Physical Audio Modeling

Modeling & Simulation Checkpoint 2

Karl Hiner, Ben Wilfong

April 2023

1 Division of Work

The modeling is naturally split into two components. In the first, the finite element method is used to produce global mass and stiffness matrices which is being completed by Ben. These outputs are then used as inputs to the physical audio modeling, which Karl is completing along with the application interface.

2 Completed and Remaining Work

2.1 Finite Element Modeling

The current capabilities of the finite element code are as follows:

1. Access command line arguments to define the mesh file and the material properties (density, Young's Modulus, and Poisson's Ratio).
2. Read a .obj file and fill arrays containing information about the vertex locations, element connectivity, and element centroids.
3. Use the arrays containing information about vertex locations, element connectivity, and element centroids to compute element stiffness matrices.
4. Use the arrays containing information about vertex locations, element connectivity, and element centroids to compute element mass matrices.
5. Use the element stiffness and mass matrices to form the global mass and stiffness matrices.
6. Output the global mass and stiffness matrices for reading by the physical audio modeling code.

The next steps in implementation are as follows:

1. Perform verification and validation.
2. Add OpenMP directives to accelerate execution (Optional).

2.2 Physical Audio Modeling & Application

Completed work:

- Load/Save 3D meshes from/to `.obj` files.
- A minimally-featured custom 3D mesh viewer.
- Load `.svg` files as 2D profiles, with support for dragging curve control points to manipulate the profile.
- Generate an axisymmetric 3D triangular mesh by extruding the 2D profile, with controls over the profile curve tesselation, number of extrusion slices, radial profile displacement.
- Generate a tetrahedral mesh from the 3D triangular mesh, with a toggle between viewing the tetrahedral or triangular mesh.
- Generate Faust audio DSP code from the tetrahedral mesh (via the existing work, `mesh2faust`, with controls over the material properties, including presets for a small subset of materials (such as aluminum, copper, steel, and glass).
- Continuously play generated audio through the selected audio device, with parameters to change the following in real-time:
  - Excitation position index
  - T60 decay/slope (the envelope profile of the bell ringing)
  - Hammer hardness and size
  - Overall gain
  - Strike gate
- Control the sample rate of audio generation and playback, as well as controls to start and stop the audio device, mute audio, and control the output volume.

Remaining work:

- Use our axisymmetric solver (via file IO), with a toggle between the 3D tetrahedral vs. 2D axisymmetric models.
- Add parameters to control the following for the tetrahedral FEM path (via `mesh2faust`):
  - Min/max frequency modes to model (currently hard-coded to 20/10k Hz, respectively)
  - Target number of modes to compute for FEM
  - Target number of modes to synthesize
- Additional control over excitation positions. Currently, we provide a dropdown to pick which excitation position index to excite with the "Gate" button. We plan to implement the following additions:
  - Limit the range of excitation position dropdown to those that are actually synthesized.
  - Show excitation position points & labels on mesh, and visually indicate little brightness animation when one is "struck" (via a gate).
  - Hover and click directly on an enabled excitation vertex on the 3D mesh to strike it.

  - Provide other means to choose excitation positions other than specifying the number and sampling them uniformly from the mesh indices. (Perhaps selecting vertices visually on the 3D mesh.)

- Add a parameter to control how fine-grained the generated tetrahedral mesh is. As mentioned above, there are already ways to control the granularity of the 2D profile used to extrude the axisymmetric 3D triangular mesh. However, the parameters behind the tetrahedral mesh generation from the triangular mesh are hard-coded. Supporting lower granularity could provide another tradeoff between realism and generation speed.
- Explore the effect of scaling of the mesh, corresponding to physical size. Currently, we normalize the mesh so that the largest dimension is equal to one.
- Stretch goal: Add waveform and spectrogram views.
- Stretch goal: Use microphone input as a "gate" signal to demonstrate use as a digital audio effect.