Rigid-Body Fracture Sound with Precomputed Soundbanks

Changxi Zheng Doug L. James

Cornell University

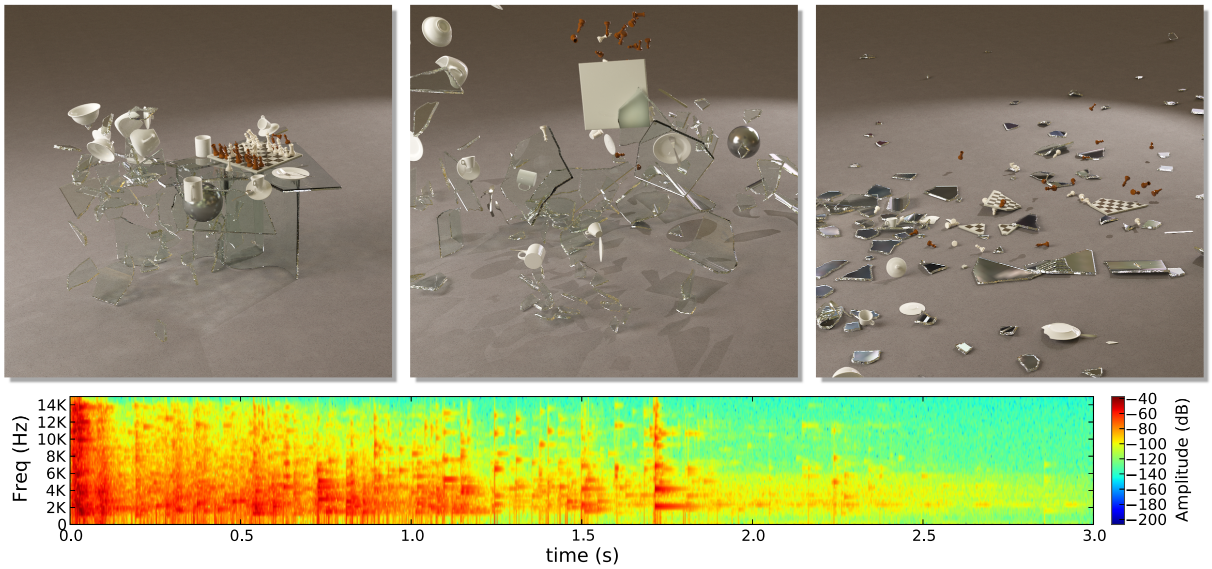

Figure 1: SMASH! *We synthesize the violent fracture and impact sounds of a glass table setting smashed into over 300 pieces (see sound spectrogram). We use time-varying rigid-body sound models to approximate this brittle fracture sound by a superposition of 4046 modal vibrations (up to $14kHz$). To avoid thousand-mode modal analysis and acoustic transfer costs for complex fracture geometry, we use sound proxies sampled from Precomputed Rigid-Body Soundbanks, here producing plausible fracture sound models at almost 500× speedup.*

Abstract

We propose a physically based algorithm for synthesizing sounds synchronized with brittle fracture animations. Motivated by laboratory experiments, we approximate brittle fracture sounds using time-varying rigid-body sound models. We extend methods for fracturing rigid materials by proposing a fast quasistatic stress solver to resolve near-audio-rate fracture events, energy-based fracture pattern modeling and estimation of "crack"-related fracture impulses. Multipole radiation models provide scalable sound radiation for complex debris and level of detail control. To reduce sound-model generation costs for complex fracture debris, we propose Precomputed Rigid-Body Soundbanks comprised of precomputed ellipsoidal sound proxies. Examples and experiments are presented that demonstrate plausible and affordable brittle fracture sounds.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Physically based modeling; I.6.8 [Simulation and Modeling]: Types of Simulation—Animation; H.5.5 [Information Systems]: Information Interfaces and Presentation—Sound and Music Computing

ACM Reference Format
Zheng, C., James, D. 2010. Rigid-Body Fracture Sound with Precomputed Soundbanks.
ACM Trans. Graph. 29, 4, Article 69 (July 2010), 13 pages. DOI = 10.1145/1778765.1778806
http://doi.acm.org/10.1145/1778765.1778806.

1 Introduction

Brittle fracture is an important and loud part of physically based animation and interactive virtual environments. Unfortunately we do not know how to procedurally synthesize fracture sounds automatically and efficiently. Current sound production methods rely instead on audio recordings of fracture events, which can inherit shortcomings of implausibility, lack of physical synchronization, and large memory footprints to avoid repetition.

In this paper, we propose the first physically based approach for automatic synthesis of synchronized brittle fracture sounds for computer animation (see Figure 1). Despite the familiar complexity of 3D fracture animation, our fracture sound synthesis method inherits the simplicity of rigid-body sound synthesis. Based on observations from laboratory fracture experiments with high-speed video and sound recordings (see Figure 2), we hypothesize that brittle fracture sounds can be efficiently and effectively approximated by time-varying rigid-body sound models (see Figure 3). Our rigid-body fracture sound synthesis has three parts: (1) a fracture preprocess which generates rigid-bodies with contact and "crack"-related fracture impulses; (2) a parallel sound model generation phase consisting of modal and acoustic transfer analysis; and (3) a sound synthesis phase where sounds are rendered at the listener's position.

We leverage prior work on fracturing rigid materials [Bao et al. 2007], and propose a sparse, direct, least-squares solver for the rank-deficient elastostatic problem ($\mathbf{Ku} = \mathbf{f}$) to resolve fracture sound events at near audio rates. An energy-based fracture model is used to model plausible fracture energy and sound generation, and also to estimate stress-based fracture impulses which excite the initially silent sound models of rigid-body debris to produce characteristically loud "crack" sounds (see Figure 2). The fracture simu-

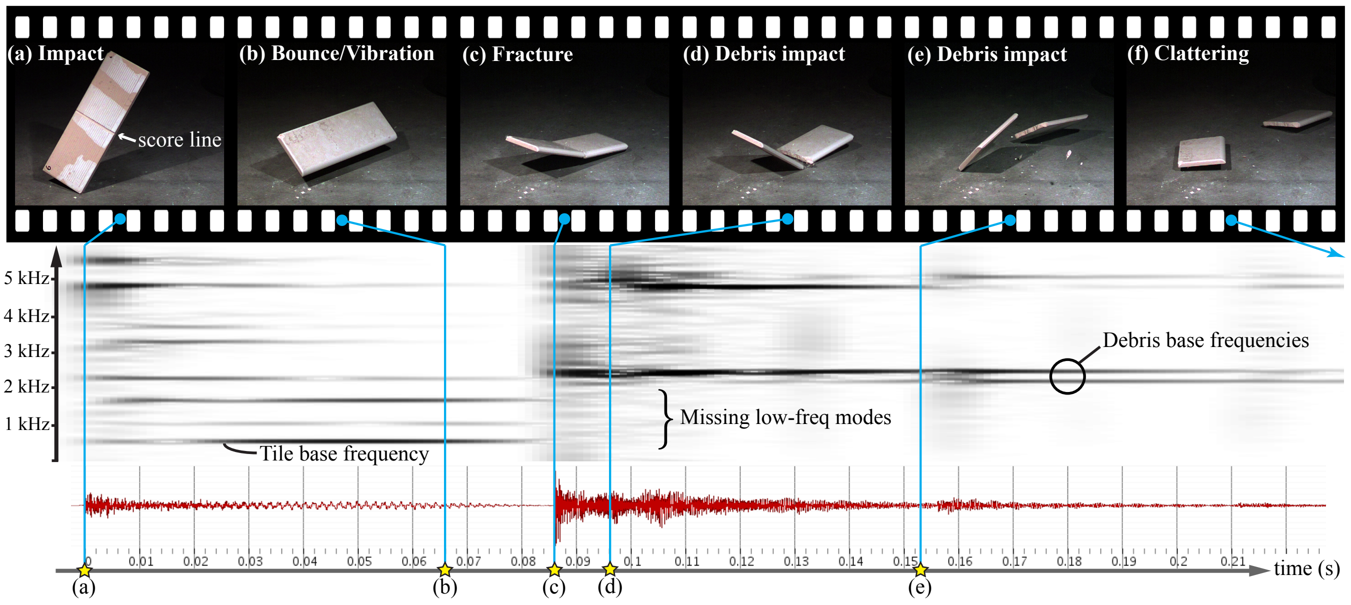

Figure 2: Laboratory experiments *reveal the time-varying modal sound structure of brittle fracture events, as shown by (Top) high-speed video at 1200 Hz, (Bottom) 96 kHz sound recordings, and (Middle) frequency spectrograms (2048 bins & 93% overlap for frequency clarity). In this experiment, a pre-scored ceramic tile dropped from two feet onto a concrete floor (a) impacts without breaking; (b) bounces and vibrates with the spectrogram showing distinct tile vibration frequencies; (c) a second impact fractures the tile halfway, and produces louder vibrations with higher relative half-tile frequencies; (d-f) additional post-fracture collision events further excite the half-tile frequencies.*

lation's fracture and contact impulses are used to excite each rigid-body's modal sound model. Frequency-domain acoustic transfer functions are computed for all vibration modes, and represented using high-order Helmholtz multipole expansions for efficient on-the-fly model generation and level of detail control. Real-time auralization and visualization of fracture simulations is possible, with GPU-accelerated transfer evaluation.

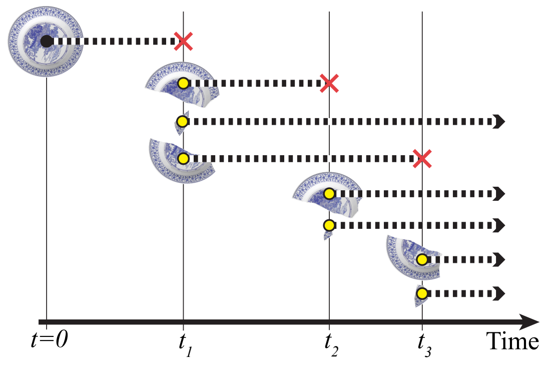

Figure 3: Time-varying rigid-body sound models *are used to approximate fracture sounds. Discrete fracture events result in spectral discontinuities due to modal sound model destruction (✕) and creation (●). Fracture impulses excite created sound models.*

Unfortunately significant sound-model generation costs (due to modal analysis and acoustic transfer) can be a bottleneck for violent fracture processes generating hundreds of rigid-body sound models. Sadly, many small debris sounds are difficult to discern, and can be masked by larger objects or loud fracture events. We propose Precomputed Rigid-Body Soundbanks to efficiently replace many rigid-body sound models with simpler ellipsoidal sound proxies. We sample the space of material-specific ellipsoidal sound models, exploiting fundamental scale dependencies to reduce the problem to a 2D lookup table. During sound synthesis, any rigid-body can retrieve its precomputed ellipsoid proxy (indexed by its inertia matrix), and scale it to have matching fundamental frequency. Rigid-body fracture and contact impulses (from the correct geometry) are then applied to the ellipsoid proxy to produce sound. Plausible fracture sounds involving hundreds of rigid bodies can be synthesized at a fraction of the cost (see Figure 1).

