Integrated Multimodal Interaction Using Normal Maps

Auston Sterling* Ming C. Lin†

University of North Carolina at Chapel Hill
http://gamma.cs.unc.edu/MultiDispTexture

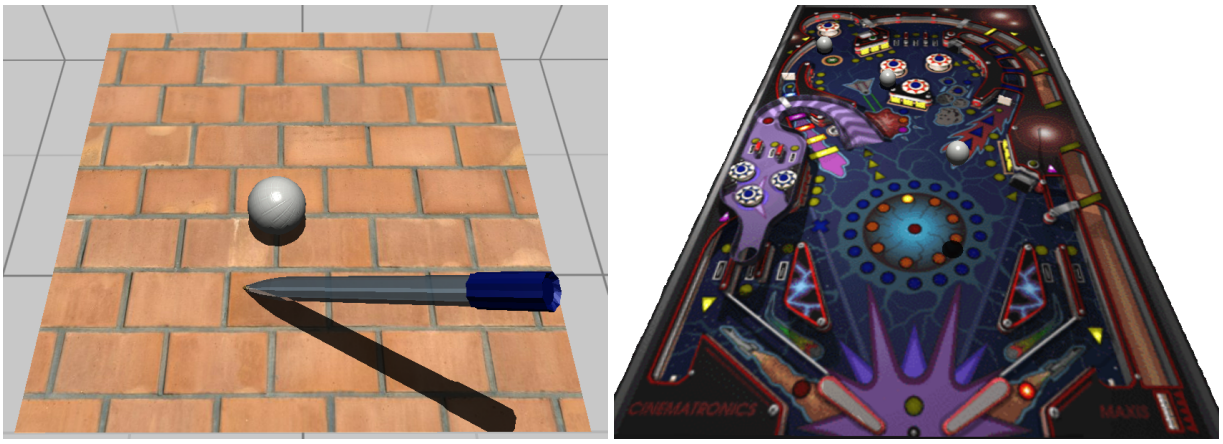


Figure 1: Results from our demonstrations: A virtual environment with multimodal interaction (left) and a pinball simulation on a flat plane (right).

Abstract

In this paper, we explore texture mapping as a unified representation for enabling realistic multimodal interaction with finely-detailed surfaces. We first present a novel approach to modifying collision handling between *textured* rigid-body objects; we then show how normal maps can be adopted as a unified representation to synthesize complex sound effects from long-lasting collisions and perform rendering of haptic textures. The resulting multimodal display system allows a user to see, hear, and feel complex interactions with textured surfaces. By using normal maps as a unified representation for seamlessly integrated multimodal interaction instead of complex triangular meshes otherwise required, this work is able to achieve up to 25 times performance speedup and reduce up to six orders of magnitude in memory storage. We further validate the results through a user study which demonstrates that the subjects are able to correctly identify the material texture of a surface through interaction with its normal map.

Index Terms: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Physically based modeling

1 Introduction

In computer graphics, texture mapping has been one of the most widely used techniques to improve the visual fidelity of objects while significantly accelerating the rendering performance. There are several popular texture representations, such as displacement maps [4], bump mapping with normal maps [1, 2], parallax maps [9, 27], relief maps [15, 20], etc., and they are used mostly as 'imposters' for rendering static scenes. These textures are usually mapped onto objects' surfaces represented with simplified geometry. The fine details of the objects are visually encoded in these texture representations. By replacing the geometric detail with a texture equivalent, the resulting rendered image can be made to appear much more complex than its underlying polygonal geometry would otherwise convey. These representations also come with a significant increase in performance: texture maps can be used for real-time augmented and virtual reality applications on low-end commodity devices.

Sensory conflict occurs when there is a mismatch between information perceived through multiple senses and can cause a break in immersion in a virtual environment. When textures are used to represent complex objects with simpler geometry, sensory conflict becomes a particular concern. In an immersive virtual environment, a user may see a rough surface of varying heights and slopes represented by its texture equivalent mapped to a flat surface. In the real world, objects behave very differently when bouncing, sliding, or rolling on bumpy or rough surfaces than they do on flat surfaces. With visually complex detail and different, contrasting physical behavior due to the simple flat surface, sensory conflict can easily occur—breaking the sense of immersion in the virtual environment. In order to capture such behaviors, the geometry used in a physics simulator would usually require a fine triangle mesh with sufficient surface detail, but in most cases a sufficiently fine mesh is unavailable or would require prohibitive amounts of memory to store.

Since the given texture maps contain information about the fine detail of the mapped surface, it is possible to use that information to recreate the behavior of the fine mesh. Haptic display and sound rendering of textured surfaces have both been independently explored [16, 23], but texture representations of detail have not been previously used for visual simulation of dynamic behavior due to collisions and contacts between rigid bodies. In order to minimize sensory conflict, it is critical to present a unified and seamlessly integrated multimodal display to users, ensuring that the sensory feedback is consistent across the senses of sight, hearing, and touch for both coarse and fine levels of detail.

Motivated by the need to address the sensory conflict due to the use of textures in a multimodal virtual environment, in this paper

*e-mail: austonst@cs.unc.edu
†e-mail: lin@cs.unc.edu

we examine the use of texture maps as a unified representation of fine detail for sight, hearing, and touch. Due to its popularity and efficiency, we revisit normal mapping and explore its uses as a unified representation of fine geometric detail to improve perceptual realism for multimodal interaction, while maintaining real-time performance. The main results of this work include:

- A new effective method for visual simulation of physical behaviors for textured rigid objects;
- A seamlessly integrated multisensory interaction system using normal maps; and
- Evaluation and analysis of texture-based multimodal display and their effects on task performance.

