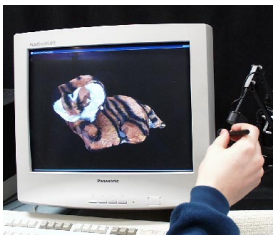
Scanning Physical Interaction Behavior of 3D Objects

Dinesh K. Pai, Kees van den Doel, Doug L. James, Jochen Lang, John E. Lloyd, Joshua L. Richmond, Som H. Yau

Department of Computer Science, University of British Columbia, Vancouver, Canada
{pai,kvdoel,djames,jlang,lloyd,jlrichmo,shyau}@cs.ubc.ca

(a) Real toy tiger. By design, it is soft to touch and exhibits significant deformation behavior.



(b) Deformable model of tiger scanned by our system, with haptic interaction.

(c) Real clay pot, with glazed regions. The pot exhibits a variety of contact textures and sounds.

(d) Virtual interaction with scanned model of pot; includes contact texture and contact sounds.

Figure 1: Examples of behavior models scanned by our system

Abstract

We describe a system for constructing computer models of several aspects of physical interaction behavior, by scanning the response of real objects. The behaviors we can successfully scan and model include deformation response, contact textures for interaction with force-feedback, and contact sounds. The system we describe uses a highly automated robotic facility that can scan behavior models of whole objects. We provide a comprehensive view of the modeling process, including selection of model structure, measurement, estimation, and rendering at interactive rates. The results are demonstrated with two examples: a soft stuffed toy which has significant deformation behavior, and a hard clay pot which has significant contact textures and sounds. The results described here make it possible to quickly construct physical interaction models of objects for applications in games, animation, and e-commerce.

Keywords: Behavioral Animation, Deformations, Haptics, Multimedia, Physically Based Modeling, Robotics, Sound Visualization

1 Introduction

Real 3D objects exhibit rich interaction behaviors which include how an object deforms on contact, how its surface feels when touched, and what kinds of sounds it makes when one interacts with it. These aspects of visual, haptic[1], and auditory behavior provide important interaction cues and increase the sense of presence in virtual environments such as games. Despite recent progress in deformation modeling (e.g., [37, 7, 18]), haptic rendering (e.g., [32]), and auditory displays (e.g., [12]), these aspects are either entirely missing from models used for computer graphics and interaction, or must be painstakingly added by highly skilled professionals. We believe that a major reason for this unfortunate situation is the difficulty of constructing these complex multimodal models by hand. In this paper we show how this problem can be solved by scanning the behavior of real objects.

Constructing behavior models requires not only acquiring measurements of real object behavior, but also designing mathematical models whose parameters can be effectively estimated from the measurements, and can be effectively used for realistic rendering. Each of these steps is important for successfully modeling real object behavior. We give detailed descriptions of how this can be done for three aspects of interaction behavior: (1) deformation models which can be rendered both visually and haptically; (2) contact texture models for capturing surface friction and roughness for haptic display and dynamics simulation; and (3) contact sound models for synthesizing interaction sounds, including sounds of continuous interaction.

Figure 1 shows some behavior models acquired by our system. Of course, it is somewhat difficult to show real time behavior on paper. The accompanying video demonstrates the behavior models better.

In this paper we also describe how the acquisition of these models can be automated using a highly integrated robotic measurement facility, and how behavior models can be registered relative to a geometric model of an object. Even though our facility is an expensive prototype and uses sophisticated robotics for interaction and behavior measurement, we believe it can be practical and economical for modeling

[1] haptics refers to the sense of touch.

because the facility can be shared by multiple users. The techniques we describe can be used to construct a *model foundry*, similar to a VLSI chip foundry but in reverse. Users could send real objects to such a foundry and receive in return a behavior model of the object.

It is helpful to compare the work presented here to 3D geometric modeling, from which we draw inspiration and instruction. 3D geometric models can be constructed by hand using 3D modeling software and for some special applications, this hand modeling and editing is essential. However, 3D scanning technology has dramatically changed the way such models are constructed. 3D geometry scanning makes it possible to capture fine details for unprecedented realism, but more importantly, empowers users with modest artistic talents with the ability to construct 3D models quickly and inexpensively. In a similar way, we expect that hand construction of interaction behavior will continue to be useful for some applications (e.g., those requiring fine creative control to match the context and narrative). But for many applications in games, animation, and e-commerce, the ability to construct highly detailed behavior quickly and inexpensively using the techniques we describe here could transform the way we construct models for 3D object interaction.

1.1 Related Work

We are not aware of any other system capable of scanning models of contact interaction behavior such as that described in this paper. However, each component of our system has many connections to previous work that we discuss in the relevant sections below. Here we briefly discuss related work in the general area of building models of 3D objects by measuring the real world.

Almost all the work in the area of modeling real world objects in computer graphics has focused on geometric modeling and reflectance modeling. In recent years there has been dramatic progress in scanning geometric models of 3D objects (e.g., [17, 8, 33, 22, 23]). In part because of the high accuracy of laser scanning, most of these techniques assume that the range data are given and focus on estimating good meshes from the data. Techniques have also been developed for geometry reconstruction from a few photographs [11] or even from a simple desktop system with a wand [2]. Acquiring the reflectance properties of existing objects has also been an active research area (e.g., [33, 9, 31, 15]). Our work has parallel goals with this type of automated model acquisition, but differs in terms of types of models acquired. Acquiring interaction models is more difficult because of the necessity of actually interacting and making contact with the objects to be measured.

Our work has some connections with image based techniques [4] and motion capture in that a recording can be considered a simple model of behavior, which can be edited and transformed (e.g., [26]). With few exceptions they have generally not been used for estimating parameters of physical models. They also do not account for inputs to the motion, and therefore can not be directly used for simulating the effects of new inputs.

Measuring the real world requires a certain amount of infrastructure and several groups have developed measurement facilities for this purpose. We mention the Cornell Light Measurement facility [15], the Columbia/Utrecht facility used for constructing a reflectance and texture database [9], the CMU Virtualized Reality^{TM} laboratory [41], the Berkeley Light Stage [10]. Field-deployable systems include the Digital Michelangelo project [23] and the IBM Pietà project [31].