2 Related Work

Sound synthesis from physically based animations has been addressed for rigid objects [van den Doel et al. 2001; O'Brien et al. 2002b; James et al. 2006], deformable objects [O'Brien et al. 2001], and vortex- and bubble-based fluid sounds [Dobashi et al. 2003; Zheng and James 2009]. However, to the best of our knowledge, no prior work has addressed physically based sound synthesis directly from 3D brittle fracture animations. Most fracture sounds in games or computer-animated movies use data-driven approaches based on pre-recorded fracture sound effects (e.g., see [Parker and O'Brien 2009]), with fracture events being natural candidates for event-based sound rendering [Takala and Hahn 1992]. Unfortunately such approaches have the usual limitations of lacking synchronization, physical correctness, variability and/or requiring large sound file databases. To improve the effectiveness of recorded sounds, Picard et al. [2009] recorded, analyzed, and resynthesized impact and breaking sounds using granular synthesis techniques. Plausible contact sound clips were generated for a rigid-body simulator, however no visual or acoustic simulations of 3D fracture processes were considered. The physical characteristics of breaking sounds are discussed briefly by Rath and Fontana [Rocchesso and Fontana 2003]. In early psychophysical experiments, Warren and Verbrugge [1984] explored bouncing and breaking sounds of a glass bottle, and the ability of listeners to correctly identify between the two stimuli. They also manually synthesized plausible sound clips by identifying and editing impact/fracture sound events. Fracture sound recordings can exploit the inability of listeners to tell which fracture sound came from which fracture event, but lack variability and synchronization for more complex scenarios. In contrast, we maintain synchronization with object fracture and collision events, while exploiting the limited ability of listeners to tell apart the modal sound model of some debris from that of its representative ellipsoid model.

The visual simulation of fracture has a long history in computer animation [Terzopoulos and Fleischer 1988; Norton et al. 1991]. O'Brien and Hodgins [1999] animated brittle fracture, avoiding mesh aliasing artifacts with tetrahedral remeshing of crack propagation through an explicitly integrated finite element model. To avoid time-step restrictions associated with fast acoustic waves in stiff brittle materials, Smith et al.[2000] proposed a constraint-based fracture approach, but lacked sub-element fracture. To address remeshing challenges and time-step restrictions due to sliver elements, the virtual node algorithm [Molino et al. 2004] and meshless methods have been proposed [Pauly et al. 2005]. None of these approaches address brittle fracture sounds, however most of these time-domain methods could, in principle, be modified to apply contact and fracture impulses to our rigid-body sound models.

Given our rigid-body-fracture-sound hypothesis, instantaneous fracture models based on rigid bodies and quasistatic stress analysis are more convenient computational models for sound synthesis. Müller et al. [2001] modeled brittle fracture using an approximate quasistatic stress analysis on a tetrahedra mesh in contact regions of interest. This method was extended by Bao et al. [2007] to fracture rigid materials (including shells) while eliminating the quasistatic stiffness matrix's null space to obtain better stress estimates. Neither of these papers estimate fracture impulses explicitly, in part since they do not model fracture sounds. Our fracture patterns are based on Voronoi-based fracture pattern generation methods [Raghavachary 2002; Bao et al. 2007; Hellrung et al. 2009].

We approximate brittle fracture sounds using time-varying rigid-body sound models based on linear modal sound synthesis [van den Doel and Pai 1996]. Modal models have a long history in animation [Pentland and Williams 1989] and in physically informed sonic modeling of complex multi-impact, percussive, and stochastic sound sources [Cook 1997]. We precompute linear modal analyses using finite element models [O'Brien et al. 2002b; Raghuvanshi and Lin 2006; James et al. 2006], since it is more efficient for rigid-body sound than more general time-domain vibration analyses [O'Brien et al. 2001]. Numerous simplified (modal) sound models have been proposed for rigid body contact. Hahn et al. [1998] introduced *timbre trees* to model parameterized sound models. A popular approach is to estimate modal sound model parameters from impact recordings, and to texture modal sound models onto virtual objects [Pai et al. 2001; van den Doel et al. 2001], thereby avoiding modal analysis. Mode culling techniques are often used to reduce modal synthesis costs for real-time applications [Doel et al. 2004; Raghuvanshi and Lin 2006], but synthesis costs are not a bottleneck in our simulation, and it is difficult to pre-cull modes correctly to avoid modal analysis and acoustic transfer costs. In contrast, we precompute rigid-body soundbanks with acoustic transfer for high-quality 3D sound. Methods exist for auditory culling and spatial level-of-detail for hundreds of moving sources [Tsingos et al. 2004], but would not avoid the fracture sound model computation. Frequency-domain methods can accelerate modal integration (albeit with some impact sound degradation) [Bonneel et al. 2008], but that is a negligible cost in fracture sound synthesis.

For improved sound quality, we use frequency-domain acoustic transfer. Special attention is needed to deal with the large numbers of dynamically created and destroyed debris objects, for which precomputation and reuse is limited. In Precomputed Acoustic Transfer [James et al. 2006], acoustic transfer functions are precomputed and efficiently represented using multi-point multipole expansions [Ochmann 1995], but the expensive model-specific precomputation and lack of level-of-detail control make them undesirable for fracture sound processing. Chadwick et al. [2009] introduced Far-Field Acoustic Transfer (FFAT) Maps to achieve $O(1)$ acoustic transfer evaluation costs per mode, but their extensive precomputation and memory requirements are undesirable for on-the-fly model generation—but could be used for precomputed soundbanks. In our implementation, we exploit traditional single-point multipole expansions with a simple precomputation step, and exploit GPU transfer evaluation for interactive performance.

Acoustic emission (AE) of solid-borne acoustic waves due to rapid micro-crack growth in brittle materials has been studied extensively in engineering [Dunegan et al. 1968; Lockner et al. 1991]. AE methods are important for understanding numerous fracture processes, including those in rocks for earthquake, landslide, and rock burst monitoring [Lockner 1993]; micro-fracture sounds have been known for thousands of years to be important defect indicators for clay pottery; and AE testing is widely used to analyze the integrity of man-made structures [Grosse and Ohtsu 2008]. Brittle fracture sounds have also been studied empirically in the quest for crunchier and crispier foods [Luyten and Vliet 2006]. Numerical methods for simulating elastic waves due to incremental crack growth also exist; for example, time-domain AE simulation has been conducted extensively in seismology [Carpinteri and Lacidogna 2007], and also for fracturing sea ice, e.g., using a 2D particle system model [Li and Bazant 1998]. Unfortunately these brute-force time-domain methods are impractical for brittle fracture sound synthesis.

3 Rigid Fracture Simulation

Our sound synthesis pipeline uses rigid-body fracture simulation to generate and animate fracture debris. Estimated contact and fracture impulses are used to excite rigid-body sound models—readers not familiar with rigid-body sound can find background on modal analysis and acoustic transfer in Appendix A. Our fracture simulator is closely related to the method for fracturing rigid materials of Bao et al. [2007]. We also use an impulse-based rigid-body simulator [Guendelman et al. 2003] since contact iteration costs amortize well over small near-audio-rate timesteps, e.g., $\Delta t = 1/10000s$ to $1/5000s$ in our examples. We propose three extensions for rigid-fracture sound synthesis: (1) a sparse, direct, least-squares solver for fast quasistatic stress analysis at high near-audio rates, (2) an energy-based method for generating plausible fracture patterns, which we used to (3) estimate stress-based fracture impulses that produce fracture-related "crack" sounds.

3.1 Fast Quasistatic Stress Analysis

Quasistatic stress analysis is commonly used to approximate brittle fracture [Müller et al. 2001; Bao et al. 2007]. Given an N node tetrahedral mesh, it involves solving the elastostatic equation

$$\mathbf{Ku} = \mathbf{f}, \tag{1}$$

for the quasistatic displacement, $u \in \mathbb{R}^{3N}$, resulting from external contact forces, $\mathbf{f}$, in order to compute element stresses using standard methods [Bonet and Wood 1997]—we also compute the elastic strain energy, E_S, for energy-based fracture (§3.2). In contrast to computer animation where (1) is solved near or at graphics rates [Müller et al. 2001; Bao et al. 2007; Su et al. 2009], for fracture sound synthesis it is desirable to timestep the system at much higher rates, e.g., 5 kHz, to resolve micro-collisions and fracture events. In practice, large fracture simulations can request solves for hundreds of thousands of micro-impact problems.

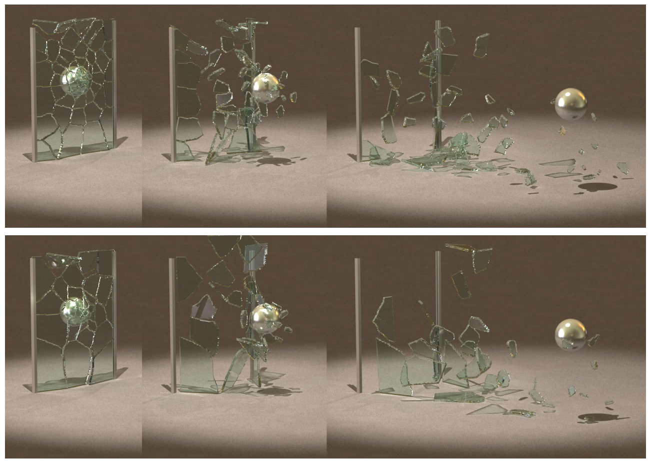

Figure 4: Fracture toughness dependent sound: *(Top) A window simulated with low fracture toughness ($G_c = 50\ J/m^2$) produces smaller debris with overall higher pitch than (Bottom) a window with higher fracture toughness ($G_c = 120\ J/m^2$).*

It is well known that $\mathbf{K}$ is sparse, symmetric, and also rank deficient: its rank is always $3N - 6$, due to a rank-6 null space associated with translation and linearized rotation of the unconstrained rigid body. To calculate the linearized quasistatic stress distribution, we can either compute the min-norm least-squares solution to (1), or just solve the least-residual problem (1) since the rigid component of $\mathbf{u}$ (or $\mathbf{f}$) produces no stress. The iterative MINRES algorithm [Paige and Saunders 1975] can solve the least-residual problem, but suffers from slow convergence. Müller et al. [2001] approximated it by anchoring a number of points in the objects, thereby enforcing extraneous constraints that break momentum conservation. Bao et al. [2007] solved (1) using a modified Conjugate Gradient method with an additional projection at each iteration to remove the null space. Unfortunately this iterative method converges slower than we would like, in part due to the difficulty preconditioning the rank-deficient $\mathbf{K}$ matrix.

We propose a sparse, direct least-squares solver that exploits temporal coherence, and is faster and more robust for sound applications. After a one-time setup/factorization cost per object, solutions can be obtained essentially via back substitution. We briefly sketch the method here, and defer solver details (on $\mathcal{P}$ and $\mathbf{V}$) to Appendix B. First, we project out the linearized rigid-body motion using an orthogonal projection $\mathcal{P}$ to ensure that the system is compatible, i.e., that $\mathcal{P}\mathbf{f} \in \mathsf{range}(\mathbf{K})$. Next we construct a special $3N \times (3N - 6)$ sparse orthogonal matrix $\mathbf{V}$, then premultiply (1) by $\mathbf{V}^T$ and substitute $\mathbf{u} = \mathbf{V}\mathbf{r}$, to obtain a sparse, symmetric and *full-rank* $(3N-6) \times (3N-6)$ system which can be solved directly with Cholesky factorization [Golub and Van Loan 1996]:

$$\mathbf{V}^T\mathbf{K}\mathbf{V}\,\mathbf{r} = \mathbf{V}^T\mathcal{P}\mathbf{f}. \tag{2}$$

Given that $\mathbf{V}\mathbf{r}$ is a least-residual solution of (1) for compatible RHS (see Appendix C for a proof), we simply project out the particular translation and rotation chosen by $\mathbf{V}$, to obtain the min-norm least-squares displacement solution, $\mathbf{u}^* = \mathcal{P}\mathbf{V}\mathbf{r}$. In summary, each time a rigid object is created we compute and cache its sparse Cholesky factorization, $\mathbf{L}\mathbf{L}^T = \mathbf{V}^T\mathbf{K}\mathbf{V}$ and data for $\mathcal{P}$. For each simulation timestep with nonzero contact forces, $\mathbf{f}$, we evaluate the least-squares quasistatic displacement incrementally (from right to left),

$$\mathbf{u}^* = \mathcal{P}\mathbf{V}(\mathbf{V}^T\mathbf{K}\mathbf{V})^{-1}\mathbf{V}^T\mathcal{P}\,\mathbf{f}, \tag{3}$$

with the primary cost being the Cholesky backsubstitution for $(\mathbf{V}^T\mathbf{K}\mathbf{V})^{-1}$; for our examples, one-time factorization costs were 0.18s–1.30s, whereas (3) solves cost only 0.007s–0.23s.