The rest of the paper is organized as follows. We first discuss why we have selected normal maps as our texture representation for multi-modal display and describe how each mode of interaction can use normal maps to improve perception of complex geometry (Sec. 3.1). We highlight how the behavior of virtual objects as they interact with a large textured surface, and describe a new method to improve visual perception of the simulated physical behaviors of colliding objects with a textured surface using normal maps. We also demonstrate how to use the same normal maps in haptic display and sound rendering of textured surfaces (Sec. 3). We have implemented a prototype multimodal display system using normal maps and performed both qualitative and quantitative evaluations of its effectiveness on perceptual quality of the VR experience and objective measures on task completion (Sec. 4). Our findings suggest that, as an early exploration of textures for seamlessly integrated multi-sensory interaction, normal maps can serve as an effective, unified texture representation for multimodal display and the resulting system generally improves task completion rates with greater ease over use of single modality alone.

2 Previous Work

Normal maps are used throughout this paper as the representation of fine detail of the surface of objects. Normal maps were originally introduced for the purposes of bump mapping, where they would perturb lighting calculations to make the details more visibly noticeable [1]. Not all texture mapping techniques for fine detail use normal maps. Displacement mapping, parallax mapping, and a number of more recent techniques use height maps to simulate parallax and occlusion [4, 9, 27]. Relief mapping uses both depths and normals for more complex shading [15, 20]. A recent survey goes into more detail about many of these techniques [25]. Mapping any of these textures to progressive meshes can preserve texture-level detail as the level-of-detail of the mesh shifts [2].

In most rendering algorithms which use height maps, the heights are used to find the intersection point with a ray to the viewer or light. The physical behaviors we seek to replicate in this work do not require computation of ray-heightmap intersections. We instead adopt the use of normal maps as it provides the normal variation required to compute the approximated physical behaviors and multimodal display of fine geometry, while allowing approximation of local depth.

Height maps mapped to object surfaces have been used to modify the behavior of collisions in rigid-body simulations [13]. We are not aware of similar work done using normal maps.

In haptic rendering, a 3D object's geometries and textures can be felt by applying forces based on point-contacts with the object [7, 8]. Complex objects can also be simplified, with finer detail placed in a displacement map and referenced to produce accurate force *and torque* feedback on a probing object [16]. The mapping of both normal and displacement maps to simplified geometry for the purposes of haptic feedback has also been explored [28]. Dynamic deformation textures, a variant of displacement maps, can be mapped to create detailed objects with a rigid center layer and deformable outer layer. The technique has been extended to allow for 6-degree-of-freedom haptic interaction with these deformable objects [6]. A common approach to force display of textures is to apply lateral force depending on the gradient of a height map such that the user of the haptic interface feels more resistance when moving "uphill" and less resistance when moving "downhill" [11, 12]. Our approach to haptic rendering of textures applies force feedback to simulate the presence of planes which reproduce this effect, and similarly we use a simplified model for interaction with dynamic rigid-body objects.

Modal analysis and synthesis are commonly used techniques for synthesizing realistic sound [29]. Modal synthesis has been integrated with rigid-body physics simulators in order to produce contact sounds that synchronize with collision events. To handle objects with arbitrary geometry, they can be decomposed with finite element methods [14]. Further speed optimizations can be made based on psychoacoustics, such as mode compression and truncation [21]. We synthesize transient impact sounds by directly using this technique.

Sounds created by long-lasting contacts between objects require some additional effort. Fractal noise is a common way of representing the small impacts generated during rolling and scraping [5]. We perform sound synthesis for lasting sounds by using the framework for synthesizing contact sounds between textured objects [23]. This work introduced a multi-level model for lasting contact sounds combining fractal noise with impulses collected from the normal maps on the surfaces of the objects. This application of normal maps to sound generation without similar application to rigid-body dynamics causes noticeable sensory conflict between the produced audio and visible physical behavior.

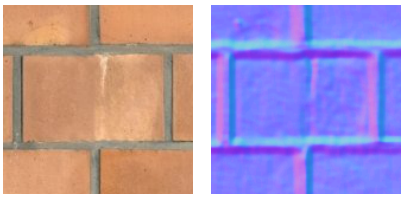
3 Overview and Representation

Our system uses three main components to create a virtual scene where a user can experience through multiple modalities of interaction. In this section, we describe the components themselves and how they use normal maps to improve sense of fine details. A rigid body physics simulator controls the movement of objects. The only form of user input is through a haptic device, which also provides force feedback to stimulate the sense of touch. Finally, modal sound synthesis is used to dynamically generate the auditory component of the system.

3.1 Design Consideration

For this initial investigation on addressing the sensory conflict due to inconsistent adoption of texture mapping, we have chosen to use normal maps as our texture representation. Other texture representations are potentially applicable and could also be independently explored. Using very high-resolution geometry would automatically produce many of the desired effects, but the performance requirements for *interactive* 3D applications significantly reduces their viability in our early deliberation. This is especially important to consider in augmented and virtual reality applications, where real-time performance must be maintained while possibly operating on a low-end mobile phone or head mounted display.

