{cases} \tag{34}$$

where $l_i = \lfloor \chi_i / h \rfloor$ and $\alpha_i = \chi_i/h - l_i$.

3.3.2 Hammer and Pluck

The discrete-time definition of Eq. (18) will not be given here. The discretisation of a mass-spring system can be found in e.g. [11, 17], and a derivation of the collision between the mass-spring and a 1D system using the non-iterative collision method used here (from [26]) is given in [13, Sec. 10.2].

For the pluck, τ will be set to 0 if $|f_e/h| > \varphi$, where φ is a threshold. This simulates a plucking interaction. Note that the force is scaled by the grid spacing so that the plucking interaction will be similar for strings with different parameters (and thus different values of h).

4. REAL-TIME APPLICATION

A real-time application implementing the models above has been created in C++ using the JUCE framework [2]. Figure 2 shows the graphical user interface of the application using the guitar preset for illustration. The application is controlled using the mouse, and divided into three main parts: the instrument area, the control panel (bottom) and the excitation panel (right). After describing the system architecture, this section will go into detail for all three components of the application. An extensive demo of the application, going through all of the functionality described in this section, can be found via [27].

4.1 System Architecture

Figure 3 illustrates the architecture of the audio-generating part of the application. The application can contain several independent instruments something about that this is more for future implementation in VR each of which can contain several resonators. Furthermore, instruments contain information about the connections between various resonators. Every 1D resonator has three exciter modules, one for each of the excitations described in this paper. Only one of the exciter modules is activated at a time, controlled by the excitation panel (see Section 4.4).

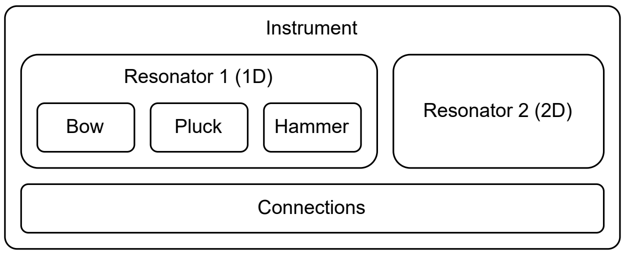

Figure 3. System Architecture.

4.2 Instrument area

The instrument area is vertically and equally subdivided over the amount of instruments in the application, which in turn are subdivided vertically and equally into the amount of resonators they contain.

4.2.1 Resonator Modules

The available resonators are: the stiff string, bar, membrane, thin plate and stiff membrane. The states of the

[2] `https://juce.com`

1D models are visualised as cyan (string) and purple (bar) lines and the states of 2D models as grey-scale squares. If the 'Add Resonator Module' button is clicked (see Section 4.3), a window pops up where the user can change all parameters of the resonator module. For the stiff string, one can choose an advanced (all parameters) and non-advanced (only the fundamental frequency and radius) list of parameters. For 2D models, an extra parameter 'maxPoints' is given which alters the grid spacing such that the number of grid points does not surpass this. This is to not overload the CPU and keep the application running in real time.

4.2.2 Connections

Connections between resonators are visualised using coloured (dotted) lines (also see Figure 2). Rigid connections are shown in solid green, linear springs in dotted orange and non-linear springs in dotted magenta. A connection location on a 1D system is accentuated using a circle of the same colour, and the same is done for a 2D system using a rectangle. A solid circle along the connection line indicates the mass ratio between the gird points of the two components ($\mathcal{M}_{s_c}/\mathcal{M}_{r_c}$); the closer this is to one component, the heavier a grid point of that component is with respect to the other.

If connections are edited, the currently active connection will be indicated by a yellow 'halo' around the connection locations. If a user tries to overlap two connections, the currently active connection will be removed from the application.

4.2.3 Exciter Modules

One can interact with the various 1D resonator modules in the instrument area. Table 1 shows how the mouse is related to various excitation parameters.

The bow is visualised with a yellow rectangle with a moving gradient, the speed of which depends on the value of the bow velocity (v_{B} in Eq. (13)). The mass used for the hammer and pluck (z in Eq. (18)), is visualised using a yellow circle and follows the mouse / cursor. The excitation distribution (e_{e} in Eq. (19)) is visualised using a grey ellipse around the cursor. Its width is determined by the value of e_{e}. For the hammer, a mouse click sets $\tau = 0$ in Eq. (18). If the hammer is triggered when the mass is further away from the string, the excitation will be of higher amplitude.

Mouse action	Bow	Hammer & Pluck
x-position	χ_{B}	χ_{e}
y-position	-	z_{off}
Scroll-wheel	$-0.2 \leq v_{\mathrm{B}} \leq 0.2$	$h \leq e_{\mathrm{w}} \leq 10h$
Click	-	τ (hammer only)

Table 1. Mouse interactions related to the various excitations.

4.3 Control panel

The control panel contains several buttons to edit an existing instrument. The main buttons are: 'Add Instrument', 'Add Resonator Module', 'Remove Resonator Module', 'Edit In- Outputs', 'Edit Connections', 'Edit Groups'. If any of the three latter buttons are clicked, the control panel will show instructions on the chosen option. The visibility of the buttons changes depending on the state of the application. Additional buttons are related to loading and saving presets, further elaborated on in Section 4.5.