3.2 Energy-based Debris Generation

We use an energy-based method for generating plausible fracture patterns, with energy also used later for plausible sound generation (§3.3). To determine when brittle fracture occurs, we use the *Rankine hypothesis* [Gross and Seeling 2006] wherein material breaks when any principal stress value exceeds a given threshold. We use a Voronoi-based fracture pattern method similar to [Raghavachary 2002; Bao et al. 2007], however, in contrast, we incrementally construct the fracture pattern to maintain bounded fracture energy. We estimate the energy required to generate fracture surfaces of total area A_F by $\boxed{E_F = G_c\,A_F}$, where G_c is the *fracture toughness* material parameter (e.g., the *c*ritical strain energy release rate for mode-I fracture) which describes the material's ability to resist fracture; G_c values used in this paper are given in Table 1. To avoid excessive fracturing and uncontrolled sound generation, we require the consumed fracture energy E_F to be less than the quasistatic strain energy, $\boxed{E_F \le \eta\,E_S}$ where the parameter $\eta \in (0, 1)$ controls how much strain energy is converted into fracture energy; we use $\eta = 0.8$ in our examples. The impact of fracture toughness on sound generation is illustrated in Figure 4.

We generate Voronoi-like fracture patterns by incrementally sampling region "seed" points $\boldsymbol{p}_i$ (with probability proportional to strain energy density) then using a region-growing strategy. Each unassigned tetrahedral node $\boldsymbol{x}$ has prority-queue values for each region i given by the weighted distance $k_i\,\mathsf{dist}(\boldsymbol{x}, \boldsymbol{p}_i)$ where k_i is the strain energy density at $\boldsymbol{p}_i$ raised to the power $\alpha > 0$; we use $\alpha = 0.15$. Tetrahedral nodes are captured by adjacent regions with minimal queue values, and tend to produce smaller pieces in regions of high strain energy density. To ensure regions are connected, we use geodesic distances with edge-based approximations computed with Dijkstra's algorithm. After each point insertion, we estimate E_F, then continue adding points until $E_F \le \eta\,E_S$ can not be satisfied. Multi-region elements are split [Bielser et al. 2004]. Fine region meshes are generated for rendering, modal analysis and Helmholtz sound radiation analysis; our mesh edge lengths h satisfy a 20 kHz sampling condition, $h < \lambda/6 \approx 5mm$ [Liu 2009].

3.3 Fracture Impulse Estimation

Previous rigid-body fracture simulations in computer animation ignore stress-based impulses introduced by fracture events, relying instead on contact forces to push debris apart [Müller et al. 2001; Bao et al. 2007]. While this can be sufficient for animation, these additional "fracture impulses" can be important contributors to debris sound. For example, snapping a candy cane in two can produce audible fracture sound even when the pieces separate cleanly without subsequent contact. We propose an energy-based model to estimate impulses from fracture events so that (1) explosive effects are introduced by the strain energy release, and (2) impulses excite the modal sound models of the initially silent debris (see Figure 5).

Our fracture-impulse model assumes that unused quasistatic strain energy is converted into kinetic energy $\boxed{\Delta E_K = E_S - E_F}$ and introduced by stress-based fracture impulses. For each fracture-surface triangle i, the exerted stress force $\boldsymbol{s}_i$ and torque $\boldsymbol{j}_i$ are

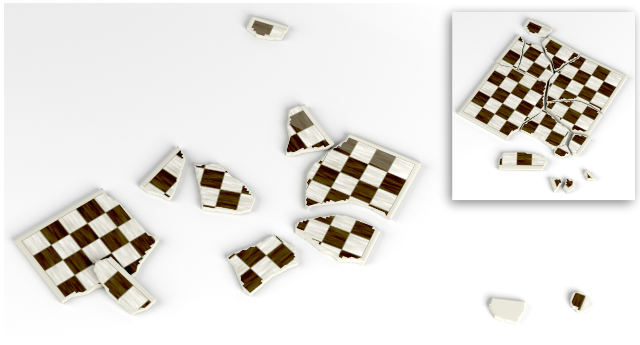

Figure 5: Bigger "cracks" with fracture impulses: *Our quasistatic fracture impulses produce more explosive fractures, thereby producing louder and more characteristic "crack" sounds. In contrast, simulations without fracture impulses applied (see inset) lack fracture-released kinetic energy. This ceramic chessboard ($38cm \times 38cm$) was dropped from one meter high onto the ground.*

$$\boldsymbol{s}_i = a_i \boldsymbol{P}_i \boldsymbol{n}_i; \quad \boldsymbol{j}_i = \boldsymbol{r}_i \times a_i \boldsymbol{P}_i \boldsymbol{n}_i \tag{4}$$

where a_i is the triangle area, $\boldsymbol{n}_i$ is unit direction perpendicular to the triangle, $\boldsymbol{r}_i$ is the distance vector from the object's center of mass to the center of the questioned triangle, and $\boldsymbol{P}$ is the Piola-Kirchhoff stress tensor of the tetrahedron on which the triangle is contained [Bonet and Wood 1997].

Let τ denote the effective duration to exert fracture forces (4). Let $\boldsymbol{v}_d^-$ and ω_d^- respectively denote the pre-fracture linear and angular velocity of the material associated with the d-th piece of debris. The post-fracture linear and angular velocity are

$$\boldsymbol{v}_d^+ = \boldsymbol{v}_d^- + \frac{\tau}{m_d} \sum_{\text{triangle } i} \boldsymbol{s}_i; \quad \omega_d^+ = \omega_d^- + \tau \mathbf{I}_d^{-1} \sum_{\text{triangle } i} \boldsymbol{j}_i, \tag{5}$$

where m_d and $\mathbf{I}_d$ are the d^{th} rigid body's total mass and angular inertia, respectively. We estimate τ by equating ΔE_K with the system's change in kinetic energy,

$$\Delta E_K = \frac{1}{2} \sum_{\text{debris } d} \left(m_d \|\boldsymbol{v}_d^+\|^2 - m_d \|\boldsymbol{v}_d^-\|^2 + (\boldsymbol{\omega}_d^+)^T \mathbf{I}_d \boldsymbol{\omega}_d^+ - (\boldsymbol{\omega}_d^-)^T \mathbf{I}_d \boldsymbol{\omega}_d^- \right)$$

which is a quadratic equation in τ, where only the smallest positive solution is physically relevant. Each time an object is fractured, we solve for τ and apply the fracture impulses to the debris.

4 Parallel All-Frequency Multipole Sound

We approximate each object's acoustic transfer function, $p(\boldsymbol{x})$, using a single-point multipole expansion for reliable estimates of far-field sound [Gumerov and Duraiswami 2004]. This simple representation also has three major benefits: (1) runtime level of detail control for complex fracture sound simulations (unlike [James et al. 2006]), (2) an efficient parallel GPU implementation, and (3) convenient support for our Rigid-Body Soundbanks (in §5) to enable scalable fracture sound synthesis.

Multipole Radiation Model: The spherical multipole wave expansion of each mode's $p(\boldsymbol{x})$ takes the form

$$p(\boldsymbol{x}) \approx \sum_{n=0}^{\bar{n}} \sum_{m=-n}^{n} S_n^m(\boldsymbol{x} - \boldsymbol{x}_0)\, M_n^m \tag{6}$$

where $M_n^m \in \mathbb{C}$ are multipole expansion coefficients, and S_n^m are multipole basis functions (singular, radiating solutions to the Helmholtz equation),

$$S_n^m(\boldsymbol{r}) = h_n^{(2)}(kr)\, Y_n^m(\theta, \phi), \tag{7}$$

where (r, θ, ϕ) are spherical coordinates of $\boldsymbol{r}$; $h_n^{(2)} \in \mathbb{C}$ are spherical Hankel functions of the second kind; and $Y_n^m \in \mathbb{C}$ are spherical harmonics. We can precompute the multipole coefficients, M_n^m, for each object since they are independent of listening position $\boldsymbol{x}$. At runtime, only $(\bar{n}+1)^2$ summation terms need be computed, which is independent of the object's geometric complexity.

In our implementation, we place the multipole expansion point, $\boldsymbol{x}_0$, at the object's center of mass. Since the distance between the listener's position and the center of mass, $\|\boldsymbol{x} - \boldsymbol{x}_0\|$, is much larger than the object's diameter, the series approximation typically converges quickly to the accurate transfer function. However, convergence is frequency dependent with higher $\bar{n}$ required at higher frequencies; we use the empirical formula, $\bar{n} = \max(\frac{1}{4}kL, 4)$ [Liu 2009], where the length scale, L, is the object diameter; fewer terms can be used to reduce transfer-evaluation costs.

Multipole Coefficient Solver: The M_n^m coefficients can be precomputed in various ways. For example, source simulation or equivalent source methods [Ochmann 1995; James et al. 2006] directly estimate M_n^m by requiring that (6) matches the Neumann boundary condition in a least squares sense (note that this requires that $\boldsymbol{x}_0$ lies inside the object). In our implementation we use the fast multipole boundary element method to first solve the associated Helmholtz boundary integral problem, and then evaluate the multipole coefficients via their integral representations; we refer the reader to Appendix D for implementation details. We exploit mode-level parallelism by computing transfer models on a cluster.

Parallel Sound Evaluation: Evaluating the sound pressure from all modes of all objects involves significant transfer evaluation costs, but is a pleasantly parallel computation. The sound contribution from an object's vibration mode, i, is estimated using $p_i(\boldsymbol{x}, t) = |p_i(\boldsymbol{x})|\, q_i(t)$. The total fracture sound is the superposition of all object mode contributions:

$$\begin{aligned} \text{sound}(\boldsymbol{x}, t) &= \sum_{\text{mode } i} |p_i(\boldsymbol{x})|\, q_i(t) \\ &= \sum_{\text{mode } i} \left| \sum_{n=0}^{\bar{n}_i} \sum_{m=-n}^{n} S_n^m(\boldsymbol{x} - \boldsymbol{x}_{0i}; i)\, M_n^m(i) \right| q_i(t) \end{aligned} \tag{8}$$

where $M_n^m(i)$ are precomputed for mode i using (22). In our parallel implementation, we evaluate $\{p_i(\boldsymbol{x})\}$ on the GPU (using NVIDIA's CUDA API) sampled in time at high graphics rates (200 Hz), then copy it back to the CPU, and (optional) apply an HRTF filter. Modal amplitudes $q_i(t)$ are computed using an IIR filter in parallel on multi-core CPUs at audio sample rates (although GPU implementations are possible [Trebien and Oliveira 2009]), then each q_i is multiplied by $|p_i(\boldsymbol{x})|$ (with p interpolated up to audio rates) to obtain the mode's sound contribution, $|p_i(\boldsymbol{x})| q_i(t)$. The GPU evaluation of $p_i(\boldsymbol{x})$ values exploits thread-level parallelism across modes (i), and in p_i's (m, n)-summation using parallel reduction [Harris 2007]. Furthermore, all modes' multipole basis functions, $S_n^m = h_n^{(2)}(kr) Y_n^m(\theta, \phi)$ are first computed in parallel using a multi-pass GPU algorithm:

- In a first pass, one thread per mode computes the $h_n^{(2)}(kr)$ values for $n = 0 \ldots \bar{n}_i$ (with $kr = k_i r_i$) using the recurrence relation, $h_{n+1}^{(2)}(kr) = \frac{2n+1}{kr} h_n^{(2)}(kr) - h_{n-1}^{(2)}(kr)$.

