Tabletop Ensemble: Touch-Enabled Virtual Percussion Instruments

Zhimin Ren [*1], Ravish Mehra [†1], Jason Coposky [‡2], and Ming C. Lin [§1]

[1]University of North Carolina at Chapel Hill
[2]Renaissance Computing Institute

Abstract

We present an interactive virtual percussion instrument system, *Tabletop Ensemble*, that can be used by a group of collaborative users simultaneously to emulate playing music in real world while providing them with flexibility of virtual simulations. An optical multi-touch tabletop serves as the input device. A novel touch handling algorithm for such devices is presented to translate users' interactions into percussive control signals appropriate for music playing. These signals activate the proposed sound simulation system for generating realistic user-controlled musical sounds. A fast physically-based sound synthesis technique, *modal synthesis*, is adopted to enable users to directly produce rich, varying musical tones, as they would with the real percussion instruments. In addition, we propose a simple coupling scheme for modulating the synthesized sounds by an accurate numerical acoustic simulator to create believable acoustic effects due to cavity in music instruments. This paradigm allows creating new virtual percussion instruments of various materials, shapes, and sizes with little overhead. We believe such an interactive, multi-modal system would offer capabilities for expressive music playing, rapid prototyping of virtual instruments, and active exploration of sound effects determined by various physical parameters in a classroom, museum, or other educational settings. Virtual xylophones and drums with various physics properties are shown in the presented system.

CR Categories: H.5 [Information interfaces and presentation]: .—H.5.5Sound and Music Computing. H.5.2User Interfaces;

Keywords: Multi-touch tabletop, multi-modal interaction, sound simulation

1 Introduction

Music is an integral part of our artistic, cultural, and social experiences and an important part of our life. With the recent advances in computing, scientists and engineers have created many digital musical instruments and synthesizers to perform, edit, record, and play back musical performances. Meanwhile, inventions of novel human computer interaction systems show a new dimension for computer applications to evolve. Particularly, multi-touch interfaces in many forms have been well studied and become prevalent among average users. These devices enable expressive user controls which are suitable for digital music playing. However, it still remains a challenge to build a virtual musical instrument system that allows users to intuitively perform music and generates life-like musical sounds that closely corresponds to user interaction in real time.

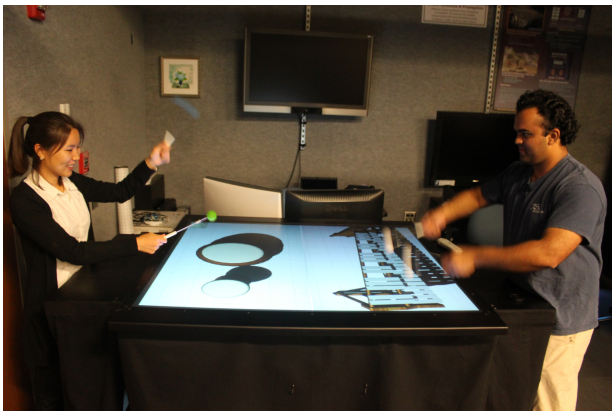


Figure 1: Tabletop Ensemble *Multiple players performing music using our virtual percussion instruments.*

In this paper, we present a virtual percussion instrument system using coupled multi-touch interfaces and fast physically-based sound simulation techniques that offer an alternative paradigm that allowing users to play several different virtual musical instruments *simultaneously* on the same platform with no overhead. The optical multi-touch table provides an interface for users to intuitively interact with the system as they would with real percussion instruments. The proposed system setup accurately captures users' performance actions, such as striking position, striking velocity, and time of impact, which are then interpreted with our *input handling* module and used to control the simulated sounds accordingly.

In addition, the size of this tabletop system enables multiple users to collaboratively participate in the musical performance simultaneously. The sound synthesis, acoustic effect simulation, and the coupling scheme between the two presented in this paper can generate rich and varying sounds for multiple sounding objects in real time. This feature also makes a collaborative and realistic music playing possible. In addition, these sound simulation techniques preserve the flexibility for easily creating new instruments of different materials, shapes, and sizes. Figure 1 shows multiple users playing virtual percussion instruments on our tabletop system. A xylophone and a set of drums of various sizes, shapes, and materials are implemented to demonstrate the system.

Main Contribution: This work is the first known system that uses physically-based sound synthesis and propagation algorithms to simulate virtual percussion instruments on an optical multi-touch tabletop. It offers the following unique characteristics over existing digital instruments:

- **Direct and Intuitive Multi-Modal Interface and Handling Suitable for Percussion Instruments** The multi-touch tabletop user interface enables users to intuitively control a virtual percussion instrument. Novice users can interact with the

*e-mail: zren@cs.unc.edu
†e-mail: ravishm@cs.unc.edu
‡e-mail: jasonc@renci.org
§e-mail: lin@cs.unc.edu

system with no learning curve. A novel algorithm for mapping touches on optical touch-sensing surfaces to percussion instrument controls is proposed to accurately interpret users' interaction with the tabletop. (Section 4)

- **Physically-Based Sound Generation** A physically-based sound synthesis technique is adopted to generate instrument sounds given interpreted user interactions. A numerical sound propagation simulation algorithm is used to model the acoustic effects of the instrument's air cavity. We propose a simple yet effective system integration setup between synthesis and propagation to enable real-time simulation of dozens of sounding instruments. The generated sound closely corresponds to users' interaction with the system, e. g. striking position, velocity etc. (Section 5 and Section 6)

- **A Reconfigurable Platform for Different Instruments and Multiple Players** Physically-based sound simulation offers the ease of employing various sounding materials, sizes, and shapes for easily creating new virtual instruments, thus making the system reconfigurable with little overhead. Our system setup is capable of accommodating and handling multiple users' simultaneous interaction with the virtual instruments and simulate the musical tunes for many sounding objects at interactive rates. It enables multiple users to collaborate for performing music on a single, portable platform.