Height maps (and displacement maps) are closely related to normal maps: normal maps are often created from the normals of the slopes of height maps, and integrating along a line on a normal map produces an approximation of the height. For sound, Ren et al. [23] used normal maps because the absolute height does not affect the resulting sound; it's the change in normal which causes a single impulse to produce meso-level sound. With regard to force display of textured surfaces, the Sandpaper system [12] has been a popular and efficient method for applying tangential forces to simulate slope. Instead of using height maps as suggested by Minsky [11], using normal maps we can scale a sampled normal vector



Diffuse map Normal map

Figure 2: Normal map example. RGB values encode normal vectors in each pixel.

to produce the normal and tangential forces. Our approach to modifying rigid-body collisions also requires knowledge of the normal to compute collision response. Since each component of the system depends directly on the normals, a normal map representation emerges as the natural choice. This is not to say that other representations would necessarily produce inferior results; normal maps are simply the most direct way of implementing and achieving the desired changes and are therefore suitable for this first investigation.

An added convenience is that normal maps are widely supported (including mobile games) and frequently included alongside color textures, while height maps are less common. The application needs, the performance requirement, the ease of implementation, and the wide availability and support on commodity systems all contribute to our decision to adopt normal maps in this work, while other representations may also be possible after some considerable amount of optimization using the same principles laid out in this paper.

3.2 Normal maps

Normal maps are usually stored as RGB images, with the color values encoding vectors normal to the details of the surface they are mapped to. Refer to Figure 2 for an example. It is common practice to create normal maps directly corresponding to a diffuse map, such that the diffuse map can be referenced at a location to get a base color and the normal map can be referenced at the same location for the corresponding normal vector.

Depending on the resolution of the normal map image and the surface area of the object it is mapped to, a normal map can provide very fine detail about the object's surface. As we describe in this paper, this detail—while still an approximation of a more complex surface—is sufficient to replicate other phenomena requiring knowledge of fine detail.

3.3 Rigid Body Dynamics

In order to simulate the movement of objects in the virtual scene, we use a rigid body dynamics simulator. These simulators are designed to run in real time and produce movements of rigid objects that visually appear believable.

Rigid body dynamics has two major steps: collision detection and collision response. Collision detection determines the point of collision between two interpenetrating objects as well as the directions in which to apply force to most quickly separate them. Modifying the normals of an object, as we do with normal maps, does not affect whether or not a collision occurs. This is a significant limitation of a normal map representation without any height or displacement information.

There are numerous algorithms for collision resolution, which determines how to update positions and/or velocities to separate the penetrating objects. In impulse-based approaches, collisions are resolved by applying an impulse in the form of an instantaneous change in each objects' velocity vector v. Δv is chosen to be large enough so that the objects separate in the subsequent timesteps. The change in velocity on an object with mass m is computed by applying a force f over a short time Δt in the direction of the surface

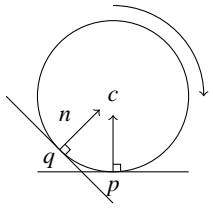


Figure 3: Contact point modification on a rolling ball: given the contact point p and sampled normal n, we want to simulate the collision at point q.

normal n of the other colliding object:

$$\Delta \mathrm{v} = \frac{f \Delta t}{m} \mathrm{n} \tag{1}$$

This process is highly dependent on the normal vectors of each object, and other collision resolution approaches have this same dependency.

3.3.1 Modifying Collision Behavior with Normal Maps

We focus on simulating collisions between small dynamic objects and large textured surfaces whose details would have a large effect on the dynamic object. To get an intuitive understanding of the behavior we seek to replicate, imagine a marble rolling on a brick-and-mortar floor. When the marble rolls to the edge of a brick, the expected behavior would be for it to fall into the mortar between bricks and possibly end up stuck at the bottom.

The level of detail needed to accurately recreate these dynamics with a conventional rigid body physics engine is too fine to be interactively represented with a geometric mesh, especially with large scenes in real-time applications. A normal map contains the appropriate level of detail and is able to represent the flat brick tops and rounded mortar indentations.

In order to change the behavior of collisions to respect fine detail, our solution is to modify the contact point and contact normal reported by the collision detection step. This is an extra step in resolving collisions, and does not require any changes to the collision detection or resolution algorithms themselves.

The contact normal usually comes from the geometry of the colliding objects, but the normal map provides the same information with higher resolution, so our new approach uses the normal map's vectors instead. Given the collision point on the flat surface, we can query the surface normal at that point and instruct the physics engine to use this perturbed normal instead of the one it would receive from the geometry. One side effect of using the single collision point to find the perturbed normal is that it treats the object as an infinitely small probe.

3.3.2 Rolling Objects and Collision Point Modification

There is a significant issue with this technique when simulating rolling objects. Refer to Figure 3 for an example. Two planes are shown, the horizontal one being the plane of the coarse geometry and the other being the plane simulated by the perturbed normal. Note that the contact points with the rolling ball differ when the plane changes. The vector n shows the direction of the force we would ideally like to apply. If we were to apply that force at the original contact point p, the angular velocity of the sphere would change and the ball would begin to roll backwards. In practice, this often results in the sphere rolling in place when it comes across a more extreme surface normal. Instead, we use the sphere radius r, the perturbed surface normal n, and the sphere center c to produce the modified contact point q:

$$q = c - (n \cdot r) \tag{2}$$

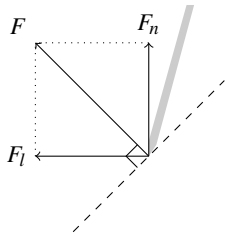


Figure 4: The applied force F can be seen as the sum of a normal force F_n keeping the gray-shaded pen above the surface and a lateral force F_l simulating the surface texture.