- Next $Y_n^m(\theta,\phi)$ is computed in two passes: (1) one thread per mode computes $Y_m^m(\theta,\phi), m=0\dots p_i$ using the recurrence relation between Y_m^m and Y_{m-1}^{m-1} [Press et al. 2007], then (2) one thread per mode per m value computes $Y_n^m, n=m+1\dots\bar{n}_i$ using the recurrence relation between Y_n^m and Y_{n-1}^m.

5 Precomputed Rigid-Body Soundbanks

Complex fracture sounds can involve very large numbers of rigid-body sound sources, each of which requires computing an expensive, object-specific sound model. For complex fracture scenarios, the simulation bottleneck quickly becomes rigid-body sound model generation (modal analysis, and multipole radiation analysis). Unfortunately the opportunities for precomputation are limited by the unpredictable nature of fractured geometry generation.

Ironically, for complex fracture scenarios, it can be difficult to fully discern each individual rigid-body sound, and sounds produced by small objects can be masked partially by large pieces of debris. We exploit this perceptual ambiguity by augmenting debris with simple precomputed sound models of similar frequency content. Specifically, we propose to use ellipsoid-shaped sound proxies during sound synthesis (see Figure 6) since (i) ellipsoids provide the smoothest shape matching the rigid-body inertial mass, and (ii) the smooth surface allows us to parameterize the proxy contact location using contact normal. While the rigid-body fracture simulation and contact impulses are based on the original fractured geometry, external forces can be applied to the proxy's sound model, thereby avoiding the bottleneck of model-specific sound model generation.

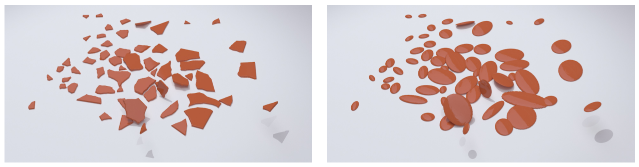
Figure 6: Fracture debris and ellipsoidal sound proxies

For each material we precompute a Rigid-Body Soundbank, where each sound model has ellipsoidal geometry, mode shapes $\mathbf{U}$, frequencies $\boldsymbol{\omega}$, and precomputed multipole coefficients M_n^m. To generate object sounds, we first retrieve its closest soundbank proxy based on an inertia tensor metric, and scale it to match our object's base frequency. Next we map the external forces to the proxy, load precomputed eigendata and multipole M_n^m values, and synthesize the sound at the relative position. This process is illustrated in Figure 7, and summarized in Algorithm 1.

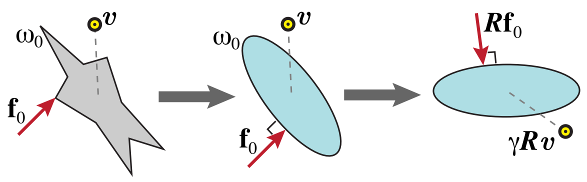

Figure 7: Ellipsoidal sound proxies *are indexed by each object's inertia matrix, and scaled γ to have matching base frequency, ω_0. Contact forces $\mathbf{f}_0$ and relative listening positions $\boldsymbol{v}$ are rotated $\boldsymbol{R}$ to the precomputed proxy frame for sound synthesis.*

5.1 Exploiting Scale Dependence

Naïve sampling of sound models over the 3D space of ellipsoidal shapes will lead to significant memory requirements and/or poor shape resolution. We therefore exploit the fact that uniformly scaling a rigid-body model does not fundamentally alter its modal vibration and multipole radiation models, but only induces power-law scalings. By precomputing a sound model for one rigid-body shape, we obtain sound models for all scaled versions. In particular, if the geometry of an object is scaled by γ, then the following scalings result (see Appendix E for derivations):

Algorithm 1: **Runtime use of Rigid-Body Soundbanks**

```
input: The soundbank S; external forces $\boldsymbol{f}$
foreach sounding object obj do
    if use_proxy(obj) then                                   // §5.4
        $\boldsymbol{a}$ ←find_best_proxy(obj,S)                    // §5.4
        $\gamma$ ←scale_factor(obj,$\boldsymbol{a}$)                       // §5.4
        $\boldsymbol{R}$ ←rotation(obj,$\boldsymbol{a}$)                          // §5.2
        $\boldsymbol{f}'$ ←map_forces($\gamma$,$\boldsymbol{R}$,$\boldsymbol{f}$)                    // §5.2
        $(\boldsymbol{\omega},\mathbf{U})$ ←load_eigen($\boldsymbol{a}$,S)
        $(\boldsymbol{\omega}',\mathbf{U}')$ ←scale_eigen($\boldsymbol{\omega}$,$\mathbf{U}$,$\gamma$)           // §5.1
        $M$ ←load_moments($\boldsymbol{a}$,S)
        $M'$ ←scale_moments($M$, $\gamma$)                      // §5.1
        generate_proxy_sound($\boldsymbol{\omega}$, $\mathbf{U}'$,$M'$,$\boldsymbol{f}'$)
    else
        directly_generate_sound(obj)
    end
end
```

$$\begin{array}{cccccc} \boldsymbol{x} & \omega & k\boldsymbol{x} & \alpha+\beta\omega^2 & \mathbf{U} & M_n^m \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ \gamma\boldsymbol{x} & \gamma^{-1}\omega & k\boldsymbol{x} & \alpha+\beta\gamma^{-2}\omega^2 & \gamma^{-3/2}\mathbf{U} & \gamma^{-5/2}M_n^m \end{array} \tag{9}$$

In order to guarantee all-frequency sound up to a high-frequency cutoff, e.g., 16 kHz, we note that shrinking models ($\gamma<1$) increases their frequency range since $[\omega_{min},\omega_{max}]\to[\frac{\omega_{min}}{\gamma},\frac{\omega_{max}}{\gamma}]$, and therefore will preserve the all-frequency property. In contrast, enlarging ($\gamma>1$) may not since $\frac{\omega_{max}}{\gamma}$ may drop below the high-frequency cutoff. We therefore precompute all-frequency models which are as large as needed to ensure $\gamma<1$.

5.2 Ellipsoidal Sound Proxies

Normalized ellipsoids: Our soundbank is based on precomputing sound models for axis-aligned ellipsoids,

$$\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}=1, \tag{10}$$

where the ellipsoid's shape is parameterized by the lengths of the principal axes, $\hat{\boldsymbol{a}}=(a,b,c)^T$. To avoid sampling this three-dimensional (a,b,c) parameter space, we exploit scaling dependences (§5.1) to eliminate scale via the normalization,

$$\|\hat{\boldsymbol{a}}\|_2=1\,meter \quad\Leftrightarrow\quad a^2+b^2+c^2=1, \tag{11}$$

effectively reducing the dimensionality from three to two. Furthermore, we can assume that $a\le b\le c$, to reduce the parameter space to a small triangular patch on the unit sphere (see Figure 8(Left)). For each ellipsoid, we conduct the modal analysis, and solve the boundary integral problems to precompute M_n^m values; the eigenmodes, frequencies, and M_n^m values are stored in the soundbank.

Inertia-matrix parameterization: Given a rigid body's symmetric inertia matrix, $\mathbb{I}\in\mathbb{R}^{3\times3}$, we identify the rigid-body's ellipsoidal sound proxy using the principal moments of inertia. These are obtained from the eigenvalue decomposition, $\mathbb{I}=\boldsymbol{V}_I\,\boldsymbol{\Lambda}_I\,\boldsymbol{V}_I^T$, where

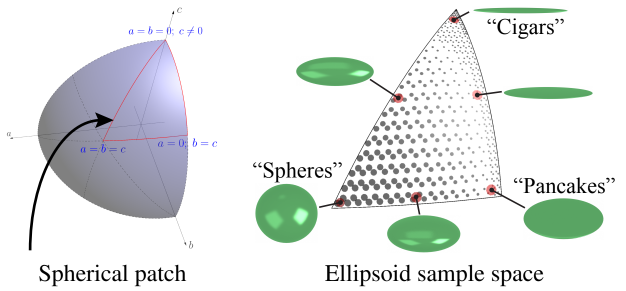

Figure 8: Ellipsoid sampling on the unit sphere

$\boldsymbol{V}_I$ are the orthogonal eigenvectors (specifying the principal axes of inertia), and the principal moments of inertia are the diagonal values of $\boldsymbol{\Lambda}_I = \mathrm{diag}(I_1, I_2, I_3)$, with $I_1 \geq I_2 \geq I_3$. The $\boldsymbol{\Lambda}_I$ are directly related to the normalized ellipsoid parameters, $\hat{\boldsymbol{a}}$: the corresponding unit ellipsoid proxy is parameterized by[1]

$$\hat{\boldsymbol{a}} = \frac{1}{\sqrt{I_1 + I_2 + I_3}} \begin{pmatrix} \sqrt{-I_1 + I_2 + I_3} \\ \sqrt{+I_1 - I_2 + I_3} \\ \sqrt{+I_1 + I_2 - I_3} \end{pmatrix} \in \mathbb{R}^3. \quad (12)$$

Proxy contact forces are estimated by mapping rigid-body contact forces to the proxy as follows. Let $\mathbf{f}_0$ denote an external rigid-body force defined in the object's material frame. The equivalent *proxy contact force* is $\mathbf{f}_p = \boldsymbol{R}\mathbf{f}_0$, where the precomputed rotation matrix $\boldsymbol{R} = \boldsymbol{V}_I^T$ maps forces from the object's material frame to the proxy's material frame. The *proxy contact point* is estimated uniquely by assuming that the contact force is applied in the normal direction, which is easily computed by solving a 3×3 linear equation (see Appendix F).

Proxy sound mapping: We evaluate each proxy's sound in its axis-aligned frame. Given the listening position vector in the rigid-body's center-of-mass frame, $\boldsymbol{v}$, we simply rotate and scale the vector into the proxy frame, $\boldsymbol{v} \rightarrow \gamma \boldsymbol{R} \boldsymbol{v}$, and evaluate the proxy sound there (see Figure 7).

5.3 Soundbank Sampling

In practice it is desirable to precompute and store as little soundbank information as possible. To avoid uniformly sampling the spherical patch in ellipsoid parameter space, we use a simple adaptive strategy to resolve faster modal frequency variations in ellipsoid samples near shape singularities at the "pancake" and "cigar" vertices (see Figure 8(Right)).

Material-specific Soundbanks: In theory, a different soundbank must be precomputed for each homogeneous material, e.g., for our glass and ceramic material parameters (see Table 1). Runtime adjustments can still be made to scale, and Rayleigh damping parameters, α and β. However, often plausible material parameters are similar, e.g., glass and ceramic, and we precompute a single "glass" soundbank and scale the models appropriately.

5.4 Proxy Retrieval and Scaling

Proxy use criterion: In practice, we use sound proxies for sufficiently small debris with base frequencies above a user-specified threshold, ω_{proxy}. Given a rigid-body candidate for proxy replacement, we first compute the lowest eigen-frequency ω_0 of the object. A rigid body will only use the proxy when $\omega_0 > \omega_{proxy}$ is satisfied. Note that this single-frequency ω_0 calculation is far cheaper than computing the object's modal sound model.

[1] This follows from the ellipsoid's principal moments of inertia, $I_1 = \frac{m}{5}(b^2 + c^2),\ I_2 = \frac{m}{5}(a^2 + c^2),\ I_3 = \frac{m}{5}(a^2 + b^2)$.

Material	Density	Young's mod.	Poisson	G_c
Glass	2600 kg/m^3	$6.2 \times 10^{10}\ GPa$	0.2	80 J/m^2
Ceramic	2700 kg/m^3	$7.2 \times 10^{10}\ GPa$	0.19	200 J/m^2

Table 1: *Material Parameters*

Proxy retrieval: Given a rigid body for proxy replacement, we first compute its normalized parameter vector, $\hat{\boldsymbol{a}}$, then select the soundbank ellipsoid, $\boldsymbol{a}'$, with the minimum Euclidean distance, $\|\boldsymbol{a}' - \hat{\boldsymbol{a}}\|_2$. We use a kd-tree to accelerate this closest-point query. Simulation-specific proxy values are shown in Figure 9.