We have also conducted early pilot study to solicit qualitative feedback and suggestions from users with various music background and skills. We briefly discuss the results and limitations of this system in Section 8.

2 Previous Work

This work builds upon two distinct large bodies of research: one in user interfaces for virtual musical instruments and the other in sound simulation for digital music generation.

2.1 Multi-Touch Interfaces for Musical Instruments

Electronic musicians have long adopted Musical Instrument Digital Interface (MIDI) protocols for creating digital music. A plethora of MIDI controllers have been built that enable users to perform music. For example, there are MIDI keyboards and MIDI drum pads. Moreover, other novel interfaces have also been explored for virtual instruments [Miranda and Wanderley 2006; Chuchacz et al. 2007; Weinberg and Driscoll 2007]. However, none of them is as intuitive or easy-to-use for average users as multi-touch interfaces.

Multi-touch hardware and software have been actively studied and innovated for many years. As early as 1985, Buxton et al. [1985] analysed touch-input devices and compared them with conventional mice and joysticks. Nowadays, multi-touch technologies are mainly categorized into three types of devices: *capacitive sensing*, *resistive sensing*, and *optical sensing*. Han [2005] invented a low-cost optical multi-touch sensing technology that made fast multi-touch on a large surface more practical. Rosenberg and Perlin [2009] designed an inexpensive and lightweight multi-touch input pad that provides pressure-sensing, and this device can be attached to assorted displays for a direct touch experience. With the prevalence of capacitive sensing devices like the iPhone and iPad, average digital device users have become familiar, comfortable, and even used to interacting with multi-touch. Such expressive interfaces encourage much more intuitive and natural interactions from users and also offer applications additional dimensions for input information.

One prominent format of such interface is the multi-touch tabletop, which has low cost and the capability for multiple-user collaboration. It is a great candidate for building multi-modal interactive systems. Researchers have employed such technology for creating music and sounds in general. Davidson and Han [2006] employed their multi-touch tabletop to control widgets that modify sound synthesis. Kaltenbrunner et al. [2006] designed a tangible multi-touch interface that allows both local and remote collaboration on synthesizing audio. Hochenbaum and Vallis [2009] built a multi-touch table and applied it to generating parametric sounds and remotely controlling real drums. However, none of these works attempts to virtually simulate musical instruments. Various techniques for finger tracking are surveyed by Schöning et al. [2009]. These methods facilitate higher-level touch interpretation and gesture recognition. Nevertheless, none of those techniques handles percussive interactions, in which case striking velocity is required to be estimated.

2.2 Sound Simulation for Musical Instruments

2.2.1 Sound Synthesis

Sound synthesis methods are well studied and applied to digitally generating music. The most realistic yet fast approach is sample-based methods, which process recorded audio samples with parametric models. Some of these models related to music instrument synthesis are presented by Cook [2002]. However, sample-based methods do not offer intuitive or rich control that closely maps to real-world sound generation mechanisms. On the other hand, physical models are superior in terms of natural and expressive controls, and they promise easy flexibilities for creating artificial instruments with expected audio effects. Numerical methods by Bilbao [2009] produce high-quality music instrument sounds. However, like other time-domain wave-equation based approaches, they are not fast enough for real-time applications that demand synthesizing multiple instruments simultaneously.

To physically-based synthesize sound in real time, Van den Doel and Pai [1998] introduced a general framework using resonance modes, i.e. *modal synthesis* ([Adrien 1991; Shabana 1997]). This approach generates sound dependent on the materials, shapes, and strike positions of the simulated sounding objects, while it assumes linear dynamics for the vibrating objects. Modal synthesis applied to simple shapes (e.g. strings, tubes, membranes, and more) with analytical modal analysis results can be found in [Cook 2002]. Bruyns [2006] showed modal synthesis on arbitrary shapes and compared the synthesized sounds' spectral contents with real-world recordings. Modal synthesis is a suitable approach for our purposes, due to its low run-time costs and flexibility as a physical model approach.

2.2.2 Acoustic Effects

The techniques to capture acoustic effects of a space can be classified into two categories - *geometric acoustics* (GA) and *numerical acoustics* (NA). GA approaches are based upon the geometric approximation of rectilinear propagation of sound waves. A large variety of methods have been developed starting from ray-tracing and image source methods in early days to current techniques that include beam tracing [Funkhouser et al. 2004], frustum tracing [Chandak et al. 2008], phonon tracing [Deines et al. 2006] and many more. A more detailed survey can be found at [Funkhouser et al. 2003].

NA techniques solve the wave equation of the sound propagation and therefore capture all the wave effects of sound. Typical numerical techniques include Finite Element Method (FEM) [Thompson 2006], Boundary Element Method (BEM) [Brebbia 1991], Finite