Our own facility is perhaps the most highly automated and flexible system available today. It was designed to provide one-stop shopping for a large number of measurements rather than for highly accurate measurement. It required significant developments in robotics to control and coordinate the various measurement and actuation subsystems of the facility. We have previously described the robotics aspects of teleprogramming and motion planning for the system, and measurement techniques for sound and deformation (e.g., [25, 24]). However, the present paper is the first to describe the complete process of modeling interaction behaviors, from real object to rendered model.

2 A Framework for Modeling Behavior

To help understand how to construct useful models of real objects, it is helpful to break the task down into four steps: selection of model structure, measurement, parameter estimation, and rendering. In the following sections we detail how these steps are carried out for acquiring models of deformation, contact texture, and sound.

Selection of model structure

This defines the fundamental class of mathematical models that will be used to represent real physical behavior. In this step we fix the structure and not the parameters of the model.

We emphasize that this step is a creative act of model design, in which the modeler balances the competing needs of the accuracy of the model's predictions, rendering speed, ease of acquiring measurements, stability of parameter estimation, etc. It is tempting to think that the model is dictated by Physics and can be "looked up" in a textbook, but this is far from the truth. In designing the model structure it is essential to realize that all models have a finite domain of applicability.

Measurement

In this step we acquire the data from the real world to construct the model. This step is critical since all subsequent results depend on it. It is non-trivial for several reasons. First, we are interested in contact behavior which can not be simply observed but must be "excited" by physically interacting with the object. Thus we not only need to measure, say, the sound produced by an object, but we also need a way to hit the object in a carefully controlled manner to produce the sound. Second,traditional measurement techniques are designed for measuring material properties of small samples of objects, but we would like to build models of entire objects without destroying them or changing their essential global properties. Thus it is necessary to be able to move either the object or the measuring instruments or both, to access different parts of the object. Third, we need to register the different measurements relative to each other. We do this by registering all measurements relative to a geometric model of the object. For instance, this allows us to use the surface roughness of a particular point on the surface to drive the sound produced by scraping the surface at that point. However this means that we must first acquire a geometric model of the object prior to scanning the behavior. Finally, to interpret the raw data the instruments need to be calibrated. We do this using special calibration objects,

but auto-calibration techniques could also be used. To facilitate this kind of measurement we have built an integrated robotic measurement facility described in §3.

Parameter estimation

Estimation connects measurements to models. It is important to realize that estimation problems are frequently ill-posed and can lead to very sensitive inverse problems. Therefore it is necessary to pre-process the raw data, use regularization (which can be done in a natural way using Bayesian priors), and to use robust estimation techniques. As an example, direct fitting of sound models to sound waveforms by least-squares techniques produces very poor results in the presence of noise. We describe robust techniques for estimation which we found work well for the various estimation problems described below.

Rendering

Finally, the estimated models must be rendered, to produce deformations, haptic forces, and sounds. This step, of course, is the primary motivation for the previous steps and influences design decisions throughout. In particular, since we are interested in *interaction* behavior, it is important that the models are rendered at interactive rates. We describe implemented algorithms suitable for interactive rendering of deformation, forces, and sounds. The rendering step is also important for *validation* of the scanned model, i.e., determining whether the estimated model approximates reality sufficiently well for a given purpose. For computer graphics and interaction, it is difficult to design sensible quantitative metrics of model validity, and one must largely rely on user perception. We show results comparing the behavior of real objects to simulations of the scanned behavior models.

3 Measurement System for Interaction Behavior

Carefully acquiring the measurements required for model building is often the most challenging part of the modeling process. For acquiring the kind of contact measurements we need, we have developed the UBC Active Measurement facility (ACME), a highly automated robotic measurement facility. The robotics aspects of the facility have been previously described in detail [25]. For completeness, we briefly outline it here. We note, however, that the techniques described below do not depend on the use of this particular facility which was designed to provide a large variety of measurements; a simpler measurement set up could be constructed to build specific kinds of models. As with 3D geometry scanning, we expect that in the future it may be possible to build portable or desktop versions of such a measurement system.

Fig. 2 shows the facility which consists of a variety of sensors and actuators, all under computer control. Its major subsystems include: a 3 DOF Test Station (bottom of image) used for precise planar positioning of the test object; a Field Measurement System (shown at the top left) consisting of a Triclops trinocular stereo vision system, a high resolution RGB camera, and a microphone, all mounted on a 5 DOF positioning robot; and a Contact Measurement System (CMS, shown on the right) consisting of a Puma 260 robot equipped with a force/torque sensor, mounted on a linear stage. The CMS is the key subsystem for contact measurements as it is used to interact with the object to be modeled. The entire facility can be controlled from any location on the Internet. We will describe the use of the sensors and actuators for measurement in the relevant sections below.

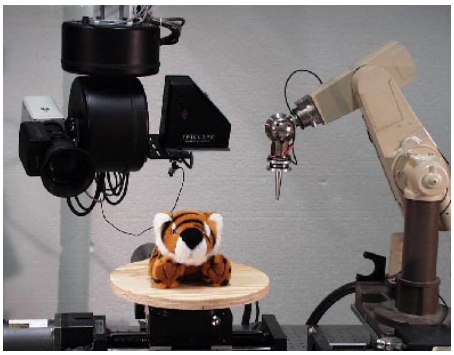


Figure 2: ACME Facility Overview

4 Geometric Modeling

The geometric models required for registering other measurements can be produced by any geometry scanning and mesh generation method (e.g., [17, 8, 33, 28, 23]). While the focus of this paper is not on geometric modeling, for completeness we will briefly outline our approach to geometric model construction starting from stereo range measurements.

Stereo range measurement

The main shape sensor in our measurement facility is a trinocular stereo vision system[2] which is capable of producing large amounts of viewpoint-registered range images at modest resolutions (approx. 2-3 mm for a typical viewpoint), in close to real time. Accurate ranging relies on image features for matching between the stereo cameras; additional features can be attached or projected onto the surface.

Stereo range data is notoriously noisy, and for best results it is filtered in several ways. First, range is calculated with variable mask sizes in a voting scheme. We also utilize the checks and filters of the Triclops stereo library. Further processing of the data by volumetric reconstruction requires approximate surface normals at range data points. The normals are estimated by plane fitting in local image neighborhoods. Further filtering of the range-data is performed based on the quality of fit, the number of valid range-data per neighborhood and the viewing angle of the plane.