Equi-frequency scaling: The retrieved ellipsoid proxy is unscaled. Instead of using inertia to scale the object, which can be a poor indicator of sound frequency, we select the proxy scale γ such that the proxy and rigid body have the same base eigen-frequency, ω_0, since it is an important perceptual attribute to match [McAdams et al. 2004]. Specifically, if the unscaled proxy's lowest eigen-frequency is ω_{p0}, then the desired scale is $\gamma = \omega_{p0}/\omega_0$. We then scale the proxy's mode shapes, frequencies, multipole coefficients, M_n^m, etc., as described in section 5.1.

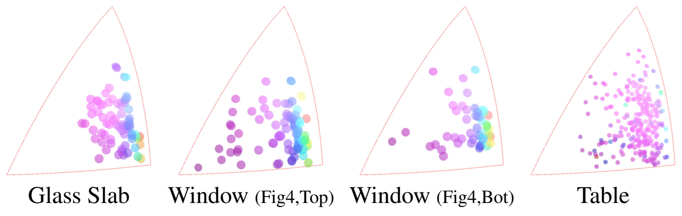

Figure 9: Soundbank access patterns *colored by object mass. Note the massive "pancake" shapes for thin glass debris.*

6 Results

For information on simulation cost versus fracture debris complexity, please see example statistics in Table 3. Representative stills are provided in Figure 10. Please see our video for all sound and animation results, including comparisons to reference laboratory fracture experiments (see Figure 11). Rigid-Body Soundbank performance improvements are described in Table 2.

Figure 11: Fracture experiments *were recorded using high-speed video (1200 FPS) and stereo audio recordings (96 kHz).*

Ground sound model: Ground vibrations can play an important role in sound generation when smashing objects onto it, especially given the ground's large size and wide frequency range. For example, a small piece of glass with extremely high pitch can produce

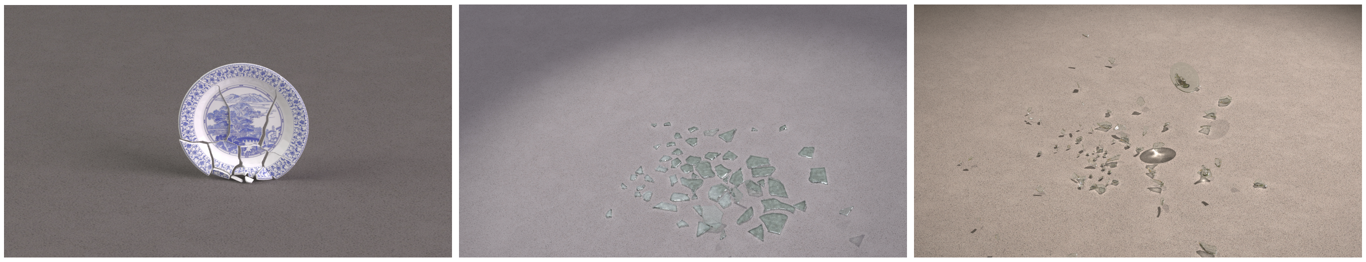

Figure 10: Fracture simulation images

Model	Complexity								Timings (min)			
	time	$1/\Delta t$	Objects	Triangles	Tetrahedra	Modes	Solves	freqRange	Frac	Modal*	Trnsfr*	Audio
Plate	5s	10kHz	1→17	100k→342k	404k→967k	76	91k	20-20k	1.41hr	19.8	8.4	1.2
Tile	0.9s	5 kHz	1→3	47k→109k	201k→419k	81	1.2k	20-20k	0.05hr	12.2	6.2	0.1
WindowHiG_c	5s	5 kHz	1→57	50k→287k	120k→467k	611	252k	20-20k	3.1hr	18.6	31.2	3.1
WindowLoG_c	5s	5 kHz	1→89	50k→344k	120k→528k	586	302k	20-20k	3.4hr	20.4	50.4	3.3
Glass Slab	2s	10kHz	1→72	50k→407k	121k→634k	377	237k	20-20k	3.6hr	24.7	38.4	2.4
Wine Glass	3.2s	10kHz	1→83	101k→674k	372k→1391k	57	538k	20-20k	4.7hr	40.6	55.2	4.6
Poor Piggy	3.2s	10kHz	17→64	405k→752k	1628k→2190k	240	384k	20-20k	6.4hr	33.6	49.2	8.2
Rich Piggy	2s	10kHz	301→358	6743k→7131k	27191k→27801k	1411	413k	20-20k	26.4hr	37.2	56.6	12
Table	5s	5 kHz	62→328	1972k→6048k	6816k→15781k	4046	1138k	20-14k	12.2hr	66.3	73.8	20

Table 3: Example Statistics *for simulation duration, rigid-body timestep size, and input and total geometric complexity; total number of scene modes; number of quastistatic stress solves. Serial timings are provided for fracture dynamics simulation and audio synthesis, whereas parallel timings ($*$) are given for modal analysis and acoustic transfer computations on a 16-node cluster of 8-core Xeons. Fortunately, most of the latter costs can be avoided using Rigid-Body Soundbanks (see Table 2).*

Example	ω_{proxy}	%Modes Replaced	CPU Sound Time (min)	
			Direct	With Proxies
Glass Slab	4000 Hz	36.4%	5631	3944
	2000 Hz	64.8%		2323
	0 Hz	100%		5.2
Window (low G_c)	3400 Hz	49.1%	4262	2202
	0 Hz	100%		9.2
Window (high G_c)	2000 Hz	45.2%	6023	3029
	0 Hz	100%		12.2
Table	2000 Hz	22.8%	17792	12980
	600 Hz	62.7%		7103
	140 Hz	95.7%		36.2

Table 2: Rigid-Body Soundbank Performance: *Increasing ellipsoid proxy use (by lowering ω_{proxy}) leads to dramatic reductions in expensive sound model generation costs (modal+transfer+audio) over direct computation (serial timings). By replacing all fragment sound models by proxies, sound differences were barely audible, and roughly 500× speedups were observed in some cases—limited by disk I/O in our implementation.*

low-frequency sound responses when dropped on the ground. We approximate ground sounds by first precomputing a modal model of a large concrete slab ($9m \times 9m \times 0.9m$) with 1000 modes with frequencies from $700Hz$ to $4kHz$, and each modes' corresponding multipole coefficients. Next we estimate the ground vibration response $\mathbf{q}(t)$ due to a unit impulse applied at the center of the slab. To synthesize ground sounds for simulations, we simply convolve the ground response with ground contact forces, and emanate multipole radiation from the appropriate contact position.

Example (Ceramic dinner plate): We simulated the fracture of a ceramic plate (see Figure 10). Laboratory experiments of plates dropped on the ground produced qualitatively similar fracture patterns and sounds. For reference, we also provide a fracture-free simulation of a spolling (spinning + rolling) plate.

Example (Ceramic tile): To compare against our laboratory tile experiments (see Figure 2), we synthesized sounds for a prescored virtual tile fracture experiment (see Figure 12). Our rigid-body restitution model produced faster bouncing behavior and our tile had slightly higher pitch, however qualitatively similar time-varying rigid-body sounds were observed.

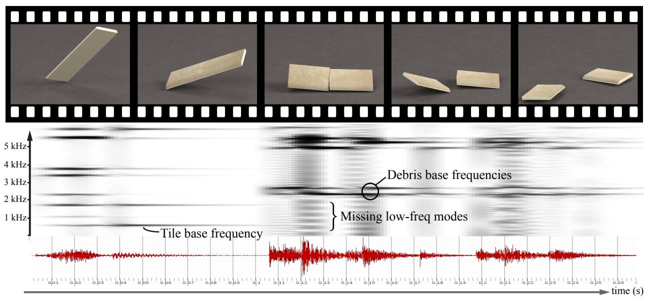

Figure 12: Simulated tile fracture experiments

Example (Glass slab): We dropped a glass slab which shattered into about 70 pieces (see Figure 10). This example was particularly well approximated by Soundbank proxies, and sounded similar even for 100% replacement.

Example (Glass window): We smashed two thick glass windows of differing fracture toughness (see Figure 4), which also fared well with proxy replacement.

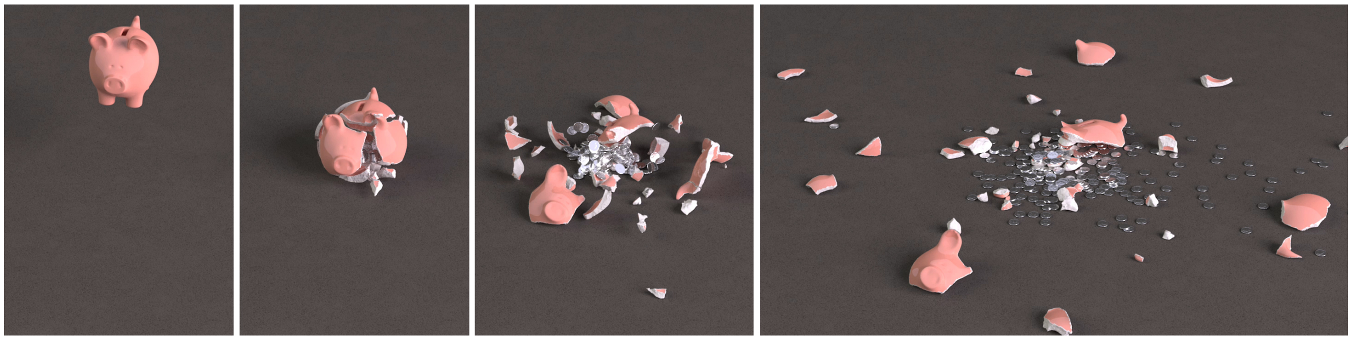

Figure 13: Breaking the bank! *A piggy bank is smashed into 58 pieces, and releases 300 coins.*

Example (Wine glass): We dropped a wine glass onto its bowl, which exploded and left its stem behind (see Figure 10). Similar laboratory experiments produced qualitatively similar sounds.

Example (Piggy bank): We simulated two piggy banks: "Poor Piggy" contained only a few coins, whereas "Rich Piggy" contained many coins (see Figure 13). The latter example had the most expensive fracture dynamics due primarily to the impulse-based solver's handling of hundreds of coins stacked inside.

Example (Table): The smashed table setting (see Figure 1) is our largest example, with over 4000 modes, and over a million quasistatic stress solves. Given the high model-generation costs, the Rigid-Body Soundbank is particularly useful on this example. The video also provides a with/without acoustic transfer comparison.

Comparison (With/without fracture impulses): Without fracture impulses (§3.3), excitations are only due to rigid-body contact forces (recall Figure 5). We synthesized the sound of the glass slab falling onto the ground with and without fracture impulses (see video), to demonstrate the more distinct "crack" obtained when using fracture impulses.

Example (Interactive GPU-accelerated demonstrations): Real-time demonstrations were performed for the dinner plate and the piggy bank. For the spolling and smashing dinner plate (with $\sum_{\text{mode}\,i} \bar{n}_i = 2176$ expansions) our OpenMP 8-core CPU implementation required 0.156s to compute all-mode transfer, whereas our GPU-based transfer implementation (§4) only required 0.0003s (3.3kHz) on an NVIDIA Tesla card—**a 520× speedup**.

Soundbank Comparisons (Varying proxy-replacement threshold): To evaluate results produced by varying degrees of rigid-body soundbank use, we simulated the glass window, glass slab, and table examples with varying proxy-replacement thresholds—as controlled by the frequency-based ω_{proxy} proxy-use criterion. Surprisingly, replacing even all the object sound models with the ellipsoidal proxies (for large reductions in sound model computations) still produced plausible sounds. See the video for these comparisons, and Table 2 for model-generation speedups.