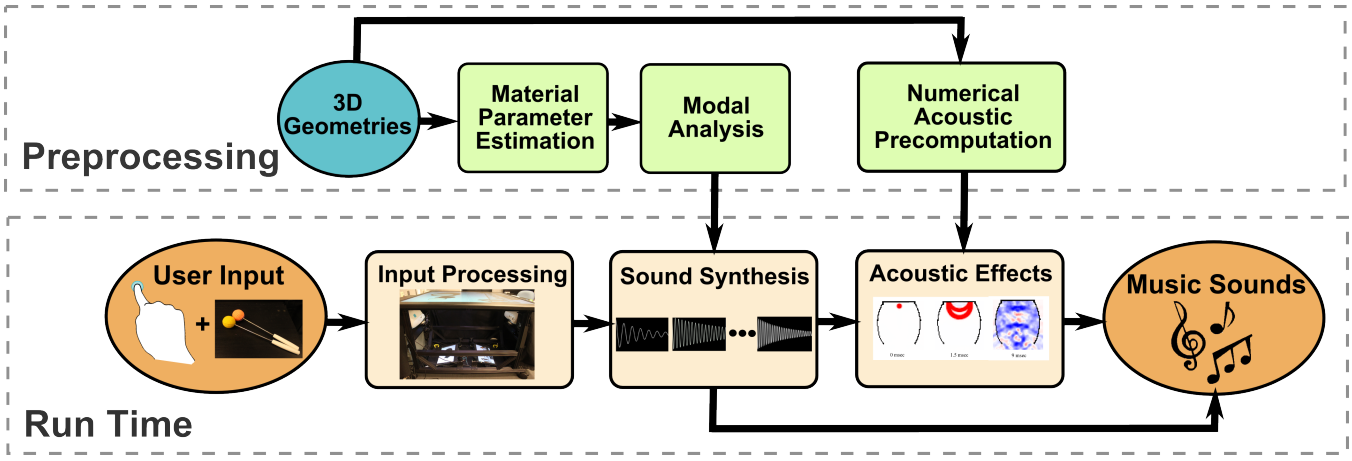


Figure 2: The system pipeline of Tabletop Ensemble. *During the preprocessing stage, our system automatically extracts the material parameters from a sample audio recording for a musical instrument. Given the geometry of each virtual instrument and its material parameters, we can precompute the acoustic effects due to the instrument's body cavity. At run time, user interaction with the multi-touch table is first interpreted by the input processing module and forwarded to sound synthesis engine. Synthesized sounds for instruments with cavity structures are modulated by the precomputed acoustic effects to generate the final audio.*

Difference Time Domain (FDTD) [Sakamoto et al. 2006], spectral methods [Boyd 2001] and more recently Adaptive Rectangular decomposition (ARD) technique [Raghuvanshi et al. 2009]. NA techniques have high computational cost and used only in offline simulations.

Recently, a new wave-based acoustic simulation technique has been proposed by [Raghuvanshi et al. 2010] for performing real-time sound propagation in complex static 3D scenes for multiple moving sources and listener. This technique captures all the acoustic wave effects like diffraction, reverberation, etc., and exploits human perception to efficiently encode the acoustic response reducing the overall memory requirements. It divides the computation into three stages: an off-line simulation to compute acoustic response of the scene, an off-line perceptually-motivated encoding of this response, and a fast run-time system to perform auralization. We adopt this method to introduce acoustic effects to our percussion instruments.

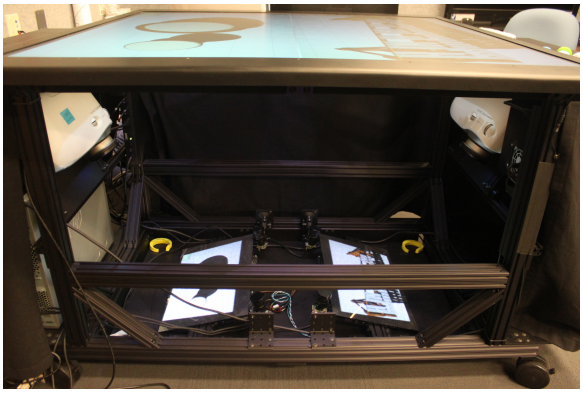


Figure 3: *The optical multi-touch table with diffuse side illumination, upon which our virtual percussion instruments are built.*

3 System Overview

This section gives an overview on the hardware configuration and the software modules that make up our touch-enabled virtual instrument system.

3.1 Hardware Apparatus

Our application is developed on top of a custom-built optical multi-touch table, using the diffuse side illumination technique. By employing a sheet of $CyroAcryliteEndLighten^{TM}$ with polished edges and a 5-foot strip of LED Edge-View Ribbon Flex from $EnvironmentalLights^{TM}$, we are able to distribute the infrared (IR) illumination more evenly. The touch detection for our tabletop is handled by four Point Grey Firefly MV FMVU-03MTM cameras. For the projection surface, we use a thin, 3mm-sheet of $AcryliteRP7D513$ rear projection acrylic. This design works out well since the thin sheet protects the more expensive Endlighten material and the projection surface has a nice touch. The table has a 62" diagonal work surface and is 40" tall (see Figure 3) with two high-definition rear-projection display (1920×2160 pixels), driven by a 3.2 GHz quad-core Xeon processor.

The entire table was designed through an architecture of commodity-level components and custom software. This high-resolution interactive display provides an effective means for multiple users to directly interact with their application system, data, and peripherals. It comfortably allows 4 to 6 people to work at the table simultaneously. The table allows tracking multiple (up to 20 or so) interactions on its surface by properly-sized objects that are infrared reflective. Tracked touch events' IDs, timestamps, contact positions, and contact area information are provided for application development. The size of this multi-touch tabletop and its interaction mechanism make it an attractive, intuitive physical interface for playing virtual percussive instruments.

Optical multi-touch interfaces like this multi-touch table accurately tracks touch points' spatial information on the 2D touch plane. However, without additional ceiling mounted camera or tracking, it is difficult to obtain information on how fast an object is approaching the table surface, i. e. hitting velocity, which is one of the most important control parameters in playing percussive instruments. In order to obtain this parameter, we propose using deformable bodies as the hitting object, and we design a velocity estimation algorithm based on this deformation data. As the input to the system (as shown in Figure 2 and supplementary video), soft sponge balls of

roughly four centimeters in diameter are used as the mallet heads for exciting the virtual instruments. Users can also play the instruments with fingers, which are tracked in the same way as the sponge balls with a slightly different configuration. The input handling process is explained in detail in Sec. 4.