Multresolution mesh construction

An initial triangle mesh is generated from the filtered range data using a volumetric approach by reconstruction software provided courtesy of NRC of Canada [28]. The number and quality of triangles depends on the surface sampling density. Further processing is required to arrive at a useful geometric model since this mesh may not be watertight; it may include some of the supporting Test Station surface; and there may

[2] A Color Triclops from Point Grey Research.

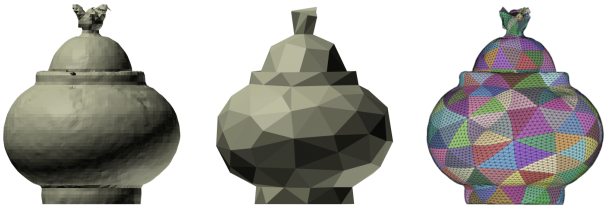


Figure 3: *Clay Pot Mesh Reconstruction:* (left) raw scan mesh (13150 vertices, 26290 faces) with various external stray polygons; (middle) watertight base level (level 0) mesh (127 vertices, 250 faces) generated via simplification of the raw scan mesh; (right) level 3 subdivision connectivity mesh (8002 vertices, 16000 faces) generated using displaced subdivision surface style piercing of the raw scan mesh.

be erroneous surfaces due to noisy data. An example is shown in the left image of Figure 3.

Finally, we construct watertight subdivision connectivity triangle meshes at several resolutions $l = 0, 1, 2, \ldots, L$ because they are desirable for later modeling. Here we exploit a common property of raw meshes produced from range data: while the region exterior to the object geometry may contain undesirable mesh features, the interior of the mesh is a fair representation of the scanned surface. This allows the construction of multiresolution meshes by expanding and refining a surface inside the object using a normal piercing process similar to the construction of displaced subdivision surfaces [21] and normal meshes [16]. The coarse resolution mesh is generated by simplifying the raw mesh (using QSlim [13]) and possibly some minor editing to ensure that the mesh is free of defects such as in regions with poor range data. To ensure that the mesh expansion process is bounded, the raw mesh is manually closed if the NRC package produces a large hole on the unimaged underside of the object; this was addressed for the tiger mesh by simply inserting a large horizontal triangle.

While this approach to mesh construction is not robust, it produces meshes of sufficient quality that we were able to proceed with our main physical modeling objectives.

Texture mapping

Texture mapping is accomplished using calibrated color images from the stereo vision system. The vertices of the meshes are projected into the color images using the pinhole camera calibration matrix, and triangles are tagged as visible if their vertices and centroid are all visible. We select the imaged triangular area as the texture map if the product of the image area times the cosine between view direction and triangle normal is maximum in all our images. We also adjust the relative global brightness of all color texture images. The texture maps could be readily improved by blending of the local textures over the entire map eliminating abrupt brightness changes at edges, as in [27].

5 Deformation Modeling

Selection of model structure

We use linear elastostatic models for deformable objects since these models can be simulated at interactive rates [18], and linearity makes parameter estimation easier. Most important, linear elastic models can be characterized using Green's functions, which capture the input-output behavior of the object's boundary for a given set of boundary conditions; therefore there is no need to observe or estimate quantities in the interior of the object.

It is a practical consideration that the model must be fixtured for it to be deformed by ACME's robots. Therefore we start by assuming that the physical object will be measured (and rendered) while attached to a support, like the Test Station. The linear elastostatic model then approximates the displacement response $\mathsf{u} = \mathsf{u}^{(l)}$ of the resolution l mesh vertices due to *applied* surface tractions[3] $\mathsf{p} = \mathsf{p}^{(l)}$ by

$$\mathsf{u} = \mathsf{U}\mathsf{p} \qquad \text{or} \qquad \mathsf{u}^{(l)} = \mathsf{U}^{(l)}\mathsf{p}^{(l)}. \tag{1}$$

Here u and p are block vectors of length n with 3-vector elements, where the displacement and traction at the k^{th} vertex is u_k and p_k, respectively; U is a square $n^{(l)}$-by-$n^{(l)}$ block matrix with 3-by-3 matrix elements. The k^{th} block column $\mathsf{U}_{:k}$ describes the displacement response contribution from the k^{th} applied traction p_k. The diagonal blocks U_{kk} describe the self-compliance of the k^{th} vertex, and play an important role in defining vertex stiffnesses for force-feedback rendering [19]. The traction vector at a vertex is the force over the area of the one-ring of a vertex. The force is distributed linearly over the area as a hat function located at the vertex.

The key behavior of the model is characterized by the displacement response of the unfixtured free surface due to forces applied to the free surface. This corresponds to the only interesting measurable portion of the U matrix, which in turn is related to the discrete *Green's function matrix* [19] for the free boundary. For instance, rows of U corresponding to vertices attached to the fixtured support necessarily have zero displacement values; these same vertices correspond to columns which are not exposed and therefore can not be actively inspected.

Finally, once the Green's function matrix for a fixture configuration is known, deformations of the object can be simulated efficiently using fast matrix updating techniques [19].

Measurement

Objects to be measured, such as the stuffed toy tiger shown in Figure 1, are fixed to the Test Station. Deformation measurements record surface displacements and contact forces resulting from active probing with ACME's CMS robot arm. During deformation the position of the contact probe can be continuously monitored at 100 Hz. The robot arm's wrist force sensor is capable of recording the six-dimensional force-torque wrench at the tip of the probe at 1000 Hz.

The robot probes surface locations corresponding to vertices of the geometric model at the desired reconstruction resolution[4] l. The position of the contact probe measures the displacement of the contacted vertex u_k, while the force-torque sensor measures the contact force. The effective contact traction p_k is the force divided by the effective vertex area (one third of the sum of adjacent triangle areas). Since there are no other *applied* tractions on the free or fixed surfaces, the complete traction vector p is zero except for the single contact traction p_k. In the following we describe how the deformation of the free surface is measured and mapped onto the vertices in order to estimate u.

[3] Traction is the force per unit area, similar to pressure.

[4] To avoid measurement artifacts the scale of the mesh is larger than the measurement probe's contact area.