7 Limitations and Future Work

Physically based fracture sound synthesis is a new area, and significant challenges remain. One significant challenge is the approximation of fracture sounds for very large objects, such as buildings, which pose many difficulties: expensive stress/modal analyses, complex debris, dense frequency spectra with huge numbers of eigenmodes, and more complex sound radiation, e.g., simple multipole sound models are invalid for large objects where the listener is inside the object.

Rigid-body sound is a convenient abstraction of brittle fracture, but not all fracture processes are so instantaneous or conveniently modeled. Gradual cracking, such as cracking lake ice or tearing, can require time-domain modeling of crack propagation which complicates modal sound modeling. Our quasistatic fracture impulse is a simple approximation of the effective fracture impulse resulting from complex time-dependent fracture processes. Since crack propagation speeds in brittle materials can be close to the Rayleigh wave speed (typically several km/s) the physical fracture impulses are determined by an extremely rapid process [Gross et al. 1993], e.g., brittle fracture of 10 cm scale objects can occur on the time-scale of a 20 kHz wave period. Due to the ill-posed nature of fracture-impulse estimation with a quasistatic model, we instead proposed an energy-based fracture-impulse model using quasistatic stress.

Ductile objects, such as metals, can undergo visible deformation during the fracture process [O'Brien et al. 2002a] and require additional investigation. Quasistatic fracture modeling works well for contact forces, but dynamic resonance-based fracture and fluid-solid coupling can also be important, e.g., a soprano smashing a wine glass with her voice. Contact coupling can significantly affect vibration-based sounds, and was only approximated by *ad hoc* contact damping. "Hair-line" fractures may not fully separate the object, and can affect vibration-based sounds. In general, modal vibrations should also be coupled with frictional contact mechanics, and pose collision detection challenges [James and Pai 2004].

Thin objects require special care for fracture simulation [Bao et al. 2007], as well as for sound modeling. For example, our wine glass involved both thin shell-like regions, and volumetric regions which proved difficult to mesh using tetrahedra. Thin-shell models also require special treatment for multipole radiation evaluation. Thin objects, such as glass panes, might also exhibit nonlinear vibrations during violent fracture processes [Chadwick et al. 2009]. Our fracture pattern generation does not produce fine debris and dust [Imagire et al. 2009], however precomputed soundbanks could be used to model their sounds.

We have modeled debris radiation using a linear superposition of non-interacting rigid-body sources, but real simulations can involve significant inter-object scattering effects. Our frequency-domain radiation model simplifies the time-domain nature of fracture. For example, far-field sound involves significant time-delay effects, and is also more complicated for large and also fast-moving and spinning objects [Morse and Ingard 1986].

Our rigid-body soundbank is based on homogeneous ellipsoidal primitives for simplicity and convenience, but audible differences can occur for nonconvex geometry (see video). However, it remains to be seen if other geometric primitives, e.g., fracture shards, provide more realistic results. In general, understanding the perceptible shape space for sound proxies, and the extent to which precomputed soundbanks can be used to replace general geometry are open problems. Finally, far more agressive speed-accuracy trade-offs can be made for interactive applications.

A Background on Rigid-Body Sound

This appendix briefly summarizes background material on rigid-body sound synthesis needed for rigid-body fracture sounds. The physical process of rigid-body sound generation is modeled using three steps: (1) contact forces are estimated using rigid-body simulation, (2) surface vibrations are estimated using modal analysis, and (3) acoustic transfer functions describing each mode's sound pressure field are estimated, and evaluated at the listener's position.

Linear modal vibrations are standard in engineering [Shabana 1990], and commonly used in rigid-body sound synthesis [O'Brien et al. 2002b; James et al. 2006]. For "rigid" objects, external forces introduce only infinitesimal deformations. Using a finite element model, the displacements $\mathbf{u} \in \mathbb{R}^{3N}$ of N nodes can be modeled by the linear elastodynamic equation,

$$\mathbf{M}\ddot{\mathbf{u}} + \mathbf{C}\dot{\mathbf{u}} + \mathbf{K}\mathbf{u} = \mathbf{f} \in \mathbb{R}^{3N}, \tag{13}$$

where $\mathbf{f}$ are external forces (e.g., from a rigid-body simulation), $\mathbf{M}$, $\mathbf{C}$ and $\mathbf{K}$ are mass, damping and stiffness matrices, respectively, given by material properties of the object. Here $\mathbf{K}$ is sparse and positive semi-definite, and $\mathbf{M}$ is also sparse and positive definite. The damping matrix $\mathbf{C}$ is approximated by *Rayleigh* damping, $\mathbf{C} = \alpha\mathbf{M} + \beta\mathbf{K}$ for user-chosen nonnegative coefficients α and β. We use tetrahedral meshes, which are generated initially using Isosurface Stuffing [Labelle and Shewchuk 2007].

For linear modal analysis, we solve the generalized eigenvalue problem, $\mathbf{KU} = \mathbf{\Lambda MU}$, where $\mathbf{\Lambda} = \mathsf{diag}(\omega_i^2)$ is the diagonal matrix of eigenvalues, with ω_i the *undamped natural frequency*, and the *mode shapes* are given by the eigenvector matrix, $\mathbf{U}$, which is mass orthogonal, $\mathbf{U}^T\mathbf{MU} = \mathbf{I}$. In practice, we only compute eigenvalues and eigenvectors for frequencies up to 20 kHz, which can be done efficiently using implicitly restarted Arnoldi methods (with "shift-and-invert" spectral transformation) in the ARPACK library [Golub and Van Loan 1996; Lehoucq et al. 1998]. Substituting the modal coordinate transformation, $\mathbf{u} = \mathbf{Uq}$ into (13) yields

$$\ddot{\mathbf{q}} + (\alpha\mathbf{I} + \beta\mathbf{\Lambda})\dot{\mathbf{q}} + \mathbf{\Lambda q} = \mathbf{U}^T\mathbf{f}, \tag{14}$$

with each row an independent simple harmonic oscillator,

$$\ddot{q}_i + (\alpha + \beta\,\omega_i^2)\dot{q}_i + \omega_i^2 q = Q_i(t), \quad i = 1 \ldots n, \tag{15}$$

where $Q_i(t)$ is the *modal force*. These oscillators can be solved efficiently in discrete time using an IIR digital filter [Hamming 1983; James and Pai 2002]. Finally, object vibrations are given by the linear superposition of all vibration modes, $\mathbf{u}(t) = \mathbf{Uq}(t)$.

Acoustic transfer: The radiated sound pressure is approximated by a linear superposition of modal sound pressure contributions. If we denote the object domain by Ω, we can approximate the time-dependent pressure contribution due to a single vibration mode by $|p(\boldsymbol{x})|\,q(t)$, $\boldsymbol{x} \notin \Omega$, where $q(t)$ is the time-harmonic solution of (15), and the spatial part $p(\boldsymbol{x}) : \mathbb{R}^3 \to \mathbb{C}$ is the *acoustic transfer function*, which captures complex diffraction and interreflection effects which are very perceptible and important for realistic sound rendering [James et al. 2006]. For a single vibration mode of frequency ω, its wave number is $k = \omega/c$ where the speed of sound in the surrounding medium is $c = 343m/s$ (at STP). The acoustic transfer function (for a modal vibration $\boldsymbol{u}(x)e^{+i\omega t}$) then satisfies the frequency-domain wave equation, or *Helmholtz equation*,

$$\begin{cases} \nabla^2 p(\boldsymbol{x}) + k^2 p(\boldsymbol{x}) = 0 & \text{in } \mathbb{R}^3 \setminus \Omega \\ \partial_{\boldsymbol{n}} p(\boldsymbol{x}) = -i\omega\rho v_n(\boldsymbol{x}) & \text{on } \partial\Omega \end{cases} \tag{16}$$

where the second equation specifies the *Neumann* boundary condition, $\partial_{\boldsymbol{n}} p = \frac{\partial p}{\partial \boldsymbol{n}}$ is the derivative along the surface's outside normal direction; the air density is $\rho = 1.184kg/m^3$ (at STP), and $v_n(\boldsymbol{x}) = i\omega u_n(\boldsymbol{x})$ is the mode's normal vibration velocity for normal displacement $u_n = \boldsymbol{n}^T\boldsymbol{u}$; a Sommerfeld radiation condition for out-going waves is also needed [Morse and Ingard 1986]. We estimate these acoustic transfer functions using the approach of §4.

B Sparse least-squares $\mathbf{Ku} = \mathbf{f}$ solver details

Constructing $\boldsymbol{\mathcal{P}}$: The null space of $\mathbf{K}$ is spanned by rigid-body modes, which we project out using $\boldsymbol{\mathcal{P}}$. Given tetrahedral node, i, with position (x_i, y_i, z_i), its translation and linearized rotation are columns of $\boldsymbol{T}_i$,

$$\boldsymbol{T}_i = \begin{bmatrix} 1 & 0 & 0 & 0 & z_i & -y_i \\ 0 & 1 & 0 & -z_i & 0 & x_i \\ 0 & 0 & 1 & y_i & -x_i & 0 \end{bmatrix} \quad \Rightarrow \quad \boldsymbol{T} = \begin{bmatrix} T_1 \\ \vdots \\ T_N \end{bmatrix},$$

so that the rank-6 matrix $\boldsymbol{T}$ spans $\mathsf{null}(\mathbf{K})$. We compute the QR factorization, $\boldsymbol{T} = \boldsymbol{QR}$, and cache the $3N$-by-6 orthogonal matrix, $\boldsymbol{Q}$. At runtime, we implement force/displacement projections $\boldsymbol{\mathcal{P}}\mathbf{v}$ using $\boldsymbol{\mathcal{P}}\mathbf{v} = \mathbf{v} - \boldsymbol{QQ}^T\mathbf{v}$ at $O(N)$ cost.

Constructing $\mathbf{V}$: We interpret $\mathbf{V}$ as a vertex displacement basis with translation and linearized rotation eliminated in a particularly convenient way; $\mathbf{V}$ has $3N-6$ columns corresponding to deformation degrees of freedom. While any orthogonal matrix $\mathbf{V}$ with size $3N \times (3N-6)$ which eliminates the rigid-body modes is applicable to (2), in practice, sparser $\mathbf{V}$ can lead to sparser factorizations of $\mathbf{V}^T\mathbf{KV}$. So that $\mathbf{V}^T\mathbf{KV}$ construction has $O(\mathsf{nnz}(\mathbf{K}))$ cost, our $\mathbf{V}$ matrix, with only $3N$ nonzero elements, has the following form:

$$V = \left[\begin{array}{cccccccc} 1 & & & & & & & \\ & \ddots & & & & \mathbf{0} & & \\ & & 1 & & & & & \\ & & & 0 & & & & \\ & & & 0 & & & & \\ & & & 0 & & & & \\ & & & a_1 & & & & \\ & & & a_2 & & & & \\ & & & a_3 & & & & \\ & & & & a_1 & b_1 & & \\ & & & & a_2 & b_2 & & \\ & & & & a_3 & b_3 & & \\ & \mathbf{0} & & & & & 1 & \\ & & & & & & \ddots & \\ & & & & & & & 1 \end{array}\right] \begin{array}{l} \\ \\ \\ \left.\begin{array}{l} \\ \\ \\ \end{array}\right\}\text{vertex } t_1 \\ \left.\begin{array}{l} \\ \\ \\ \end{array}\right\}\text{vertex } t_2 \\ \left.\begin{array}{l} \\ \\ \\ \end{array}\right\}\text{vertex } t_3 \\ \\ \\ \\ \end{array}$$

where t_1, t_2 and t_3 are any three non-collinear tetrahedral vertices upon which $\mathbf{V}$ applies the following six constraints: (1-3) we fix t_1 by imposing zero translation; then (4) to eliminate the rotation (c.f. [Bao et al. 2007]) we constrain t_2 to move along a fixed direction, $\boldsymbol{a} = (a_1, a_2, a_3) = \mathsf{normalize}(\boldsymbol{x}_{t_2} - \boldsymbol{x}_{t_1})$; then (5-6) we constrain t_3 to lie on an perpendicular plane by letting $(b_1, b_2, b_3) = \mathsf{normalize}([\boldsymbol{a} \times (\boldsymbol{x}_{t_3} - \boldsymbol{x}_{t_1})] \times \boldsymbol{a})$. All other rows match the identity matrix. By construction $\mathbf{V}$ is orthogonal.