3.2 Algorithmic Modules

Figure 2 illustrates the overall algorithmic pipeline. The virtual percussion instrument implementation depends on two separate stages. One is the preprocessing phase, where an instrument's 3D geometry and a recorded impact sound of an example material are analyzed. In *modal analysis* and *numerical acoustic precomputation*, *resonance* and *wave simulation* data are generated respectively, which are later used in sound synthesis and acoustic effect modules in the next phase. During runtime, touch messages from tabletop hardware are interpreted by the *input processing* module, and excitation to virtual sounding instruments are generated accordingly. Given the excitation, *sound synthesis* module generates sound using modal synthesis techniques. For instruments with air cavities, their synthesized sounds are further processed by *acoustic effects* (i.e. sound propagation) module for adding important audio effects due to resonance in their cavities. Details on the preprocessing and runtime modules are presented in the following sections.

4 Touch Input Processing

The impulse applied by the striking body to the virtual instrument is directly used as excitation to our sound synthesis engine. We choose impulse over pressure, because pressure at one instance does not necessarily reflect how hard users are hitting, e.g. users can be statically pressing against the surface but this should not excite the vibration of the virtual sounding objects. Therefore, how accurately we can capture the impulse information determines how well we can model a performer's control over the generated music sounds. Impulses are proportional to the change of velocity, and derived by estimating the rate of change of the striking body's velocity.

Without the loss of generality, let us assume the touch-surface is the X-Y plane. It is relatively easy and already a standard technique to track the velocity in the X-Y plane on an optical multi-touch tabletop. However, velocity along the Z-axis (perpendicular to the tabletop) cannot be directly acquired. While systems with extra cameras mounted perpendicular to X-Y plane or full 3D motion capture are feasible for tracking velocity in Z-axis, such a set up adds additional hardware and calibration overhead. More importantly, with camera-based tracking, when multiple users are interacting with the system, multiple interaction points (e.g. hands) occlude one another, which might greatly impact the accuracy of the tracking. For processing multiple (and possibly simultaneous) touch inputs, we propose to use soft bodies, representing either sponge balls as the mallet heads or user's finger tips, as an input device for the multi-touch table. We describe how velocity perpendicular to the touch surface can be tracked through the deformation of the soft bodies. This approach involves simple and easy computations that can be adopted to add velocity tracking for the third dimension beyond the touch surface for any type of optical touch-enabled device – *with or without pressure sensing*.

4.1 Z-Velocity Tracking

In order to track Z-velocity, a sequence of occlusion information registered by the multi-touch system is recorded for each striking soft body. The recorded occlusion information includes the touch center position, the occluded area, and the time stamp of this occlusion event. Figure 4 illustrates a snapshot in time of a squeezed soft

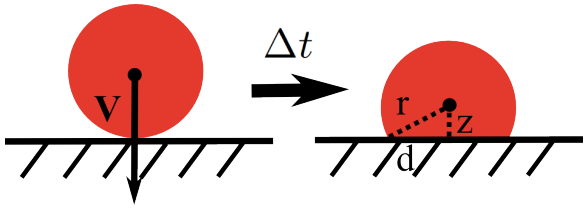


Figure 4: *A snapshot of the cross section of a soft ball striking against X-Y plane at a velocity V. After time Δt, the ball is deformed, and its center position in Z-direction can be calculated. Velocity in Z-axis can be derived from this position and the elapsed time information as discussed in Sec. 4.1.*

sphere after it strikes against the X-Y plane. By using this snapshot of occlusion data, i.e. the radius of the occluded circle (denoted as d_n) and given the striking soft sphere's radius (denoted as r), we can derive the striker's center distance from the X-Y plane with $z_n = r - \sqrt{r^2 - d_n^2}$. Therefore, the average Z-velocity v_z in one time step Δt can be quickly calculated with

$$v_z = \frac{z_n - z_{n-1}}{\Delta t}. \quad (1)$$

With a single strike, one or more time steps may have elapsed, we need to use the *average* velocity throughout all time steps of the whole sequence to more accurately estimate the Z-velocity for this one strike. However, with slower strikes, the whole sequence can span a long interval. Computing the average velocity at the end of these strikes would introduce a significant latency between the hit motion and the generated audio to users. In order to eliminate perceptible latency, we adopt a temporal window. Velocity values from the initial contact time to the last time step within this window are averaged to approximate the Z-velocity of this strike. According to a perceptual study by Guski et al. [2003], 200ms of latency is the tolerance for human to reliably perceive an audio signal and a visual signal as a unitary event, and this temporal window size is also adopted and verified by Bonneel et al. [2008] for plausible sound simulation. In our case, we employ an even smaller temporal window of 100ms for average Z-velocity estimation, which gives us good results. When the occlusion area of one touch sequence decreases with time, that touch is considered a release, and the buffer associated with this touch is cleared. At each time instance, the occlusion centered within a small spatial range near a touch in the last time step is considered coming from the same soft body and therefore stored in the same buffer for velocity estimation. K-d tree is used for nearest neighbor search to accelerate this clustering process. With this approach, we can easily track a large number of simultaneous touches from multiple soft bodies, sponge balls and/or user fingers, to estimate their Z-velocity and generate corresponding excitation to our sound simulation.

4.2 Implementation and Results

Theoretically, the higher the cameras' frame rate, the more accurate the Z-velocity estimation. However, increasing frame rate also lowers cameras' resolution, which undermines the accuracy for tracking occlusion area. Through experiments, we decided on adopting 504×480 as the region of interest (ROI) resolution with the cameras in our system, and under such configuration, the frame rate is roughly 60 frames per second. Better camera hardware is likely to increase the accuracy of the proposed velocity estimation heuristic. In our implementation, the radius parameters for the sponge balls are 15 pixels, while the radius for hand contacts are set as 10 pixels.