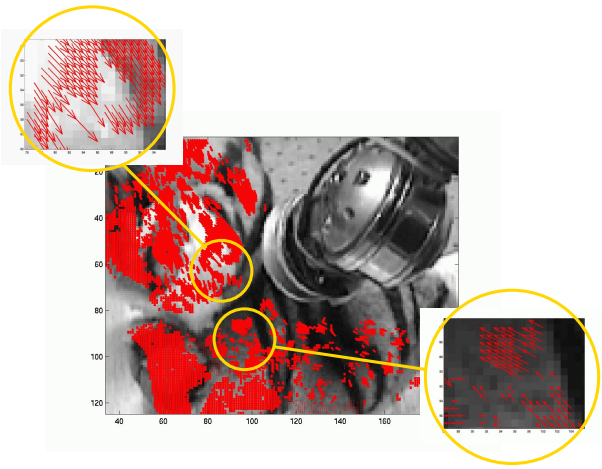


Figure 4: 2D Flow on tiger, poked near the neck.

The surface deformation of the object is measured visually using the Triclops trinocular stereo-vision system in the Field Measurement System (FMS) of ACME. The measurement is based on tracking visual surface features in three spatial dimensions in a range-flow [42] approach. Several methods for calculating dense range-flow from stereo and optical flow exist [41, 43, 34].

Our method utilizes redundancy in the imagery from the trinocular stereo-system to increase robustness [20]; It is summarized as:

- Segment image into "object surface" and "other," based on the geometric model of the undeformed object.
- Calculate range from stereo.
- Calculate simultaneous optical flow in images from each camera for the "object surface".
- Combine optical flow by voting and map the result into three dimensions based on range data.
- Use the "range-flowed" object surface to segment the next image in the sequence and continue with stereo calculations as above.

The optical flow during deformation measurement is shown in Figure 4. In our approach, the surfaces need sufficient visual texture because of the reliance on stereo vision and optical flow. Most objects have sufficient visual texture for stereo matching but if not non-permanent visual texture may be applied (e.g., using pins, stickers, or water soluble paint).

The next step in surface deformation measurement is the mapping of range flow vectors to displacements of vertices of a chosen mesh level. Flow vectors are associated with a start position on a triangular patch of the undeformed mesh. We estimate the displacement of a vertex with a robust averaging process rather than just using the closest flow vector to a vertex (see, for example, [1] for a discussion on robust motion estimation). The flow vectors associated with triangular patches joined at a vertex are median filtered. This is followed by a weighted average based on distance from the vertex. The flow vectors have to be dense enough on the surface relative to the mesh level for this process to be robust. In our set-up, we get high density by positioning the camera close to the object ($\approx 0.8m$). The measured displacement u covers surface area visible from a chosen view point. In the estimation section below, we discuss how to combine multiple measurements for the same contact vertex.

Parameter estimation

Our approach to the estimation of the matrix U from the measurements of displacement u and traction p addresses two main issues: (1) noise in the form of outliers in the displacement measurement and (2) incomplete measurements. Outliers can, on occasion, still be observed in the displacement measurement despite the above described filtering process. These outliers originate from consistent incorrect flow vectors due to mismatches in the range-flow or in the stereo processing over an image area. Incomplete measurements arise from partially observed object surfaces, from reachability limits of the CMS and from anisotropic responses of the object to probing.

A single block element U_{ij} describes the relationship between the displacement u_i and a single applied traction p_j:

$$\mathsf{u}_i = \mathsf{U}_{ij}\mathsf{p}_j.$$

For each element, we obtain $m \leq M$ measurements by probing each vertex location M times. We arrive at a standard least squares problem to be solved for U_{ij}^T:

$$\left[\mathsf{p}_j^1 \mathsf{p}_j^2 \cdots \mathsf{p}_j^m\right]^T \mathsf{U}_{ij}^T = \left[\mathsf{u}_i^1 \mathsf{u}_i^2 \cdots \mathsf{u}_i^m\right]^T$$

We solve this least squares problem if our measured tractions are a response to a set of probe directions which span 3-space. This guarantees that we excite the model in all directions. However, because of the possibility of (in general) noncompliant responses combined with the limited resolution of the measurements, this does not guarantee that the solution $(\mathsf{U}_{ij})^T$ has full rank. Therefore, we calculate the solution by means of the truncated singular value decomposition (TSVD). We truncate the singular values at the approximate resolution of the traction measurements. Furthermore, we select a subset of measurements if we observe an unsatisfactory fit of the estimated $(\mathsf{U}_{ij})^T$ and repeat. This subset selection process is performed using the least trimmed squares (LTS) method [29], implemented efficiently as in [30]. This process is robust even if $(M-4)/2$ measurements are spoiled.

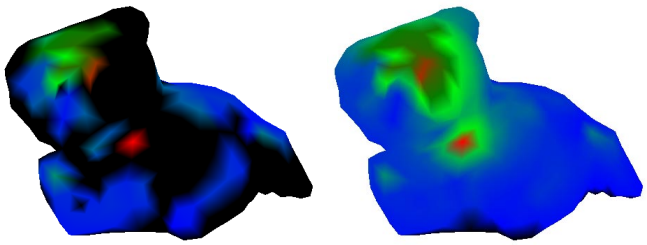


Figure 5: *Plots of estimated displacement responses:* (left) Missing observations result in unestimated response components (shown in black); the remaining nodes are color coded with red indicating the greatest displacement and blue the least. (right) These values are estimated by an interpolating reconstruction to obtain the final deformation responses.

At this stage of the estimation process, the measured displacement field columns of U still contain unestimated

elements due to missing observations of the deformed surface (shown in Figure 5). This problem can be minimized by obtaining more measurements but not entirely avoided. Scattered data reconstruction is used to fill in elements for each column individually. We currently interpolate missing displacements by solving Laplace's equation over the set of unestimated vertices, but better methods are currently being investigated. The result of this interpolation process is shown in Figure 5.

Finally, in order to improve rendering quality and reduce measurement and estimation time we exploit the multiresolution mesh structure to optionally infer Green's function responses for unmeasured vertices. This is done by actively poking the model at a resolution $(l-1)$ one level coarser than the resolution l used to estimate displacement fields (illustrated in Figure 6). The k^{th} odd vertex on level l has a response $\mathsf{U}_{:k}$ inferred if both even vertices (k_1, k_2) of its parent edge have responses. If so, the k^{th} response $\mathsf{U}_{:k}$ is linearly interpolated from the two parent responses, $(\mathsf{U}_{:k_1}, \mathsf{U}_{:k_2})$. The local responses, U_{kk} and U_{jk} when vertex j is a one-ring neighbor of k, are handled differently.