This modification applies the force directly towards the center of mass and causes no change in angular velocity, but is less accurate for large spheres and extreme normal perturbations.

This contact point modification is important for perceptually believable rolling effects. Shapes other than spheres do not have the guarantee that the contact point will be in the direction of the $c-n$ vector, so this does not apply in the general case. Generally, we can simply modify the normal without changing the contact point. Including height map information could help determine the true contact points with more complex objects.

3.4 Haptic Interface

We have designed our system to use a PHANToM Desktop haptic device [10]. This device has six degrees of freedom: three translational and three rotational, but only display 3-degree-of-freedom (DOF) forces (i.e. no torques). We have chosen to represent the PHANToM as a pen inside the virtual environment, which matches the scale and shape of the grip. While we could use forces determined by the rigid-body physics engine to apply feedback, the physics update rate (about 60 Hz) is much lower than the required thousands of Hz needed to stably simulate a hard surface.

We simulate the textured surface by projecting the tip of the PHANToM Desktop grip onto the surface in the direction of the coarse geometry's normal. The fine surface normal is queried and interpolated from nearby normal map vectors. The PHANToM simulates the presence of a plane with that normal and the projected surface point. Given the normal vector obtained from the normal map n and pen tip position projected onto the surface p, the equation modeling this plane is:

$$n_x x + n_y y + n_z z - (p_x n_x + p_y n_y + p_z n_z) = 0 \tag{3}$$

The PHANToM now needs to apply the proper feedback force to prevent the pen's tip from penetrating into the plane. This is accomplished using a penalty force, simulating a damped spring pulling the point back to the surface. Using the modified normal vector, the simulated plane serves as a local first order approximation of the surface. Note that while the slopes of the planes produced by the PHANToM can vary significantly based on the normal map, at the position of the pen the plane will coincide with the surface. One way of interpreting this is to see the pen following the flat plane of the surface, with the modified normals adding a lateral force F_l as shown in Figure 4. If the user attempts to move "uphill", the plane will produce a lateral force resisting that motion. This creates an illusion of feeling a textured surface while keeping the pen in contact with the flat underlying surface geometry.

We use a simplified model to interact with dynamic objects. The PHANToM's corresponding pen appearance in the environment is added as an object in the rigid-body physics simulator. Whenever this pen comes in contact with a dynamic object, the physics simulator computes the forces on the objects needed to separate them. We can directly apply a scaled version of this force to the haptic device. This ignores torque as our 3-DOF PHANToM can only apply translational forces. This approach is fast, simple, and lets the user push and interact with objects around the environment.

3.5 Sound Synthesis

Sound is created due to a pressure wave propagating through a medium such as air or water. These waves are often produced by the vibrations of objects when they are struck, and human ears can convert these waves into electrical signals for the brain to process and interpret as sound. One of the most popular physically-based approaches to modeling the creation of sound is modal sound synthesis, which analyzes how objects vibrate when struck at different locations to synthesize contact sounds.

3.5.1 Modal Analysis and Synthesis Background

In order to perform modal analysis, we represent the objects using a discretized representation such as a spring-mass system or a tetrahedral mesh. The dynamics of the object can be represented with the system of differential equations:

$$M\ddot{r} + C\dot{r} + Kr = f \tag{4}$$

r is a vector of displacements from the given starting positions, which are assumed to be at rest. f is the vector of external forces applied to the system. M and K are the mass and stiffness matrices, respectively, which describe the distribution of mass and connectivity of the object. For the damping matrix C, we use Raleigh damping which expresses C as a linear combination of M and K.

This system of equations can be decoupled to produce a bank of modes of vibration. The equation for each mode is a standard damped oscillator, which vibrates at a certain frequency and decays exponentially over time. Almost all of the complex calculations are dependent only of the properties of the objects and therefore can be precomputed and stored.

Sound synthesis at runtime is done in two steps. When an object is struck, the modes of vibration are excited depending on the strike's location and direction. Once the vibrations begin, the modes are sampled and updated at around $44,100$ Hz to create perceptually realistic sound.

3.5.2 Textures and Lasting Sounds

Modal synthesis works well for generating sound that varies for each object, material, and impulse. However, for long-lasting collisions such as scraping, sliding, and rolling, the sound primarily comes from the fine details of the surface which are not captured in the geometry of the input mesh when using texture maps. We adopt the method by Ren et al. [23], which uses three levels of detail to represent objects, with normal maps providing the intermediate level of detail.

At the macro level, the object is represented with the provided triangle mesh.The first frame in which a collision is detected, it is considered transient and impulses are applied according to conventional modal synthesis. If the collision persists for multiple frames, we instead use the lower levels described below.

Even surfaces that look completely flat produce rolling, sliding, and scraping sounds during long-lasting collisions. The micro level of detail contains the very fine details that produce these sounds and are usually consistent throughout the material. Sound at this level is modeled as fractal noise. Playback speed is controlled by the relative velocity of the objects, and the amplitude is proportional to the magnitude of the normal force.

The meso level of detail describes detail too small to be efficiently integrated into the triangle mesh, but large enough to be distinguishable from fractal noise and possibly varying across the surface. Normal maps contain this level of detail, namely the variation in the surface normals. This sound is produced by following the path of the collision point over time. Any time the normal vector changes, the momentum of the rolling or sliding object must change in order to follow the path of that new normal. This change produces an impulse which can be used alongside the others for modal synthesis. This can be mathematically formulated as follows.