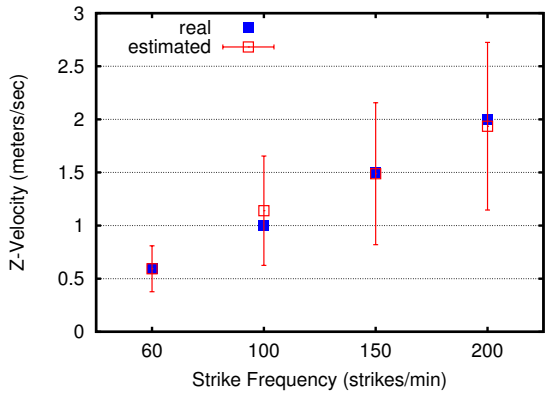


Figure 5: *Estimated Z-velocity vs. real velocity values: This experiment is performed under four different tempos for strikes, i.e. 60, 100, 150, and 200 strikes per minute.*

Using a metronome, we repeatedly strike the table from 30cm above at four different tempos, namely 60, 100, 150, and 200 strikes per minute. For each tempo, 100 such strikes are performed. The mean Z-velocity of all those strikes estimated with our method along with their standard deviation are shown as red in Fig. 5, while the real values directly computed from dividing the strike distance and the strike interval are shown as blue. With our method, we can accurately estimate the Z-velocity with only the deformation data.

5 Sound Synthesis

As mentioned in Sec. 2, *modal synthesis* technique has been employed for sound synthesis in our system. It is one of the most widely adopted approaches for generating sounds based on the first principle of physics in graphics, game, and music communities [O'Brien et al. 2002; Raghuvanshi and Lin 2006]. *Modal analysis* is performed during the preprocessing stage to analyze an arbitrary 3D geometry and its material parameters to compute the resonance modes of that object. The output is a bank of damped sinusoidal waves, i. e. *modes*. At run time, different excitations to the model trigger different modal responses. This approach assumes linear dynamics for the vibration of sounding objects with proportional damping, which is also often adopted for simplification. These assumptions make this method less suited for modeling some highly complex sound phenomena, yet sufficiently (physically) correct for our purpose in real-time synthesis of musical tunes that closely correspond to user interaction. It also offers the flexibility of changing instruments' physical properties for rapid prototyping.

One of the challenging and important elements for high-quality synthesis of modal sounds is to acquire appropriate material parameters for modal analysis. This process is time-consuming when 3D model meshes are not sufficiently fine and detailed for directly using real physical parameters. More importantly, parameters like proportional damping coefficients do not directly map to real-world materials, therefore impossible to look up for modal analysis. In our preprocessing, we adopt a new example-guided parameter estimation algorithm, using only a single example audio recording for each material [Ren et al. 2011]. Guided by a sample audio recording of a xylophone bar and of a drum, this automatic, offline process facilitates quick determination of material parameters of realistic sounding materials the simulated xylophone and a drum set.

In our system, we modulate synthesized audio for instruments that have notable acoustic effects, in addition to vibration sounds. This process is presented next in Section 6. Real-time synthesized audio samples for each sounding object are formated into buffers of size 2048 and then passed on to the acoustic simulation module in a separate interprocess communication pipe.

6 Acoustic Effects

In most musical instruments, especially percussion instruments, sound produced by a generating surface (membrane or string) propagates inside the cavity of the instrument and gets modulated due to its shape, size and the material. The propagation of sound inside the air cavity produces resonance resulting in amplification of certain frequencies and loss of others. This vibration of the generating surface along with the acoustic effect of the air cavity, gives the musical instrument its characteristic sound. Therefore, while designing virtual music instruments, it is critical to properly model acoustics of the instrument i.e. the way its shape, size and material changes the sound. In our system, we perform *one-way coupling* of the sound synthesis and acoustic simulation stages. The sound generated by the synthesis stage enters the instrument cavity after which the acoustic simulation stage propagates this sound inside the cavity to model its acoustic effects. The final propagated sound then leaves the music instrument towards the surroundings. This one-way coupling is a good approximation for percussion instruments that are open at one end like congo drums.

In order to capture the acoustics inside the instrument's cavity in a physically-accurate way, we chose wave-based simulation technique of [Raghuvanshi et al. 2010]. This technique captures all the wave-effects of sound including diffraction, interference, scattering and reverberation. It performs real time sound propagation for multiple sources in a static environments in a fast and memory efficient manner. This capability enables us to handle multiple percussion instruments and their acoustics at interactive rates and maintain low latency in our system, a critical requirement for satisfactory user-experience.

In the pre-processing stage (see Figure 6), we start with a 3D model of the instrument and assign material absorption coefficients to its various parts. We then sample positions on the sound generating surface (2D membrane in case of drum), place an impulsive sound source at each position, run an acoustics simulation [Raghuvanshi et al. 2009] and determine the sound propagated inside the instrument cavity including reflection, diffraction and interference of sound waves. This propagated sound produced by the impulsive source is called acoustic *impulse response* (IR). It completely determines the acoustics of the instrument cavity. IRs are recorded at sampled 3D locations inside the cavity and stored in a highly compact representation as discussed in [Raghuvanshi et al. 2010].

At run-time, the hit position and the corresponding sound produced by the synthesis module is passed as input to the propagation technique. The listeners are placed on the sound generating surface to capture the sound emitted by it. Based on the hit position, this technique performs a look-up of the IR corresponding to the nearest sound source at the given listener position and performs an interpolation to produce the correct IR for that hit position. This IR is convolved with the synthesized sound in real time to produce the final propagated sound capturing the acoustics of the instrument.

In our test scenarios, we have simulated the acoustics of five drums with different shapes, sizes and materials. Large drums trap the sound more effectively and hence have longer reverbs (more echoing). On the other hand, small drums placed high above the ground are less reverberant. We have also tested two different material properties - metallic and wooden. Since the metallic drums have low absorption coefficient, they have longer reverberation times compared to the highly absorbing wooden drums.