Unlike long range displacement influences which are smoothly varying, these local values are associated with a cusp in the displacement field. Simple interpolation for these values is biased and leads to incorrect contact forces during rendering. Instead, the local values are computed as the weighted average of parent responses which have had their local parameterizations smoothly translated from even vertex k_* to odd vertex k, e.g., U_{kk} is linearly interpolated from $(\mathsf{U}_{k_1k_1}, \mathsf{U}_{k_2k_2})$ not $(\mathsf{U}_{kk_1}, \mathsf{U}_{kk_2})$. This shifting of the parent's local response before averaging yields a good estimator of the local response at vertex k. The resulting displacement field $\mathsf{U}_{:k}$ is also linearly independent of $\mathsf{U}_{:k_1}$ and $\mathsf{U}_{:k_2}$.

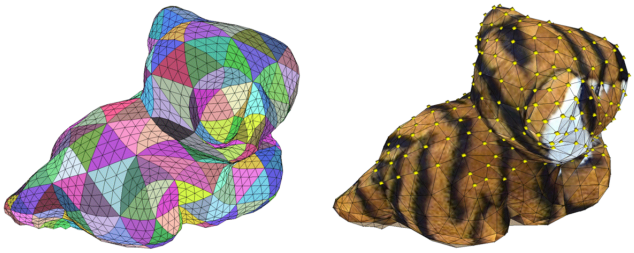


Figure 6: *Multiresolution Mesh and Contact Sampling Pattern:* (left) Coarse $l = 0$ parameterization of model, used for active contact measurement, displayed on finest $l = 2$ displaced subdivision surface mesh; (right) yellow points drawn on the $l = 1$ resolution mark the nodes at which the system's displacement response to applied tractions was either measured (even vertices) or inferred (odd vertices).

Rendering

By design, the Green's function models can be rendered at interactive rates using the algorithm described in [18, 19]. Contact forces are rendered using a PHANToM force-feedback interface with contact force responses computed using vertex pressure masks [19]. The multiresolution deforming surface is also displacement mapped using *displaced subdivision surfaces* [21] to add extra geometric detail to the model. Figure 1 and the accompanying video (and the CAL demonstration) show interaction with scanned tiger model using the PHANToM. In general the results are quite satisfactory, capturing non-local effects such as the movement of the head when the back of the tiger is poked. The model does show some of the limitations of the linear model structure for large input displacements, with somewhat exaggerated deformations. For moderate input displacements[5], the scanned model behaves quite realistically.

6 Contact Texture

Selection of model structure

Contact texture denotes the way an object feels when it is rubbed or scraped. The two principal aspects we focus on in this paper are friction and surface roughness.

For modeling friction, we use the standard "textbook" Coulomb friction model

$$\mathbf{f}_f \;=\; -\mu \,\|\mathbf{f}_n\|\mathbf{u}_m \tag{2}$$

where $\mathbf{f}_f$ is the frictional force, μ is the coefficient of friction (and the model parameter to be determined), $\mathbf{f}_n$ is the normal force, and $\mathbf{u}_m$ is a unit vector in the direction of motion.

Surface roughness is a more elusive property and whole books have been written on how to model it [38]. Roughness is usually associated with small-scale variations in the surface geometry, which create variations in the tangential contact force proportional to the normal force. These tangential force variations can be modeled as local variations $\tilde{\mu}$ in the coefficient of friction. Combined with μ, this yields an *effective* coefficient of friction μ_e for a given displacement x along some particular surface direction:

$$\mu_e(x) = \mu + \tilde{\mu}(x).$$

This formulation has the advantage of unifying haptic rendering of friction and roughness, particulary with commercial haptic devices like the PHANToM which implement their own contact and friction algorithms which may not correspond to the textbook model of Coulomb friction.

Forces due to roughness tend to have some randomness but often also contain periodic components, particularly in human artifacts. We assume that the roughness is isotropic and that people are sensitive only to statistical features of the roughness force variation, and can not discern the specific waveform. To capture both randomness and periodicity we model the friction variations $\tilde{\mu}(x)$ as an autoregressive AR(p) process, driven by noise, so that

$$\tilde{\mu}(x) = \tilde{\mu}(k\Delta) \equiv \tilde{\mu}(k) = \sum_{i=1}^{p} a_i \tilde{\mu}(k-i) + \sigma\epsilon(k)$$

where k is the sample index, Δ is the spatial discretization, σ is the standard deviation of the input noise, and $\epsilon(k)$ is a zero-mean noise input with standard deviation of one. The model parameters to be determined are the a_i and σ.

The AR model is very suitable for real-time simulation and rendering, and typically one needs only a few parameters to reflect the roughness properties (as illustrated by Fig. 7). An AR(2) model is often sufficient in practice because it allows the modeling of a random sequence combined with one principal frequency.

[5] approximately $< 15\%$ of the tiger's diameter

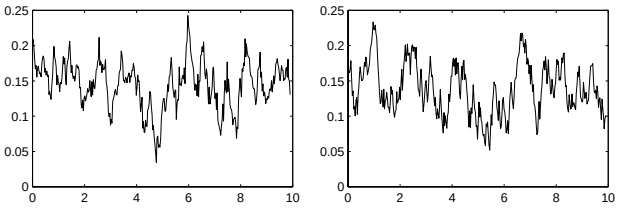


Figure 7: (left) Measured values of μ_e for a 10 mm displacement along a rough section of the clay pot shown in Fig. 8. (right) Simulation of μ_e with $\mu = 0.142$ and $\tilde{\mu}$ reproduced by an AR(2) model with $a_1 = .783$, $a_2 = .116$, and $\sigma = 0.0148$.

Measurement and Estimation

Friction and surface roughness are noticeable in terms of the forces they produce on a contacting instrument. Hence we can measure them in the same way: the robotic system performs a series of local rubs over the object surface with a probe (attached to a 6 DOF force sensor; Fig. 8) and the resulting force profiles are then analyzed.