	Mesh Size	Offline Memory	Runtime Memory	Physics Time	Visual Time	Haptic Time
Ours	10KB	2.7 MB	270 KB	175 μs	486 μs	60 μs
Coarse Mesh	4.5 MB	288 GB*	450 MB*	3.0 ms	2.1 ms	–**
Fine Mesh	19 MB	4500 GB*	1700 MB*	4.9 ms	7.0 ms	–**

Figure 5: Memory and timing results for our (texture-based) method compared to a similarly detailed coarse mesh (66,500 vertices) and fine mesh (264,200 vertices). Entries marked with * are extrapolated values, since the memory requirements are too high to run on modern machines. Haptic time (**) was not measurable for triangle meshes due to an API limitation. Our method is able to achieve up to **25 times** of runtime speedup and up to **6 orders of magnitude** in memory saving.

Given an object with mass m moving with velocity vector v_t along a face of the coarse geometry with normal vector N whose nearest normal map texel provides a normal n, the momentum p_N orthogonal to the face is:

$$p_N = m(-\frac{v_t \cdot n}{N \cdot n})N \quad (5)$$

This momentum is calculated every time an object's contact point slides or rolls to a new pixel, and the difference is applied as an impulse to the object. More extreme normals or a higher velocity will result in higher momentum and larger impulses. Whenever objects are in collision for multiple frames, both the micro-level fractal noise and the meso-level normal map impulses are applied, and the combined sound produces the long-lasting rolling, sliding, or scraping sound.

4 Implementation and Results

We have described each component of our multimodal system using normal maps. We implemented this prototype system in C++, using NVIDIA's PhysX as the rigid body physics simulator, OGRE3D as the rendering engine, VRPN to communicate with the PHANToM [26], and STK for playing synthesized sound [3].

Our objects were represented using a spring-mass system. This limited us to treating each object as a hollow object with a thin shell, which is not ideal for all situations but is commonly done for efficiency. Another popular approach is finite element methods, which would be more accurate but have a higher precomputation cost. All scenarios we created contained at least one textured surface acting as the ground of the environment, and only its normal map was used to modify collision response, haptic display, or sound rendering.

4.1 Performance Analysis

The sound synthesis module generates samples at 44100Hz, the physics engine updates at 60Hz, and while the PHANToM hardware itself updates at around 1000Hz, the surface normal is sampled to create a new plane once per frame. On a computer with an Intel Xeon E5620 processor and 24GB RAM, the program consistently averages more than 100 frames per second. This update rate is sufficient for real-time interaction, with multi-rate updates [16, 23].

A natural comparison is between our texture-based method and methods using meshes containing the same level of detail. Most of our normal maps are 512×512, so recreating the same amount of detail in a similarly fine mesh would require more than $512^2 = 262114$ vertices and nearly twice as many triangles. As a slightly more realistic alternative, we also compare to a relatively coarse 256×256 mesh with more than $256^2 = 65536$ vertices.

Figure 5 presents memory and timing information when comparing our method to methods using the equivalent geometry meshes instead. The coarse mesh used for modal analysis is greatly reduced in size compared to the finer meshes. We generated these finely-detailed meshes for the sake of comparison, but in practice, neither mesh would be available to a game developer and they would have to make do with the constraints considered in our method.

Modal analysis for audio generation on the finer meshes requires significantly more memory than is available on modern machines, so a simplified mesh is required. The listed "Runtime Memory" is the runtime requirement for modal sound synthesis and primarily consists of the matrix mapping impulses to modal response.

Our method is faster than use of the fine meshes in each mode of interaction. Haptic rendering time using our method took merely 60 μs per frame. The listed "Visual Time" is the time taken to render the surface, either as a flat normal mapped plane, or as a diffuse mapped mesh without normal mapping. The PHANToM's API integrated with VRPN does not support triangular meshes, and we could not test performance of collision detection and haptic rendering manually, though the time needed to compute collision with an arbitrary triangular mesh would taken significantly larger (at least by one to two orders of magnitude based on prior work, such as H-COLLIDE).

The main sound rendering loop runs at around 44 kHz regardless of the chosen representation of detail. The only difference comes from the source of sound-generating impulses: our method collects impulses from a path along the normal map while a mesh-based approach collects impulses reported by the physics engine. Applying impulses to the modal synthesis system is very fast relative to the timed modes of interaction.

4.2 User Study Set-up

In order to evaluate the effectiveness of this multimodal system, we conducted a user study consisting of a series of tasks followed by a questionnaire. One objective of this user study was to determine the overall effectiveness of our system. A user is interacting with the normal map through sight, touch, and sound. If each of these components are well designed and implemented, a user should be able to identify the material by multimodal interaction. The other goal is to see how well the use of multiple senses helps to create a cohesive recognition of the material being probed. Even if users find the haptic display alone is enough to understand the texture of the material being probed, does adding sound cues speed up their process of identifying textures or instead cause sensory conflict?

Twelve participants volunteered to take part in this study experiment. Each user was trained on how to use the PHANToM and was given some time to get used to the system by playing in a test scene (see Figure 1, left). The user then completed a series of six trials. In each trial, a material for the surface was chosen at random, and all aspects of it *except* its visual appearance were applied. That is, the user would be able to feel the surface's texture with the PHANToM, hear the sound generated from ball and PHANToM pen contacts, and see the rolling ball respond to ridges and valleys on the surface. The user was able to cycle through each material's visual appearance (in the form of a texture) by pressing the button on the PHANToM's grip. Their task was to select the material's unknown visual appearance based on the multimodal cues received.