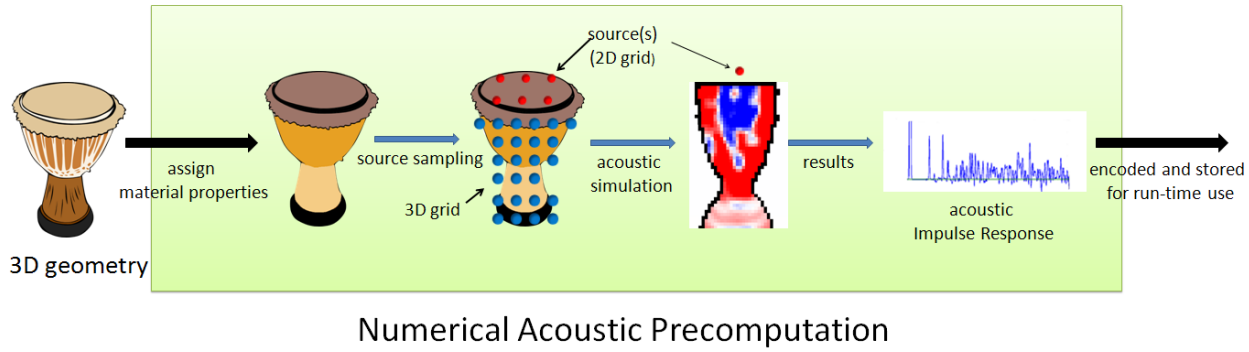
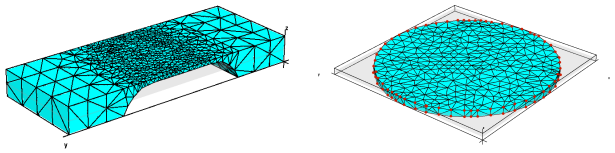


Figure 6: ***Numerical acoustics precomputation pipeline****: The input to our system is a 3D model of the virtual instrument. We assign material properties to its different parts based on the type of percussion instrument we want to model. Next, we place impulsive sound sources (red spheres) at sampled positions on its sound generating surface, run the numerical simulation and collect impulse responses at 3D grid positions (blue spheres) corresponding to each source. This impulse response is encoded and stored for run-time use.*

(a) Xylophone (b) Drum Set

Figure 7: *(a) shows a virtual metallic xylophone, and (b) shows a five-piece drum set.*



(a) Xylophone Bar Mesh (b) Drum Head Mesh

Figure 8: *Discretized mesh representation for the xylophone bar and drum head models used in this system. The red dots in 8(b) indicate fixed nodes.*

7 Instrument Modeling and Implementation

Two types of representative percussion instruments are simulated: a xylophone (Fig. 7(a)) and a set of five drums (Fig. 7(b)) with various membrane sizes, cavity shapes, and drum wall materials.

7.1 Sound Generation

We model xylophone bars with arched curve at the bottom just like real xylophone bars (see Figure 8(a)). Real xylophone bars are normally strung to their nodal points to minimize the damping from external attachment. Therefore, in our simulation, we simplified the mechanism by allowing xylophone bars to freely vibrate for sound generation purposes. For drum heads, we model them as circular plate with a small thickness. The rim vertices of a real drum are firmly attached to a drum body. Therefore, we specify all rim nodes of our virtual drum as fixed nodes in modal analysis (shown in red dots in Figure 8(b)). In our implementation, we first discretize 3D geometries into tetrahedra with TetGen [Si 2011], with no tetrahedron's radius-edge ratio greater than 2.0 (shown in Figure 8). We then perform finite element analysis on the discretized geometries to acquire the simulation meshes which are used in modal analysis.

7.2 Acoustic Simulation

We now discuss the implementation details of acoustic simulation in our multiple-drum scenario. In the pre-processing stage, the sampled sound sources are placed on membrane of each drum at 20 cm distance resulting in, typically, $5 - 10$ sampled sources per drum. We tested two material properties for the drums - metallic and wooden, having absorption coefficients of 10% and 30% respectively[1] (Fig. 9). The run-time propagation system can handle a maximum of 10 instruments playing at the same time. Since we are mainly interested in the sound propagating from the drum membrane, our listeners are placed at the center of each drum's membrane. For the multi-drum scenario, the final auralized sound is a mix of the sounds received at the listener of each instrument.

7.3 System Integration

In order to capture the acoustics of percussion instruments, we propose a simple and efficient method to couple the synthesis and acoustic simulation systems. Our sound synthesis pipeline performs *modal analysis* to generate sound due to the vibration of the drum membrane. This sound is packaged in audio buffers and transferred to the acoustics simulation system over the Windows interprocess communication (IPC) framework called *Named pipe* [framework Named pipes 2011]. To avoid any communication delay between the two systems, we use asynchronous data transfer over the pipes. At the acoustic simulation side, the audio buffers are convolved with the appropriate impulse response to generate the auralized audio. This auralized audio is sent to the sound card for playback using the XAudio2 [XAudio2 2011].

The size of the buffer and number of pipes used depends on the latency requirement of the application. Small buffer size implies low latency between the two applications but higher communication cost per byte due to large number of buffers transferred. Large buffers have low communication cost per byte but high latency. We found out experimentally that a buffer size of 2048, corresponding

[1] Absorption coefficient of 10% means the surface will absorb 10% of the incoming acoustic energy at each interaction with the sound wave

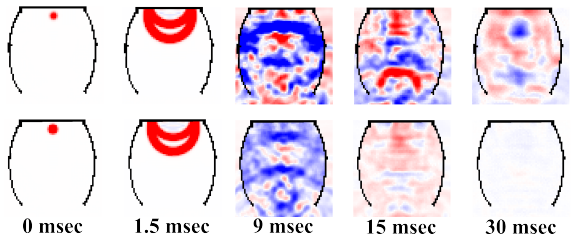


Figure 9: *Acoustic simulation results for metallic (top row) vs wooden (bottom row) drum at different time-steps with absorption coefficient of* 10% *and* 30% *respectively.*

to 50ms at sampling rate of 44.1kHz, satisfied our latency requirements. Since all the instruments can potentially be played at the same time by multiple users, we create a dedicated pipe for each sounding object in the musical instrument to transfer data from the synthesis to the acoustic simulation system. Therefore, the number of pipes is equal to number of sounding objects in the system.