If any button is pressed, the states of all resonator modules will be set to 0 to prevent audible artefacts when editing the settings.

4.3.1 Outputs

The output of any model can be obtained by listening to $q(\boldsymbol{l}_{\mathrm{o}}, t)$ for an output location $\boldsymbol{l}_{\mathrm{o}}$. In the application it is possible to change the output locations by clicking the 'Edit In- Outputs' button. For 1D systems, outputs will show as downwards pointing arrows from the system state at the respective output locations. For 2D systems, rectangles around the output grid point are used. Left, right and stereo channels are shown in white, red and yellow respectively. Functionality to add inputs has been left for future work (see Section 6).

4.3.2 Groups

Various 1D models can be grouped together to be excited simultaneously. If the 'Edit Groups' button is pressed, the user can add groups and click on resonator modules to add them to groups

4.4 Excitation panel

The excitation panel contains a dropdown menu with the various excitations: 'bow', 'hammer' and 'pluck'. Selecting one of these activates the corresponding exciter module in all 1D resonator modules. Furthermore, an 'Excite!' toggle (de)activates the currently active exciter module for all 1D resonator modules. If the excitation is deactivated, all resonators (including 2D ones) can be excited using a raised cosine (although this is only used for testing purposes).

4.5 Presets

Presets are saved in '.xml' files and have the following structure

- Instrument 0
  - Resonator 0 (type), parameters
    - Parameter 0 (value)
    - Parameter 1 ...
    - Output 0 (channel, loc)
    - Output 1 ...
  - Resonator 1 ...
  - Connection 0 (type)
    - Resonator idx 0
    - Location 0
    - Resonator idx 1
    - Location 1

- Connection 1 ...

- Instrument 1 ...

4.6 Fixed parameters

The parameters of the resonator modules can easily be altered by the user when adding one to the application. The parameters found in Table 2, however, can not be altered by the user, and have been set to result in pleasing sounds for many different model configurations.

Name	Symbol (unit)	Value
Bow		
Free parameter	a_{B} (-)	100
Bow force	f_{B} (N)	$40 \cdot \rho A$
Hammer / pluck		
Mass	M_{z} (kg)	0.01
Spring coefficient	K_{z} (N/m)	1000
Collision stiffness	K_{e} (N/m$_{\text{e}}^{\alpha}$)	10^6
Nonlin. coll. coeff.	α_{e} (-)	1.3
Pluck force threshold	φ (N/m)	500
Connections		
Linear spring coeff.	K_1 (N/m)	10^8
Nonlin. spring coeff.	K_3 (N/m^3)	10^{10}

Table 2. List of fixed parameter values.

5. RESULTS AND DISCUSSION

Table 3 shows the CPU usage

Tests were carried out on a MacBook Pro with a 2,3 GHz Intel i9 processor.

To the best of the authors' knowledge, the way of real-time interactive excitation used for the hammer and the pluck has not been done before.

Although experiments with overlapping connections have been done, it has been decided to exclude these from the application. Following [11], although any number of overlapping connections can be explicitly calculated, one needs to do a matrix inverse every sample. Therefore, it was decided to exclude overlapping connections from the application.

Strings ($N = 118$)				
No.	1	5	10	20
No graphics	1.7	7.4	12.3	26.8
Graphics	8.3	14.0	19.4	35.0
Plate				
$N_x \cdot N_y$				
No graphics				
Graphics				
Two connected strings ($N = 118$)				
Type	None	Rigid	Linear	Nonlinear
No graphics	3.8	8.3	13.1	13.3
Graphics	12.7	28.4	42.3	43.1

Table 3. CPU usage (in %) of two identical strings ($N = 118$) with all moving grid points connected ($C = 117$) using various connection types.

6. CONCLUSION

Applying excitations to 2D systems.

Port to Virtual Reality

Adding inputs so that the application can be used as an effect.

Acoustic tubes

Exciting resonator with exciter of another (bowed tube, lip-excited string)

Pitch change

Acknowledgments

This work has been funded in part by the European Art-Science-Technology Network for Digital Creativity (EASTN-DC), project number 883023.

7. REFERENCES

[1] G. Borin, G. De Poli, and A. Sarti, "A modular approach to excitator-resonator interaction in physical models syntheses," *Proceedings of the International Computer Music Conference*, 1989.

[2] C. Cadoz, A. Luciani, and J.-L. Florens, "Responsive input devices and sound synthesis by simulation of instrumental mechanisms: the CORDIS system," *Computer Music Journal*, vol. 8, no. 3, pp. 60–73, 1983.

[3] J. D. Morrison and J.-M. Adrien, "Mosaic: A framework for modal synthesis," *Computer Music Journal*, vol. 17, no. 1, pp. 45–56, 1993.