The object's surface mesh representation is used to plan "where and how" to do the rubbing. At present, the system assigns a contact texture model to each mesh vertex. This is determined either by explicit measurement, or by the interpolation of models at nearby vertices. The system employs a process of "active exploration", in which models are initially sampled over the object surface at a low resolution, with further, higher resolution sampling in areas where the model parameters change significantly.

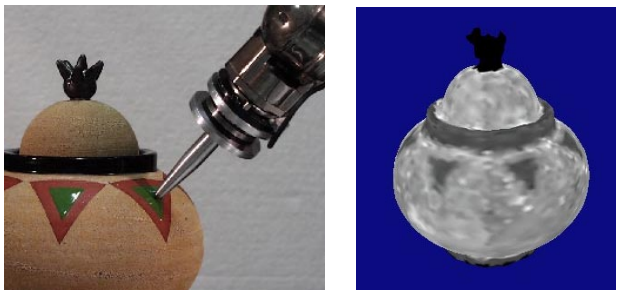


Figure 8: (left) Robotic system rubbing the pot to determine contact texture. (right) Surface friction map for the clay pot, in which the brightness is scaled according to the local value of μ on the surface (with white corresponding to the maximum value of $\mu = 0.5$). The enameled portions of the pot, with $\mu \approx 0.09$, are clearly visible. The ornament on top of the pot was not sampled.

The nominal friction coefficient μ is estimated first. If the surface normal $\mathbf{n}$ were known accurately, one could directly determine $\mathbf{f}_n$ and $\mathbf{f}_f$ in Eq.(2) and use this to solve for μ. However, $\mathbf{n}$ is known only approximately (due to uncertainty in the surface mesh and the actual contact location), and also varies along the path. To compensate for this, we stroke the surface *twice*: once in a forward direction and once in a reverse direction. At any point along the path, we then have a force value $\mathbf{f}^+$ which was measured during the forward motion, and another value $\mathbf{f}^-$ which was measured during the reverse motion.

$\mathbf{f}^+$ and $\mathbf{f}^-$ each have components parallel to the surface normal $\mathbf{n}$, along with friction components $\mathbf{f}_f^+$ and $\mathbf{f}_f^-$ which lie opposite to the motion directions and are perpendicular to $\mathbf{n}$. Now even if $\mathbf{n}$ is unknown, and the magnitudes of $\mathbf{f}^+$ and $\mathbf{f}^-$ differ, μ can still be estimated from the angle θ between $\mathbf{f}^+$ and $\mathbf{f}^-$:

$$\mu = \tan(\theta/2).$$

This calculation is quite robust, as it is independent of travel speed, the probe contact force and orientation, and of course the surface normal itself. By averaging the values of μ obtained at various points along the path, a reasonable estimate for μ over the whole path may be obtained. Our ability to determine μ reliably is illustrated in Fig. 8.

The values of $\mathbf{n}$ at each path point can also be estimated from the direction of $(\mathbf{f}_f^+ + \mathbf{f}_f^-)$ and used to produce a smooth (typically quadratic) model of $\mathbf{n}(x)$ along the path.

To calculate the effective friction, we use $\mathbf{n}(x)$ to relate μ_e to the observed force values $\mathbf{f}$ acting on the probe tip:

$$\mathbf{f} = f_n[\mathbf{n}(x) - \mu_e \mathbf{u}_m(x)],$$

where $\mathbf{u}_m(x)$ is the direction of motion along the path and f_n is the magnitude of the normal force. The unknowns in this equation are f_n and μ_e. Solving for μ_e at every path point x yields $\mu_e(x)$. An AR model is then fitted to $\tilde{\mu}(x) = \mu_e(x) - \mu$, using autoregressive parameter estimation via the covariance method (e.g., the `arcov` function in MATLAB).

Rendering

To demonstrate the rendering of contact texture, we used a PHANToM haptic device to implement a virtual environment in which a user can rub an object with a point contact (represented as a red ball in the video). The GHOST software supplied with the PHANToM was used to perform collision detection and to generate the corresponding contact and frictional forces.

The friction value at the point of contact is generated by weighting the AR parameters at each vertex by the barycentric coodinates in the triangle. The distance traveled along the surface divided by the spatial discretization Δ of the measurements determines the number of values to generate using the AR model. The last 2 values generated are then interpolated to obtain the effective coefficient of friction μ_e. This value is passed to both the static and dynamic friction parameters in GHOST.

The resulting contact textures are quite convincing; it is easy to distinguish between the different surface preparations of the clay pot using the haptics alone. These results are best evaluated using the PHANToM haptic device (e.g., in our CAL demo) though Figs. 7 and 8 give a good indication of the sensitivity of our measurement technique.

7 Sound Modeling

Selection of model structure

We model the contact sounds of an object by filtering an excitation (the "audio-force") through a modal resonator bank which models the sonic response of the object. The details of this technique are explained in [39]. For this we need to acquire both a modal resonance model of the object, which will depend on its shape and internal composition, and an excitation model, which will depend mainly on the surface structure.

The modal model $\mathcal{M} = \{\boldsymbol{f}, \boldsymbol{d}, \boldsymbol{A}\}$, consists of a vector $\boldsymbol{f}$ of length N whose components are the modal frequencies in Hertz, a vector $\boldsymbol{d}$ of length N whose components are the (angular) decay rates in Hertz, and an $N \times K$ matrix $\boldsymbol{A}$, whose elements a_{nk} are the gains for each mode at different locations. The modeled response for an impulse at location k is given by

$$y_k(t) = \sum_{n=1}^{N} a_{nk} e^{-d_n t} \sin(2\pi f_n t), \tag{3}$$

for $t \geq 0$ (and is zero for $t < 0$).

The surface texture generating the audio-force can be modeled in real-time by filtering an appropriate noise source with the location dependent autoregressive filter models obtained from the surface measurements described in Section 6.

For audio we need to know the audio surface texture at a much higher resolution than for haptics texture modeling. In the future we plan to measure the surface properties at higher resolutions and use AR(p) models acquired automatically. We have verified that such an approach yields good sound models but have not yet integrated this with the rest of the system. For now, we acquire audio-resolution surface properties by hand.