8 Results and Discussions

We present a multi-modal interaction system that allows users to intuitively perform percussion music on a xylophone and a drum set. We achieve interactive handling of multiple users' touch inputs, sound synthesis, sound modulation using physically-accurate acoustic simulation, and final auralization – *all in real time*.

8.1 Results

Our multi-touch interface tracks touches on the tabletop surface and also estimates hit velocity perpendicular to the tabletop. This capability allows us to model the musical performance of the user over the percussive instrument. In the supplementary video, we show how volume of the generated sound is directly modulated by performers' hit vigor, i. e. the faster the strike against the instrument, the louder the simulated audio. Users' hit position is also accurately tracked and transformed into corresponding excitations to the system, resulting in generation of position-dependent musical sounds based on user interaction.

The sound simulation engine implemented in this system efficiently couples a sound synthesis module and a numerical acoustics simulation. Generated audio captures essential reverberation effects due to air cavity in the instrument. The coupled system handles multiple numbers of sounding objects and adds prominent acoustic effects to the synthesized sounds all in real time without perceptible latency. The complete sound simulation effects are shown on the five-piece drum set in the accompanying video. Note how added acoustic effects instantly change the overall sound quality of the drum simulation. The physically-based sound synthesis and propagation models adopted by the system also provide the flexibility for changing the simulated instruments based on their shapes, sizes, and materials easily. Results of instruments with different physical properties are also shown in the supplementary video.

We invited people of different age groups and various music playing backgrounds, from novice players to professional musicians, to play our virtual instrument system. All the users were able to interact with the system and play the instrument easily without any learning curve or significant familiarization with the setup. Multiple users were able to collaborate naturally on the tabletop. Users also appreciated the fact that the size of the virtual instruments on the touch table resembled the real xylophones and drums, which made it easy to play.

8.2 Limitations

Although the proposed hit velocity tracking and estimation method works well with the sponge balls adopted in our implementation, the estimation for direct interaction with hands is not as accurate. Limited deformation of fingers and palms, and the small variation in occluded area data make it hard to accurately estimate velocity along the direction perpendicular to the touch surface. Moreover, human hands vary in size from person to person. Even for the same person, the size of a finger contact is very different from a palm contact. Without a fixed, known deformation model for hands, it is difficult to provide correct velocity estimation purely based on the tracked deformation. One possible solution is to incorporate a user-specific hand deformation model. However, such a computation may be too costly for interactive user experiences. For accurate control over hit vigor, users employ the sponge ball mallets. Due to limited frame rates and resolution of the cameras used in our system (discussed in Sec. 4.2), the implemented system has an upper limit for strike velocity that it can handle. Therefore, when users are attacking the touch surface at a very high speed and lifting up immediately, it is likely that those touches are not registered, resulting in missed strikes.

Our user input processing through the touch-enabled interface provides users an experience that emulates performing on real instruments. Synthesized sounds correspond to users' performance and also the intrinsic characteristics of the instrument itself. However, the current system does not incorporate all user controls. For example, users cannot damp the resonance bodies with contacts to achieve articulation like staccato. Additionally, in more complex musical instruments, the propagated sound can in turn affect the vibrations on the sound generating surface resulting in a reverse feedback that has to be modeled as *two-way coupling*, which is not simulated in our current implementation.

9 Conclusions and Future Work

In conclusion, we present a virtual instrument system that enables multiple users to simultaneously perform musical pieces together on a single platform. It uses an efficient and responsive approach that interprets the user inputs from an optical multi-touch interface and generates excitation information for real-time sound simulation to create realistic sounds depending on striking position, impact vigor, instrument shapes, and instrument cavity. While our current hardware setup suits collaborative purposes for scenarios like museums and schools, these design principles can be easily adopted to run on other input devices, such as multiple tablet PC, iPad, or other commercial multi-touch displays. Based on early user feedback, this multi-modal system is intuitive, easy-to-use, and fun to play with for novice users and experienced musicians alike.

For future work, we plan to introduce new interfaces for users to change instrument parameters and properties on the fly and experience the fun of building their own virtual instruments. In addition, more accurate physical models for sound simulation can be explored to achieve richer and more realistic sounds, especially nonlinear effects in synthesis and two-way coupling between synthesis and propagation.s With further algorithmic advances and novel features, we hope to provide users with more forms of virtual instruments in the future.

Acknowledgments

We would like to thank Maggie Zhou, Dave Millman, Micah Taylor, Jake Waits, and Cameron Britt for working with our system and provide early feedback.

This research is supported in part by National Science Foundation, Intel Corporation, and Carolina Development Foundation.

References

ADRIEN, J.-M. 1991. Representations of musical signals. MIT Press, Cambridge, MA, USA, ch. The missing link: modal synthesis, 269–298.

BILBAO, S. 2009. *Numerical Sound Synthesis*. Wiley Online Library.

BONNEEL, N., DRETTAKIS, G., TSINGOS, N., VIAUD-DELMON, I., AND JAMES, D. 2008. Fast modal sounds with scalable frequency-domain synthesis. In *ACM SIGGRAPH 2008 papers*, ACM, New York, NY, USA, SIGGRAPH '08, 24:1–24:9.

BOYD, J. P. 2001. *Chebyshev and Fourier Spectral Methods: Second Revised Edition*, 2 revised ed. Dover Publications, December.