C Proof of least-residual solution to (2)

Given the solution $\mathbf{r}$ of (2), we show that $\mathbf{u} = \mathbf{V}\mathbf{r}$ is the least-residual solution of the compatible system $\mathbf{K}\mathbf{u} = \mathcal{P}\mathbf{f}$. Since $\mathbf{K}$ is a $3N \times 3N$ symmetric and rank-deficient matrix, we write it as $\mathbf{K} = \mathbf{U}\mathbf{S}\mathbf{U}^T$ using thin SVD, where $\mathbf{U}$ is the $3N \times (3N-6)$ orthogonal basis matrix spanning $\mathsf{range}(\mathbf{K})$, and $\mathbf{S}$ is an invertible $(3N-6) \times (3N-6)$ diagonal matrix of singular values. By compatibility, we can write $\mathcal{P}\mathbf{f} = \mathbf{U}\tilde{\mathbf{f}}$. Substituting these expressions into (2), we see that $\mathbf{r}$ satisfies

$$\mathbf{V}^T\mathbf{U}\mathbf{S}\mathbf{U}^T\mathbf{V}\mathbf{r} = \mathbf{V}^T\mathbf{U}\tilde{\mathbf{f}}. \tag{17}$$

Since $\mathbf{V}^T\mathbf{U}$ is nonsingular (both $\mathbf{U}$ and $\mathbf{V}$ are orthogonal), we have $\mathbf{U}^T\mathbf{V}\mathbf{r} = \mathbf{S}^{-1}\tilde{\mathbf{f}}$. Therefore $\mathbf{V}\mathbf{r}$ is a least-residual solution:

$$\mathbf{K}\mathbf{V}\mathbf{r} = (\mathbf{U}\mathbf{S}\mathbf{U}^T)\mathbf{V}\mathbf{r} = \mathbf{U}\mathbf{S}(\mathbf{U}^T\mathbf{V}\mathbf{r}) = \mathbf{U}\mathbf{S}\mathbf{S}^{-1}\tilde{\mathbf{f}} = \mathcal{P}\mathbf{f}. \tag{18}$$

D Multipole Coefficient Solver

Background on Helmholtz Boundary Integral Equation: If we denote the object domain as Ω and its surface as $\Gamma = \partial\Omega$, at any point $\boldsymbol{x}$ outside of the vibrating object the acoustic transfer function satisfying (16) is given by the *Kirchhoff integral formula*

$$p(\boldsymbol{x}) = \int_\Gamma \left[G(\boldsymbol{x};\boldsymbol{y}) \frac{\partial p}{\partial \boldsymbol{n}}(\boldsymbol{y}) - \frac{\partial G}{\partial \boldsymbol{n}}(\boldsymbol{x};\boldsymbol{y})\, p(\boldsymbol{y}) \right] d\Gamma_{\boldsymbol{y}} \tag{19}$$

where $G(\boldsymbol{x};\boldsymbol{y})$ is the Helmholtz Green's function. In this paper, we assume each sounding object is isolated from others, and ignore the scattering interaction between multiple sounding objects, therefore we use the free-space Green's function, $G(\boldsymbol{x};\boldsymbol{y}) = e^{-ik\|\boldsymbol{x}-\boldsymbol{y}\|}/(4\pi\|\boldsymbol{x} - \boldsymbol{y}\|)$. Evaluating (19) requires the acoustic transfer function values $p(\boldsymbol{y})$ and its outward-pointing normal derivative $\partial_{\boldsymbol{n}} p(\boldsymbol{y})$ on the object's surface $\boldsymbol{y} \in \Gamma$; the normal derivative is given by the Neumann boundary condition of (16), and the boundary transfer values are computed using the boundary element solver. In our implementation, we obtain $p(\boldsymbol{y})$ on Γ using the *FastBEM Acoustics* implementation (`www.fastbem.com`) of the fast multipole boundary element method [Liu 2009].

Multipole Expansion: The cost of evaluating the acoustic transfer $p(\boldsymbol{x})$ with (19) is linear in the number of surface triangles, and becomes expensive for fine surface meshes. Unfortunately, breaking objects into tiny pieces almost inevitably leads to increased geometric complexity which will slow down our sound synthesis pipeline. We circumvent this issue by using a standard multipole expansion of $p(\boldsymbol{x})$. The Green's function can be expanded in a series of singular and regular basis functions using the identity [Gumerov and Duraiswami 2004],

$$G(\boldsymbol{x};\boldsymbol{y}) = ik \sum_{n=0}^{\infty} \sum_{m=-n}^{n} S_n^m(\boldsymbol{x} - \boldsymbol{x}_0)\, R_n^{-m}(\boldsymbol{y} - \boldsymbol{x}_0), \tag{20}$$

where S_n^m is the singular spherical Helmholtz basis function described previously (7); R_n^m is its regular counterpart,

$$R_n^m(\boldsymbol{r}) = j_n(kr)\, Y_n^m(\theta, \phi) \tag{21}$$

where $j_n \in \mathbb{R}$ are the *spherical Bessel functions*; here $\boldsymbol{x}_0$ is an arbitrary fixed point satisfying $\|\boldsymbol{x} - \boldsymbol{x}_0\| > \|\boldsymbol{y} - \boldsymbol{x}_0\|$ to ensure that (20) converges absolutely and uniformly; in our implementation, we place $\boldsymbol{x}_0$ at the object's center of mass. Substituting (20) into (19), we can pull the S_n^m out of the surface integral to obtain the multipole expansion of $p(\boldsymbol{x})$:

$$\begin{aligned} p(\boldsymbol{x}) = \int_\Gamma ik \Bigg[& \sum_{n=0}^{\infty} \sum_{m=-n}^{n} S_n^m(\boldsymbol{x} - \boldsymbol{x}_0)\, R_n^{-m}(\boldsymbol{y} - \boldsymbol{x}_0)\, \frac{\partial p}{\partial \boldsymbol{n}}(\boldsymbol{y}) \\ & - p(\boldsymbol{y}) \sum_{n=0}^{\infty} \sum_{m=-n}^{n} S_n^m(\boldsymbol{x} - \boldsymbol{x}_0)\, \frac{\partial R_n^{-m}}{\partial \boldsymbol{n}}(\boldsymbol{y} - \boldsymbol{x}_0) \Bigg] d\Gamma_{\boldsymbol{y}} \\ = & \sum_{n=0}^{\infty} \sum_{m=-n}^{n} S_n^m(\boldsymbol{x} - \boldsymbol{x}_0)\, M_n^m \end{aligned}$$

where the multipole coefficients, M_n^m, can be evaluted numerically from the formula

$$M_n^m = ik \int_\Gamma \left[R_n^{-m}(\boldsymbol{y} - \boldsymbol{x}_0)\, \frac{\partial p}{\partial \boldsymbol{n}}(\boldsymbol{y}) - p(\boldsymbol{y})\, \frac{\partial R_n^{-m}}{\partial \boldsymbol{n}}(\boldsymbol{y} - \boldsymbol{x}_0) \right] d\Gamma_{\boldsymbol{y}}, \tag{22}$$

with $\partial_{\boldsymbol{n}} p$ from the Neumann BC (16), and p from the BEM solver.

E Derivation of Proxy Scaling Relations

The scalings (9) follow from $\boldsymbol{x} \to \gamma\boldsymbol{x}$ as follows. Since the object's volume and mass scale as γ^3, the mass matrix scales as $\mathbf{M} \to \gamma^3\mathbf{M}$. Since the modes are mass orthogonal, $\mathbf{U}^T\mathbf{M}\mathbf{U} = \mathbf{I}$, it follows that $\mathbf{U} \to \gamma^{-3/2}\mathbf{U}$. Stiffness matrix scaling, $\mathbf{K} \to \gamma\mathbf{K}$, follows from the fact that matrix elements are integrals of energy Hessians (with γ^{-2} scaling) over volumes (with γ^3 scaling). It follows that the eigenvalues scale as $\omega^2 \to \omega^2\, \mathsf{scaling}(\mathbf{K})/\mathsf{scaling}(\mathbf{M}) = \omega^2/\gamma^2$, so that $\omega \to \omega/\gamma$, $k \to k/\gamma$, and $k\boldsymbol{x} \to k\boldsymbol{x}$. Multipole M_n^m scaling follows from (22), where it suffices to consider the term, $ik\, R_n^{-m}(\boldsymbol{y} - \boldsymbol{x}_0)\, \frac{\partial p}{\partial \boldsymbol{n}}(\boldsymbol{y}) d\Gamma_{\boldsymbol{y}}$: k gives a γ^{-1} factor; R_n^{-m} is scale invariant since it depends on kr; the $d\Gamma$ area introduces a γ^2 factor; $\partial_{\boldsymbol{n}} p$ is the Neumann BC, which scales with $\omega^2 u_n$ as $\gamma^{-2}\gamma^{-3/2} = \gamma^{-7/2}$. Multiplying $\gamma^{-1} \cdot \gamma^2 \cdot \gamma^{-7/2}$ we obtain $M_n^m \to \gamma^{-5/2} M_n^m$.

F Estimation of Proxy Contact Point

Given a force $\mathbf{f}_p$ applied to the elliptical sound proxy, we apply it to the surface position where the surface normal direction $\boldsymbol{n}$ is opposite to the force direction, $\boldsymbol{n} = -\mathbf{f}_p/\|\mathbf{f}_p\|$. Let the implicit surface for the ellipsoid be $g(x, y, z) = \frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} - 1$, then the surface normal direction is coincident with its gradient $\nabla g = (\frac{2x}{a^2}, \frac{2y}{b^2}, \frac{2z}{c^2})^T$. The surface position with matching normal direction $\boldsymbol{n}$ satisfies the equation $\frac{\nabla g}{\|\nabla g\|} = \boldsymbol{n} \equiv (n_x, n_y, n_z)^T$. Letting $X = x^2/a^2$, $Y = y^2/b^2$ and $Z = z^2/c^2$, we obtain a 3×3 linear equation,

$$\begin{bmatrix} (n_x^2 - 1)/a^2 & n_x^2/b^2 & n_x^2/c^2 \\ n_y^2/a^2 & (n_y^2 - 1)/b^2 & n_y^2/c^2 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{23}$$

whose solution yields the proxy's surface contact point,

$$\boldsymbol{p}_{scp} = \left(\mathsf{sgn}(n_x)\sqrt{a^2 X}, \mathsf{sgn}(n_y)\sqrt{b^2 Y}, \mathsf{sgn}(n_z)\sqrt{c^2 Z} \right). \tag{24}$$

Acknowledgements

We thank Ethan James for *Fracture Lab* assistance, anonymous reviewers for criticism, and the Yafaray free open-source raytracing engine. This work was supported in part by the National Science Foundation (HCC-0905506), the Alfred P. Sloan Foundation, Pixar, Intel, Advanced CAE Research, Vision Research, and Autodesk. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

BAO, Z., HONG, J.-M., TERAN, J., AND FEDKIW, R. 2007. Fracturing rigid materials. *IEEE Transactions on Visualization and Computer Graphics*, 14, 370–378.