For the pot example shown in the accompanying video, we manually segment the surface into areas of substantially different textures and generate an excitation force from recordings made with a contact microphone at a reference speed and force and store them in wave-tables, just like audio signals. This approach is analogous to image-based rendering; as described below, the recorded excitation can be transformed by a few run-time interaction parameters to produce a realistic audio-force [39].

Measurement

One way to estimate the modal model is to excite (i.e., hit) the object with an arbitrary force and measure both the audio response and the input force at the same high rate, and deconvolve the input force from the audio signal. This is the approach followed in [6]. However this can be delicate because measuring forces at audio frequencies requires very stiff force sensors, since otherwise the force signal can be contaminated by the resonances of the sensor itself. Deconvolution is also a numerically sensitive inverse problem. We have chosen instead to build a device for applying a light, highly peaked force which is a good finite approximation of an impulsive force; the measured audio signal can then be treated as the impulse response and used directly for parameter estimation. The device consists of a small push-type solenoid mounted at the tip of the robot arm; the solenoid is activated for a brief period so that the small plunger moves and hits the object ballistically and bounces off. The far field sound is recorded at 44.1 KHz using microphones mounted on the field measurement system. Fig. 9 shows the device pinging the clay pot. The robot systematically pings the object at the vertices of the base mesh. Several recordings are made at each mesh vertex for better estimation.

Parameter estimation

We have developed a technique for estimating the modal model $\mathcal{M}$ from the recorded impulse responses. The number of modes to extract is manually set to a large value and we discard the modes with very low gain which will not contribute to the sound. Precisely how many modes we want to use for synthesis depends on factors such as the desired accuracy of the reconstruction.

Figure 9: Contact sound measurement

The modal frequencies are first estimated from the average power spectrum of the recordings (corrected for background noise) using peak identification with a quadratic interpolation of the discrete windowed Fourier transform. For a typical window size of $20ms$ this gives frequencies with errors of about $50Hz$. This initial estimate is then refined by performing a phase reconstruction by fitting complex frequency trajectories of the windowed Fourier transforms of the signals with a sum of a small number of damped exponentials using the Steiglitz-McBride algorithm [36]. This will provide us with the estimated couplings $\boldsymbol{A}$, the dampings $\boldsymbol{d}$, and corrected estimates of the frequencies $\boldsymbol{f}$. In some cases very closely spaced modes arise because of approximate symmetries of the object which we resolve by fitting each trajectory with multiple complex exponentials. This "phase unwrapping" procedure has been used before to obtain very accurate frequency estimates [3]. Our application differs in that we are interested also in the dampings and coupling amplitudes, and we also want to be able to resolve densely spaced frequencies into their separate components.

We have tested the accuracy of the parameter estimation on artificially constructed impulse responses in the form of Eq. 3 and found that the frequencies and dampings have errors no larger than 0.0001%, and the gains have errors of about 1%. See Fig. 10.

Rendering

During simulation, an audio kernel filters the audio-force excitation — which is parameterized by contact parameters such as velocity and friction — through the modal resonator bank and produces audio in real time. The audio-force can be a short impulse for impacts, or a noise-like excitation for scraping. Filtering with a modal reson bank can be computed very efficiently with an $O(N)$ algorithm [14, 5, 40] for a model of N modes. Details of our contact sound rendering techniques are described in [39].

The geometrical locations on the surface of the object are mapped to points in the "timbre-space" of the object, which we define as the space spanned by the gain vectors $\boldsymbol{a}$. This is done by associating gains a_{nk} with each mesh vertex k at which the sounds were sampled during the measurement. In this manner we model the observed timbre shifts in the sound

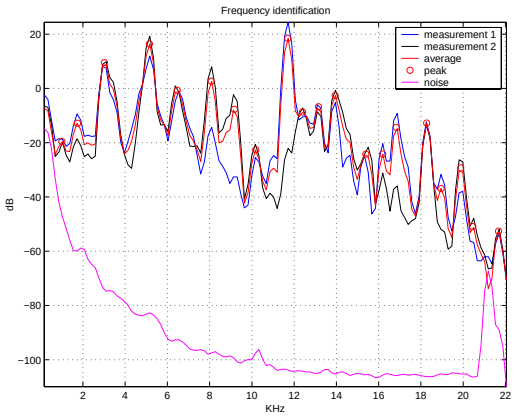


Figure 10: The power spectrum of two recorded impulse responses, their average, and the power spectrum of the background noise. The 20 most important peaks are indicated on the graph. The "best" peaks are those considered to stand out from their surrounding most clearly by "local height"

when an object is excited at different locations. Sudden jumps in the timbre during scraping can be heard clearly if we switch gain vectors discretely. We therefore utilize a form of "audio anti-aliasing": at locations between mesh vertices we smoothly interpolate the gains from the vertex gains, using the barycentric coordinates of the location in the triangle. Note that because the gains a_{nk} at different vertices share the same modal frequencies, there is no need for frequency interpolation.

If the AR(2) filters turn out to be resons [35], we can scale the resonance frequency measured at a reference contact speed with the actual contact speed in the simulation. This produces the effect of a shift in "pitch" dependent on the sliding velocity. If the AR(2) filters are not resons (i.e., if their poles are real), which will occur if there is no prominent characteristic length scale in the surface profile, this model does not produce the illusion of scraping at a changing speed. The perceptual cue for the contact speed seems to be contained in the shifting frequency peak. We have found that higher order AR models in such cases will find these peaks, but we have not completed this investigation at the time of writing.

If an audio-force wave-table is used, it is pitch-shifted using linear sample interpolation to correspond to the actual simulation contact speed and the volume is adjusted proportional to the power-loss as determined by friction and speed [39].

8 Conclusions

We have described a system for modeling the interaction behavior of 3D objects by scanning the behavior of real objects. Modeling interaction behavior is essential for creating interactive virtual environments, but constructing such models has been difficult. We show how a variety of important interaction behaviors, including deformation, surface texture for contact, and contact sounds can be effectively scanned. We provided a description of the complete modeling process which could be used construct these types of models. We also described our own measurement facility which automates many of the steps in measuring contact interaction behavior using robotics.

We believe that the techniques described in this paper could greatly improve the way virtual environments and animations are created. In addition to geometry and appearance, our methods will allow behaviors to be essential and easily obtained properties of virtual objects. Our methods make it feasible to build compelling interactive virtual environments populated with a large number of virtual objects with interesting behavior.