The first three trials provided all three cues—sound, ball, and pen—but in each of the remaining three trials only two of the three cues would be available. The user would be informed before the trial began if any cues were missing. The users were recommended to use all available cues to make their decision, but were otherwise unguided as to how to distinguish the materials. After the trials

Figure 6: The available materials for the user study. 1–3 sounded like bricks, 4–5 sounded like porcelain, 6–8 sounded like metal, and 9–10 sounded like wood.

	ID rate	Time (s)	Ease (1–10)
All modes	78%	38 ± 18	7.9 ± 1.3
No sound	81%	46 ± 45	4.9 ± 2.2
No haptics	54%	41 ± 23	3.6 ± 1.8
No physics	72%	47 ± 58	6.4 ± 2.6

Figure 7: Results comparing effectiveness when limiting the available modes of interaction. "Ease" is evaluated by the users where 1 is difficult and 10 is easy. When using all modes of interaction, users were generally able to identify the material more frequently than when only using two modes and reported that they found identification to be easiest when all modalities of interaction were engaged.

were completed, a short questionnaire was provided for subjective evaluation and feedback.

This study utilizes sensory conflict to guide the users to correctly identify the visual appearance. If the multimodal cues present the sounds, haptic texture, and physical response of a metal surface with regular grooves, but the user has currently selected the visual appearance of a flat, smooth wooden surface, they should recognize the sensory conflict and reject the wooden surface as the answer. Once the user has selected the correct visual appearance (grooved metal in this example), they should feel relatively little sensory conflict and from that realize they have found the answer.

Figure 6 shows the materials chosen for the user study. The users were allowed to look at each of these textures before the trials began, but were not able to feel or hear them. Some of these were specifically chosen to be challenging to distinguish.

4.3 Experimental Results

In Figure 7, we compare the results when varying which modes of interaction are available to users. The ID rate is the percentage of trials in which the user was able to correctly identify the material, and the mean time only takes into account time for correct guesses. The "ease" was provided by the users on the questionnaire, where they were asked to rate on a scale from 1–10 how easy they found it was to identify the material for each combination of modes of interaction. Higher "ease" scores mean the user found it easier to identify the material.

In all cases, the identification rate was higher than 50%, and usually much higher than that. The loss of haptic feedback caused the largest drop in ID rate and ease. The loss of sound actually improved material identification—although the difference is not statistically significant—but users still found identification to be much more perceptually challenging.

	Guesses (%)									
ID	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	**50**	0	33	0	0	17	0	0	10	0
2	0	**80**	0	20	0	0	0	0	0	0
3	0	0	**100**	0	0	0	0	0	0	0
4	0	0	0	**83**	17	0	0	0	0	0
5	0	13	25	0	**50**	0	12	0	0	0
6	0	0	17	0	0	**83**	0	0	0	0
7	8	0	8	0	0	8	**60**	8	8	0
8	0	0	0	0	0	0	0	**75**	25	0
9	0	0	17	0	0	0	0	16	**67**	0
10	0	0	0	0	0	0	0	0	12	**88**

Figure 9: Confusion matrix showing the guesses made by users. For all materials, a significant majority of subjects were able to identify the right materials.

Two more noteworthy results were gathered from a subjective questionnaire, with results shown in Figure 8. Users were asked how frequently they used each of the modes in identifying the material. The users were also asked how well each mode of interaction represented how they would expect the materials to sound or feel. These results could help explain the low identification rate when haptics are disabled: most users both relied heavily on tactile senses and found it be the most accurate mode. The users considered the sound and physics somewhat less accurate but still occasionally useful for determining the materials.

More detailed results from the study are presented in Figure 9. An entry in row i and column j is the percentage of times the user was presented material i and guessed that it was material j. The higher percentages along the diagonal demonstrate the high correct identification rate. Also note that in most categories there is no close second-place guess. The largest exception is that 33% of the time material 1 (brick grid) was mistakenly identified as material 3 (pebbles), likely due to similarity in both material sounds and patterns.

4.4 Discussion

4.4.1 Analysis

Due to the low sample size in the study, many of the possible direct comparisons are not statistically significant. Between identification rates, there was no statistically significant change when removing a mode ($p > .05$), but the removal of haptics came close with $p = .066$.

The subjective user-reported values of ease and accuracy were generally more significant. Users reported that they found material identification to be more difficult when either sound or haptics were removed in comparison to having all modes available ($p < .05$), but did not find identification more difficult when the physics modification was removed ($p > .05$). Cohen's effect size values (d) of 1.66 for the removal of sound and 2.79 for the removal of haptics suggest a very large change in perceptual difficulty when removing these modes. Users also reported that they found the haptics to be more accurate than physics or sound ($p < .05$), but did not find a significant difference in accuracy between physics and sound ($p > .05$). Cohen's effect size values of 1.02 comparing haptics to physics and 1.36 comparing haptics to sound suggest a large difference in the perception of how accurate these modes are.

Overall, these results demonstrate that each mode of interaction is effectively enabled through use of normal maps. Combining multiple modes increases accuracy, which suggests that the users are receiving cohesive, non-conflicting information across their senses. This was a deliberately challenging study, using materials which sounded similar and had similar geometric features and patterns. Furthermore, the task asked users to carefully consider properties of

	Always	Frequently	Occasionally	Rarely	Never	Reported accuracy (1–10)
Haptics	88%	0%	6%	0%	6%	9.3 ± 0.9
Sound	34%	22%	22%	11%	11%	7.6 ± 1.4
Physics	29%	6%	47%	6%	12%	7.3 ± 2.6

Figure 8: Results from question asking how often subjects used each mode of interaction and question asking how well each mode represented the materials (10 is very accurate).