[4] J.-M. Adrien, "The missing link: Modal synthesis," in *Representations of Musical Signals*, G. De Poli, A. Picalli, and C. Roads, Eds. MIT Press, 1991, pp. 269–298.

[5] R. Rabenstein, S. Petrausch, A. Sarti, G. D. Sanctis, C. Erkut, and M. Karjalainen, "Block-based physical modeling for digital sound synthesis," *IEEE Signal Processing Magazine*, vol. 24, no. 2, pp. 42–54, 2007.

[6] J. O. Smith, "Physical modeling using digital waveguides," *Computer Music Journal*, vol. 16, no. 4, pp. 74–91, 1992.

[7] P. Ruiz, "A technique for simulating the vibrations of strings with a digital computer," Master's thesis, University of Illinois, 1969.

[8] L. Hiller and P. Ruiz, "Synthesizing musical sounds by solving the wave equation for vibrating objects: Part I," *Journal of the Audio Engineering Society (JASA)*, vol. 19, no. 6, pp. 462–470, 1971.

[9] ——, "Synthesizing musical sounds by solving the wave equation for vibrating objects: Part II," *Journal of the Audio Engineering Society (JASA)*, vol. 19, no. 7, pp. 542–550, 1971.

[10] A. Chaigne, "On the use of finite differences for musical synthesis. Application to plucked stringed instruments," *Journal d'Acoustique*, vol. 5, no. 2, pp. 181–211, 1992.

[11] S. Bilbao, "A modular percussion synthesis environment," in *Proceedings of the 12th International Conference on Digital Audio Effects (DAFx-09)*, 2009.

[12] S. Bilbao, A. Torin, P. Graham, J. Perry, and G. Delap, "Modular physical modeling synthesis environments on GPU," in *Proceedings of the International Computer Music Conference (ICMC)*, 2014.

[13] S. Willemsen, "The emulated ensemble: Real-time simulation of musical instruments using finite-difference time-domain methods," Ph.D. dissertation, Aalborg University Copenhagen, Jul. 2021.

[14] S. Bilbao, M. Ducceschi, and C. Webb, "Large-scale real-time modular physical modeling sound synthesis," in *Proceedings of the 22th International Conference on Digital Audio Effects (DAFx-19)*, 2019.

[15] D. Südholt, R. Russo, and S. Serafin, "A faust implementation of coupled finite difference schemes," in *Proceedings of the 2nd Nordic Sound and Music Computing (NordicSMC) Conference*, 2021.

[16] S. Willemsen, N. Andersson, S. Serafin, and S. Bilbao, "Real-time control of large-scale modular physical models using the sensel morph," *Proc. of the 16th Sound and Music Computing (SMC) Conference*, pp. 275–280, 2019.

[17] S. Willemsen, S. Serafin, S. Bilbao, and M. Ducceschi, "Real-time implementation of a physical model of the tromba marina," in *Proceedings of the 17th Sound and Music Computing Conference*, 2020, pp. 161–168.

[18] T. Lasickas, S. Willemsen, and S. Serafin, "Real-time implementation of the shamisen using finite difference schemes," in *Proceedings of the 18th Sound and Music Computing (SMC) Conference*, 2021, pp. 100–107.

[19] D. Südholt, S. V. K. Lyster, O. B. Winkel, and S. Serafin, "A real-time interactive physical model of the langeleik using finite difference schemes and web audio," in *Proceedings of the 2nd Nordic Sound and Music Computing (NordicSMC) Conference*, 2021.

[20] S. R. Mosen, O. I. Kristjansson, A. Adjorlu, and S. Serafin, "Real-time implementation of a physical model of the persian santur," in *Proceedings of the 2nd Nordic Sound and Music Computing (NordicSMC) Conference*, 2021.

[21] J. Bensa, S. Bilbao, R. Kronland-Martinet, and J. O. Smith, "The simulation of piano string vibration: From physical models to finite difference schemes and digital waveguides," *Journal of the Acoustical Society of America (JASA)*, vol. 114, no. 2, pp. 1095–1107, 2003.

[22] C. Webb and S. Bilbao, "On the limits of real-time physical modelling synthesis with a modular environment," in *Proceedings of the 18th International Conference on Digital Audio Effects (DAFx)*, 2015.

[23] N. H. Fletcher and T. D. Rossing, *The Physics of Musical Instruments*. Springer, 1998.

[24] S. Bilbao, *Numerical Sound Synthesis: Finite Difference Schemes and Simulation in Musical Acoustics*. John Wiley & Sons, 2009.

[25] H. Hertz, "Über die berührung fester elastischer körper," *Journal für die Reine und Angewandte Mathematik*, vol. 92, pp. 156–171, 1881.

[26] M. Ducceschi, S. Bilbao, S. Willemsen, and S. Serafin, "Linearly-implicit schemes for collisions in musical acoustics based on energy quadratisation," *Journal of the Acoustical Society of America (JASA)*, vol. 149, pp. 3502–3516, 2021.

[27] S. Willemsen, "Modular instrument demo," 2021. [Online]. Available: https://youtu.be/o6I2DlFIbsc