BREBBIA, C. A. 1991. *Boundary Element Methods in Acoustics*, 1 ed. Springer, October.

BRUYNS, C. 2006. Modal synthesis for arbitrarily shaped objects. *Computer Music Journal 30*, 3, 22–37.

BUXTON, W., HILL, R., AND ROWLEY, P. 1985. Issues and techniques in touch-sensitive tablet input. In *ACM SIGGRAPH Computer Graphics*, vol. 19, ACM, 215–224.

CHANDAK, A., LAUTERBACH, C., TAYLOR, M., REN, Z., AND MANOCHA, D. 2008. Ad-frustum: Adaptive frustum tracing for interactive sound propagation. *IEEE Transactions on Visualization and Computer Graphics 14*, 1707–1722.

CHUCHACZ, K., O'MODHRAIN, S., AND WOODS, R. 2007. Physical models and musical controllers: designing a novel electronic percussion instrument. In *NIME '07: Proceedings of the 7th international conference on New interfaces for musical expression*, ACM, New York, NY, USA, 37–40.

COOK, P. 2002. *Real sound synthesis for interactive applications*. AK Peters, Ltd.

DAVIDSON, P., AND HAN, J. 2006. Synthesis and control on large scale multi-touch sensing displays. In *Proceedings of the 2006 conference on New interfaces for musical expression*, IRCAM-Centre Pompidou, 216–219.

DEINES, E., MICHEL, F., BERTRAM, M., HAGEN, H., AND NIELSON, G. 2006. Visualizing the phonon map. In *Eurovis*.

FRAMEWORK NAMED PIPES, M. I. 2011. Named pipes.

FUNKHOUSER, T., TSINGOS, N., AND JOT, J.-M. 2003. Survey of methods for modeling sound propagation in interactive virtual environment systems. *Presence and Teleoperation*.

FUNKHOUSER, T., TSINGOS, N., CARLBOM, I., ELKO, G., SONDHI, M., WEST, J. E., PINGALI, G., MIN, P., AND NGAN, A. 2004. A beam tracing method for interactive architectural acoustics. *The Journal of the Acoustical Society of America 115*, 2, 739–756.

GUSKI, R., AND TROJE, N. 2003. Audiovisual phenomenal causality. *Perception & psychophysics 65*, 5, 789.

HAN, J. 2005. Low-cost multi-touch sensing through frustrated total internal reflection. In *Proceedings of the 18th annual ACM symposium on User interface software and technology*, ACM, 115–118.

HOCHENBAUM, J., AND VALLIS, O. 2009. Bricktable: A musical tangible multi-touch interface. *Proceedings of Berlin Open Converence 09*.

KALTENBRUNNER, M., JORDA, S., GEIGER, G., AND ALONSO, M. 2006. The reactable*: A collaborative musical instrument.

MIRANDA, E., AND WANDERLEY, M. 2006. *New digital musical instruments: control and interaction beyond the keyboard*. AR Editions, Inc.

O'BRIEN, J., SHEN, C., AND GATCHALIAN, C. 2002. Synthesizing sounds from rigid-body simulations. In *Proc. of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, ACM Press New York, NY, USA, 175–181.

RAGHUVANSHI, N., AND LIN, M. 2006. Interactive sound synthesis for large scale environments. In *Proceedings of the 2006 symposium on Interactive 3D graphics and games*, ACM, 101–108.

RAGHUVANSHI, N., NARAIN, R., AND LIN, M. C. 2009. Efficient and accurate sound propagation using adaptive rectangular decomposition. *IEEE Transactions on Visualization and Computer Graphics 15* (September), 789–801.

RAGHUVANSHI, N., SNYDER, J., MEHRA, R., LIN, M., AND GOVINDARAJU, N. 2010. Precomputed wave simulation for real-time sound propagation of dynamic sources in complex scenes. *ACM Trans. Graph. 29* (July), 68:1–68:11.

REN, Z., YEH, H., AND LIN, M. C. 2011. Example-Guided Physically-Based Sound Synthesis. *UNC Techinical Report*.

ROSENBERG, I., AND PERLIN, K. 2009. The unmousepad: an interpolating multi-touch force-sensing input pad. In *ACM SIGGRAPH 2009 papers*, ACM, 1–9.

SAKAMOTO, S., USHIYAMA, A., AND NAGATOMO, H. 2006. Numerical analysis of sound propagation in rooms using the finite difference time domain method. *The Journal of the Acoustical Society of America 120*, 5, 3008.

SCHÖENING, J., HOOK, J., MOTAMEDI, N., OLIVIER, P., ECHTLER, F., BRANDL, P., MULLER, L., DAIBER, F., HILLIGES, O., LOECHTEFELD, M., ROTH, T., SCHMIDT, D., AND VON ZADOW, U. 2009. Building interactive multi-touch surfaces. *Journal of Graphics, GPU, and Game Tools 14*, 3, 35–55.

SHABANA, A. 1997. *Vibration of discrete and continuous systems*. Springer Verlag.

SI, H. 2011. TetGen: A Quality Tetrahedral Mesh Generator and a 3D Delaunay Triangulator.

THOMPSON, L. L. 2006. A review of finite-element methods for time-harmonic acoustics. *The Journal of the Acoustical Society of America 119*, 3, 1315–1330.

VAN DEN DOEL, K., AND PAI, D. 1998. The sounds of physical shapes. *PRESENCE-CAMBRIDGE MASSACHUSETTS- 7*, 382–395.

WEINBERG, G., AND DRISCOLL, S. 2007. The interactive robotic percussionist: new developments in form, mechanics, perception and interaction design. In *Proceedings of the ACM/IEEE international conference on Human-robot interaction*, ACM, New York, NY, USA, HRI '07, 97–104.

XAUDIO2, M. 2011. XAudio2.