BIELSER, D., GLARDON, P., TESCHNER, M., AND GROSS, M. H. 2004. A state machine for real-time cutting of tetrahedral meshes. *Graphical Models 66*, 6, 398–417.

BONET, J., AND WOOD, R. D. 1997. *Nonlinear Continuum Mechanics for Finite Element Analysis*. Cambridge University Press.

BONNEEL, N., DRETTAKIS, G., TSINGOS, N., VIAUD-DELMON, I., AND JAMES, D. 2008. Fast modal sounds with scalable frequency-domain synthesis. *ACM Transactions on Graphics 27*, 3 (Aug.), 24:1–24:9.

CARPINTERI, A., AND LACIDOGNA, G. 2007. *Earthquakes and Acoustic Emission*. Taylor & Francis.

CHADWICK, J. N., AN, S. S., AND JAMES, D. L. 2009. Harmonic Shells: A practical nonlinear sound model for near-rigid thin shells. *ACM Trans. Graph. 28*, 5, 1–10.

COOK, P. 1997. Physically Informed Sonic Modeling (PhISM): Synthesis of percussive sounds. *Computer Music Journal*, 38–49.

DOBASHI, Y., YAMAMOTO, T., AND NISHITA, T. 2003. Real-time rendering of aerodynamic sound using sound textures based on computational fluid dynamics. *ACM Transactions on Graphics 22*, 3 (July), 732–740.

DOEL, K., KNOTT, D., AND PAI, D. 2004. Interactive simulation of complex audiovisual scenes. *Presence: Teleoperators & Virtual Environments 13*, 1, 99–111.

DUNEGAN, H., HARRIS, D., AND TATRO, C. 1968. Fracture analysis by use of acoustic emission. *Eng Fracture Mechanics 1*, 1.

GOLUB, G. H., AND VAN LOAN, C. F. 1996. *Matrix Computations*, 3rd ed. Johns Hopkins University Press, Baltimore, MD, USA.

GROSS, D., AND SEELING, T. 2006. *Fracture mechanics: with an introduction to micromechanics*. Springer.

GROSS, S., FINEBERG, J., MARDER, M., MCCORMICK, W., AND SWINNEY, H. 1993. Acoustic emissions from rapidly moving cracks. *Physical review letters 71*, 19, 3162–3165.

GROSSE, C., AND OHTSU, M. 2008. *Acoustic Emission Testing*. Springer Verlag.

GUENDELMAN, E., BRIDSON, R., AND FEDKIW, R. 2003. Nonconvex rigid bodies with stacking. In *Proc. ACM SIGGRAPH*, 871–878.

GUMEROV, N. A., AND DURAISWAMI, R. 2004. *Fast Multipole Methods for the Helmholtz Equation in Three Dimensions*, first ed. Elsevier Science.

HAHN, J., FOUAD, H., GRITZ, L., AND LEE, J. 1998. Integrating sounds and motions in virtual environments. *Presence 7*, 1, 67–77.

HAMMING, R. W. 1983. *Digital Filters*. Prentice-Hall.

HARRIS, M. 2007. Optimizing Parallel Reduction in CUDA. *NVIDIA Developer Technology*.

HELLRUNG, J., SELLE, A., SHEK, A., SIFAKIS, E., AND TERAN, J. 2009. Geometric fracture modeling in Bolt. In *SIGGRAPH 2009: Talks*, ACM, New York, NY, USA, 1–1.

IMAGIRE, T., JOHAN, H., AND NISHITA, T. 2009. A fast method for simulating destruction and the generated dust and debris. *The Visual Computer 25*, 5–7 (May), 719–727.

JAMES, D. L., AND PAI, D. K. 2002. DyRT: Dynamic response textures for real time deformation simulation with graphics hardware. In *Proc. ACM SIGGRAPH*, ACM, New York, NY, USA, 582–585.

JAMES, D. L., AND PAI, D. K. 2004. BD-Tree: Output-sensitive collision detection for reduced deformable models. *ACM Transactions on Graphics 23*, 3 (Aug.), 393–398.

JAMES, D. L., BARBIC, J., AND PAI, D. K. 2006. Precomputed Acoustic Transfer: Output-sensitive, accurate sound generation for geometrically complex vibration sources. *ACM Transactions on Graphics 25*, 3 (July), 987–995.

LABELLE, F., AND SHEWCHUK, J. R. 2007. Isosurface stuffing: Fast tetrahedral meshes with good dihedral angles. *ACM Transactions on Graphics 26*, 3 (July), 57:1–57:10.

LEHOUCQ, R., SORENSEN, D., AND YANG, C. 1998. *ARPACK Users' Guide: Solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods*. SIAM.

LI, Z., AND BAZANT, Z. 1998. Acoustic emissions in fracturing sea ice plate simulated by particle system. *Journal of Engineering Mechanics 124*, 1, 69–79.

LIU, Y. J. 2009. *Fast Multipole Boundary Element Method: Theory and Applications in Engineering*. Cambridge University Press.

LOCKNER, D., BYERLEE, J., KUKSENKO, V., PONOMAREV, A., AND SIDORIN, A. 1991. Quasi-static fault growth and shear fracture energy in granite. *Nature 350* (March), 39–42.

LOCKNER, D. 1993. The role of acoustic emission in the study of rock fracture. In *Intl. J. Rock Mech. Mining Sci.*, vol. 30, Pergamon, 883–889.

LUYTEN, H., AND VLIET, T. 2006. Acoustic emission, fracture behavior and morphology of dry crispy foods: A discussion article. *Journal of Texture Studies 37*, 3, 221–240.

MCADAMS, S., CHAIGNE, A., AND ROUSSARIE, V. 2004. The psychomechanics of simulated sound sources: Material properties of impacted bars. *The Journal of the Acoustical Society of America 115*, 3, 1306–1320.

MOLINO, N., BAO, Z., AND FEDKIW, R. 2004. A virtual node algorithm for changing mesh topology during simulation. *ACM Transactions on Graphics 23*, 3 (Aug.), 385–392.

MORSE, P., AND INGARD, K. 1986. *Theoretical Acoustics*. Princeton University Press.

MÜLLER, M., DORSEY, J., AND MCMILLAN, L. 2001. Real-time simulation of deformation and fracture of stiff materials. In *Eurographics Computer Animation and Simulation*, Springer-Verlag Wien, 113–124.

NORTON, A., TURK, G., BACON, B., GERTH, J., AND SWEENEY, P. 1991. Animation of fracture by physical modeling. *The Visual Computer 7*, 4, 210–219.

O'BRIEN, J. F., AND HODGINS, J. K. 1999. Graphical modeling and animation of brittle fracture. In *SIGGRAPH 99*, Computer Graphics Proceedings, Annual Conference Series, 137–146.

O'BRIEN, J. F., COOK, P. R., AND ESSL, G. 2001. Synthesizing sounds from physically based motion. In *Proceedings of ACM SIGGRAPH 2001*, Computer Graphics Proceedings, Annual Conference Series, 529–536.

O'BRIEN, J. F., BARGTEIL, A. W., AND HODGINS, J. K. 2002. Graphical modeling and animation of ductile fracture. *ACM Transactions on Graphics 21*, 3 (July), 291–294.

O'BRIEN, J. F., SHEN, C., AND GATCHALIAN, C. M. 2002. Synthesizing sounds from rigid-body simulations. In *2002 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, 175–182.

OCHMANN, M. 1995. The Source Simulation Technique for Acoustic Radiation Problems. *Acustica 81*.

PAI, D. K., VAN DEN DOEL, K., JAMES, D. L., LANG, J., LLOYD, J. E., RICHMOND, J. L., AND YAU, S. H. 2001. Scanning physical interaction behavior of 3d objects. In *ACM SIGGRAPH 2001*, Computer Graphics Proceedings, Annual Conference Series, 87–96.

PAIGE, C. C., AND SAUNDERS, M. A. 1975. Solution of sparse indefinite systems of linear equations. *SIAM J. Numerical Analysis 12*, 617–629.

PARKER, E. G., AND O'BRIEN, J. F. 2009. Real-time deformation and fracture in a game environment. In *SCA '09: Proc. 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ACM, New York, NY, USA, 165–175.

PAULY, M., KEISER, R., ADAMS, B., DUTRÉ, P., GROSS, M., AND GUIBAS, L. J. 2005. Meshless animation of fracturing solids. *ACM Transactions on Graphics 24*, 3 (Aug.), 957–964.

PENTLAND, A., AND WILLIAMS, J. 1989. Good Vibrations: Modal Dynamics for Graphics and Animation. In *Computer Graphics (Proceedings of SIGGRAPH 89)*, vol. 23, 215–222.

PICARD, C., TSINGOS, N., AND FAURE, F. 2009. Retargetting example sounds to interactive physics-driven animations. In *AES 35th International Conference on Audio for Games*.

PRESS, W., TEUKOLSKY, S., VETTERLING, W., AND FLANNERY, B. 2007. *Numerical Recipes: The art of scientific computing*. Cambridge University Press.

RAGHAVACHARY, S. 2002. Fracture generation on polygonal meshes using Voronoi polygons. In *ACM SIGGRAPH 2002 Conference Abstracts and Applications*, ACM, New York, NY, USA, 187–187.

RAGHUVANSHI, N., AND LIN, M. C. 2006. Interactive Sound Synthesis for Large Scale Environments. In *SI3D '06: Proceedings of the 2006 symposium on Interactive 3D graphics and games*, ACM Press, New York, NY, USA, 101–108.

ROCCHESSO, D., AND FONTANA, F., Eds. 2003. *The Sounding Object*. Edizioni di Mondo Estremo, Florence, Italy, ch. High-level models: bouncing, breaking, rolling, crumpling, pouring (by M. Rath and F. Fontana), 186–187.

SHABANA, A. A. 1990. *Theory of Vibration, Volume II: Discrete and Continuous Systems*. Springer-Verlag, New York, NY.

SMITH, J., WITKIN, A., AND BARAFF, D. 2000. Fast and controllable simulation of the shattering of brittle objects. In *Graphics Interface*, Blackwell Publishing, 27–34.

SU, J., SCHROEDER, C., AND FEDKIW, R. 2009. Energy stability and fracture for frame rate rigid body simulations. In *SCA '09: Proc. 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ACM, New York, NY, USA, 155–164.

TAKALA, T., AND HAHN, J. 1992. Sound rendering. In *Computer Graphics (Proceedings of SIGGRAPH 92)*, 211–220.

TERZOPOULOS, D., AND FLEISCHER, K. 1988. Modeling inelastic deformation: Viscoelasticity, plasticity, fracture. In *Computer Graphics (Proceedings of SIGGRAPH 88)*, 269–278.

TREBIEN, F., AND OLIVEIRA, M. M. 2009. Realistic real-time sound re-synthesis and processing for interactive virtual worlds. *The Visual Computer 25*, 5–7 (May), 469–477.

TSINGOS, N., GALLO, E., AND DRETTAKIS, G. 2004. Perceptual audio rendering of complex virtual environments. *ACM Transactions on Graphics 23*, 3 (Aug.), 249–258.

VAN DEN DOEL, K., AND PAI, D. K. 1996. Synthesis of shape dependent sounds with physical modeling. In *Intl Conf. on Auditory Display*.

VAN DEN DOEL, K., KRY, P. G., AND PAI, D. K. 2001. FoleyAutomatic: Physically-Based Sound Effects for Interactive Simulation and Animation. In *ACM SIGGRAPH 2001*, Computer Graphics Proceedings, Annual Conference Series, 537–544.

WARREN, W., AND VERBRUGGE, R. 1984. Auditory perception of breaking and bouncing events: A case study in ecological acoustics. *Journal of Experimental Psychology 10*, 5, 704–712.

ZHENG, C., AND JAMES, D. L. 2009. Harmonic Fluids. *ACM Transactions on Graphics 28*, 3 (July), 37:1–37:12.