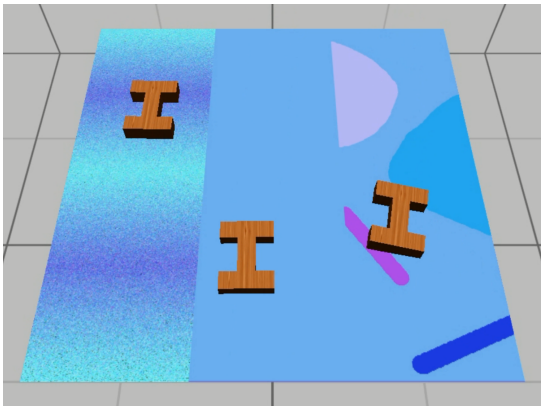


Figure 10: Letter blocks sliding down varied sloped surfaces. The normal map is identical to the diffuse map.

materials not often noticed. Not many people take the time to consider the difference in frequency distributions between the sounds of porcelain and metal, but that distinction could have been important for these tasks. Within such a context, a 78% rate for identifying the correct material out of ten options appears rather promising, and significantly better than random selection.

4.4.2 Applications

We demonstrate several possibilities on the potential use of normal maps as a unified representation for accelerating multi-modal interaction in the supplementary video. Given the prevalence of texture mapping in numerous interactive 3D graphics applications (e.g. games and virtual environment systems), our techniques enable the users to interact with textured objects that have extremely simple underlying geometry (such as flat surfaces) so that they would be able to observe *consistent* dynamic behaviors of moving textured objects, hear the resulting sounds from collisions between them, and feel the object contacts, as shown in Figure 1 (left). The example of the simplified pinball game in Figure 1 (right), balls rolling down Lombard Street in San Francisco City in Figure 11, and letter blocks sliding down sloped surfaces with noise or obstacles in Figure 10 are a few additional examples, where normal maps can be incorporated into physics simulation with multimodal display to provide a more cohesive, immersive experience without sensory disparity. Please see the supplementary video for demonstration of these results.

4.4.3 Comparison with Level-of-Detail Representations

While we have shown comparisons between normal maps and high-resolution meshes as representations of fine detail, using multiple level-of-detail (LOD) when appropriate can also improve runtime performance [18, 17, 30]. These LOD meshes can also reduce the complexity of the geometry while trying to retain the most important features, as determined by perceptual metrics.

However, there would be a number of challenges to overcome in designing a multimodal LOD system. The metrics defining important visual features are known to be different than the metrics defining important haptic features [19]. It remains an open problem to create metrics for selecting important audio features for switching between LODs in a multimodal system. Furthermore, the haptic LOD meshes are different from LOD meshes for visual rendering [19], leading to significantly higher memory requirements than texture-based representation in general.

5 Conclusion

In this paper, we presented an integrated system for multimodal interaction with textured surfaces. We demonstrated that normal maps can be used as a unified representation of fine surface detail for visual simulation of rigid body dynamics, haptic display and sound rendering. We showed that in a system which uses normal maps to present fine detail to users through multiple modes of interaction, users are able to combine this information to create a more cohesive mental model of the material they are interacting with. Our user evaluation result further provides validation that our system succeeded in reducing sensory conflict in virtual environments when using texture maps.

Normal maps serve as a good starting point for investigating the minimization of sensory conflict through a unified representation of fine detail. They are sufficient for recreating the physical phenomena described in this paper, but have some limitations. Haptic display rendering both forces and torques would need displacement maps in order to find the penetration depth [16] and to properly apply torques, though further optimization and acceleration would be required to incorporate such a technique into a fully multimodal system. Displacement maps or other texture maps could also further improve the contact point modification, finding more precise contact points with the displaced surfaces to compute the correct torque—though likely at a higher cost.

For future research, it may be possible to explore the integration of other texture representations, such as relief maps, displacement maps, etc., as well as incorporation material perception [22, 24] for multimodal display based on some of the principles described in this paper. We hope this work will lead to further interest in development of techniques on minimizing sensory conflicts when using texture representations for interactive 3D graphics applications, like VR and AR systems.

Acknowledgments

This research is supported in part by National Science Foundation and the UNC Arts and Sciences Foundation.

References

[1] J. F. Blinn. Simulation of wrinkled surfaces. *SIGGRAPH Comput. Graph.*, 12(3):286–292, Aug. 1978.

[2] J. Cohen, M. Olano, and D. Manocha. Appearance-preserving simplification. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, pages 115–122, New York, NY, USA, 1998. ACM.

[3] P. R. Cook and G. P. Scavone. The synthesis toolkit (stk). In *In Proceedings of the International Computer Music Conference*, 1999.

[4] R. L. Cook. Shade trees. In *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '84, pages 223–231, New York, NY, USA, 1984. ACM.

[5] K. V. D. Doel, P. G. Kry, and D. K. Pai. Foleyautomatic: Physically-based sound effects for interactive simulation and animation. In *in*

Figure 11: Lombard street diffuse map with normal map (left) and mapped to a plane with rolling balls (right).

Computer Graphics (ACM SIGGRAPH 01 Conference Proceedings, pages 537–544. ACM Press, 2001.

[6] N. Galoppo, S. Tekin, M. A. Otaduy, M. Gross, and M. C. Lin. Interactive haptic rendering of high-resolution deformable objects. In *Proceedings of the 2Nd International Conference on Virtual Reality*, ICVR'07, pages 215–233, Berlin, Heidelberg, 2007. Springer-Verlag.

[7] C.-H. Ho, C. Basdogan, and M. Srinivasan. A ray-based haptic rendering technique for displaying shape and texture of 3d objects in virtual environments. In *ASME Winter Annual Meeting*, 1997.

[8] C.-H. Ho, C. Basdogan, and M. A. Srinivasan. Efficient point-based rendering techniques for haptic display of virtual objects. *Presence: Teleoper. Virtual Environ.*, 8(5):477–491, Oct. 1999.

[9] T. Kaneko, T. Takahei, M. Inami, N. Kawakami, Y. Yanagida, T. Maeda, and S. Tachi. Detailed shape representation with parallax mapping. In *In Proceedings of the ICAT*, pages 205–208, 2001.

[10] T. H. Massie and J. K. Salisbury. The phantom haptic interface: A device for probing virtual objects. In *Proceedings of the ASME winter annual meeting, symposium on haptic interfaces for virtual environment and teleoperator systems*, volume 55, pages 295–300. Chicago, IL, 1994.

[11] M. Minsky, O.-y. Ming, O. Steele, F. P. Brooks, Jr., and M. Behensky. Feeling and seeing: Issues in force display. *SIGGRAPH Comput. Graph.*, 24(2):235–241, Feb. 1990.

[12] M. D. R. R. Minsky. *Computational Haptics: The Sandpaper System for Synthesizing Texture for a Force-feedback Display*. PhD thesis, Cambridge, MA, USA, 1995. Not available from Univ. Microfilms Int.

[13] S. Nykl, C. Mourning, and D. Chelberg. Interactive mesostructures. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, I3D '13, pages 37–44, New York, NY, USA, 2013. ACM.

[14] J. F. O'Brien, C. Shen, and C. M. Gatchalian. Synthesizing sounds from rigid-body simulations. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '02, pages 175–181, New York, NY, USA, 2002. ACM.

[15] M. M. Oliveira, G. Bishop, and D. McAllister. Relief texture mapping. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 359–368, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[16] M. Otaduy, N. Jain, A. Sud, and M. Lin. Haptic display of interaction between textured models. In *IEEE Visualization Conference*, pages 297–304, Oct 2004.

[17] M. A. Otaduy and M. C. Lin. CLODs: Dual hierarchies for multiresolution collision detection. *Eurographics Symposium on Geometry Processing*, pages 94–101, 2003.

[18] M. A. Otaduy and M. C. Lin. Sensation preserving simplification for haptic rendering. *ACM Trans. on Graphics (Proc. of ACM SIGGRAPH)*, pages 543–553, 2003.

[19] M. A. Otaduy and M. C. Lin. Sensation preserving simplification for haptic rendering. In *ACM SIGGRAPH 2005 Courses*, SIGGRAPH '05, New York, NY, USA, 2005. ACM.

[20] F. Policarpo, M. M. Oliveira, and J. a. L. D. Comba. Real-time relief mapping on arbitrary polygonal surfaces. In *Proceedings of the 2005 Symposium on Interactive 3D Graphics and Games*, I3D '05, pages 155–162, New York, NY, USA, 2005. ACM.

[21] N. Raghuvanshi and M. C. Lin. Interactive sound synthesis for large scale environments. In *Proceedings of the 2006 Symposium on Interactive 3D Graphics and Games*, I3D '06, pages 101–108, New York, NY, USA, 2006. ACM.

[22] Z. Ren, H. Yeh, R. Klatzky, and M. C. Lin. Auditory perceptiojn of geometry-invariant material properties. *Proc. of IEEE VR*, 2013.

[23] Z. Ren, H. Yeh, and M. Lin. Synthesizing contact sounds between textured models. In *Virtual Reality Conference (VR), 2010 IEEE*, pages 139–146, March 2010.

[24] Z. Ren, H. Yeh, and M. C. Lin. Example-guided physically-based modal sound synthesis. *ACM Trans. on Graphics*, 32(1):Article No. 1, January 2013.

[25] L. Szirmay-Kalos and T. Umenhoffer. Displacement mapping on the GPU - State of the Art. *Computer Graphics Forum*, 27(1), 2008.

[26] R. M. Taylor II, T. C. Hudson, A. Seeger, H. Weber, J. Juliano, and A. T. Helser. Vrpn: a device-independent, network-transparent vr peripheral system. In *Proceedings of the ACM symposium on Virtual reality software and technology*, pages 55–61. ACM, 2001.

[27] A. Tevs, I. Ihrke, and H.-P. Seidel. Maximum mipmaps for fast, accurate, and scalable dynamic height field rendering. In *Proceedings of the 2008 Symposium on Interactive 3D Graphics and Games*, I3D '08, pages 183–190, New York, NY, USA, 2008. ACM.

[28] V. Theoktisto, M. F. Gonzlez, and I. Navazo. Hybrid rugosity mesostructures (hrms) for fast and accurate rendering of fine haptic detail. *CLEI Electron. J.*, pages –1–1, 2010.

[29] K. van den Doel and D. K. Pai. The sounds of physical shapes. *Presence*, 7:382–395, 1996.

[30] S. Yoon, B. Salomon, M. C. Lin, and D. Manocha. Fast collision detection between massive models using dynamic simplification. *Eurographics Symposium on Geometry Processing*, pages 136–146, 2004.