References

[1] M.J. Black and P. Anandan. The Robust Estimation of Multiple Motions: Parametric and Piecewise-smooth Flow Fields. *Computer Vision and Image Understanding*, 63(1):75–104, 1996.

[2] Y. Bouguet and P. Perona. 3D Photography on Your Desk. In *Proc. ICCV98*, pages 43–50, 1998.

[3] J.C. Brown and M.S. Puckette. A High Resolution Fundamental Frequency Determination Based on Phase Changes of the Fourier Transform. *J. Acoust. Soc. Am.*, 94(2):662–667, 1993.

[4] M. Cohen, M. Levoy, J. Malik, L. McMillan, and E. Chen. Image-based Rendering: Really New or Deja Vu? ACM SIGGRAPH 97 Panel, pages 468 - 470.

[5] P.R. Cook. Integration of Physical Modeling for Synthesis and Animation. In *Proceedings of the International Computer Music Conference*, pages 525–528, Banff, 1995.

[6] P.R. Cook and D. Trueman. NBody: Interactive Multidirectional Musical Instrument Body Radiation Simulations, and a Database of Measured Impulse Responses. In *Proceedings of the International Computer Music Conference*, San Francisco, 1998.

[7] S. Cotin, H. Delingette, J-M Clement, V. Tassetti, J. Marescaux, and N. Ayache. Geometric and Physical Representations for a Simulator of Hepatic Surgery. In *Proceedings of Medicine Meets Virtual Reality IV*, pages 139–151. IOS Press, 1996.

[8] B. Curless and M. Levoy. A Volumetric Method for Building Complex Models from Range Images. In *SIGGRAPH 96 Conference Proceedings*, pages 303–312, 1996.

[9] K.J. Dana, B. van Ginneken, S.K. Nayar, and J.J. Koenderink. Reflectance and Texture of Real-world Surfaces. *ACM Transactions on Graphics*, 18(1):1–34, 1999.

[10] P. Debevec, T. Hawkins, C. Tchou, H-P Duiker, W. Sarokin, and M. Sagar. Acquiring the Reflectance Field of a Human Face. In *SIGGRAPH 2000 Conference Proceedings*, pages 145–156, 2000.

[11] P.E. Debevec, C.J. Taylor, and J. Malik. Modeling and Rendering Architecture from Photographs: A Hybrid Geometry- and Image-Based Approach. In *SIGGRAPH 96 Conference Proceedings*, pages 11–20, 1996.

[12] T.A. Funkhouser, I. Carlbom, G. Pingali G. Elko, M. Sondhi, and J. West. Interactive Acoustic Modeling of Complex Environments. *J. Acoust. Soc. Am.*, 105(2), 1999.

[13] M. Garland and P.S. Heckbert. Surface Simplification Using Quadric Error Metrics. In *SIGGRAPH 97 Conference Proceedings*, pages 209–216, 1997.

[14] W.W. Gaver. Synthesizing Auditory Icons. In *Proceedings of the ACM INTERCHI 1993*, pages 228–235, 1993.

[15] D.P. Greenberg, K.E. Torrance, P. Shirley, J. Arvo, J.A. Ferwerda, S. Pattanaik, E.P.F. Lafortune, B. Walter, S. Foo, and B. Trumbore. A Framework for Realistic Image Synthesis. In *SIGGRAPH 97 Conference Proceedings*, pages 477–494, 1997.

[16] I. Guskov, K. Vidimce, W. Sweldens, and P. Schroder. Normal Meshes. In *SIGGRAPH 2000 Conference Proceedings*, pages 95–102, 2000.

[17] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, and W. Stuetzle. Piecewise Smooth Surface Reconstruction. In *SIGGRAPH 94 Conference Proceedings*, pages 295–302, 1994.

[18] D.L. James and D.K. Pai. ArtDefo, Accurate Real Time Deformable Objects. In *SIGGRAPH 99 Conference Proceedings*, pages 65–72, 1999.

[19] D.L. James and D.K. Pai. A Unified Treatment of Elastostatic Contact Simulation for Real Time Haptics. *Haptics-e, The Electronic Journal of Haptics Research (www.haptics-e.org)*, 2001. (To appear).

[20] J. Lang and D. K. Pai. Estimation of Elastic Constants from 3D Range-Flow. In *Proceedings of the Third International Conference on 3-D Digital Imaging and Modeling*, 2001.

[21] A. Lee, H. Moreton, and H. Hoppe. Displaced Subdivision Surfaces. In *SIGGRAPH 2000 Conference Proceedings*, pages 85–94, 2000.

[22] A. Lee, W. Sweldens, P. Schröder, L. Cowsar, and D. Dobkin. MAPS: Multiresolution Adaptive Parameterization of Surfaces. In *SIGGRAPH 98 Conference Proceedings*, pages 95–104, 1998.

[23] M. Levoy, et al. The Digital Michelangelo Project: 3D Scanning of Large Statues. In *SIGGRAPH 2000 Conference Proceedings*, pages 131–144, 2000.

[24] D. K. Pai, J. Lang, J. E. Lloyd, and J. L. Richmond. Reality-based Modeling with ACME: A Progress Report. In *Proceedings of the Intl. Symp. on Experimental Robotics*, 2000.

[25] D. K. Pai, J. Lang, J. E. Lloyd, and R. J. Woodham. ACME, A Telerobotic Active Measurement Facility. In *Proceedings of the Sixth Intl. Symp. on Experimental Robotics*, 1999.

[26] Z. Popovic and A. Witkin. Physically Based Motion Transformation. In *SIGGRAPH 99 Conference Proceedings*, pages 11–20, 1999.

[27] C. Rocchini, P. Cignoni, C.Montani, and P. Scopigno. Multiple Textures Stitching and Blending on 3D Objects. In *10th Eurographics Workshop on Rendering*, pages 173–180, Granada, Spain, 1999.

[28] G. Roth and E. Wibowoo. An Efficient Volumetric Method for Building Closed Triangular Meshes from 3-D Image and Point Data. In *Proc. Graphics Interface*, pages 173–180, 1997.

[29] P.J. Rousseeuw. Least Median of Squares Regression. *Journal of the American Statistical Association*, 79(388):871–880, 1984.

[30] P.J. Rousseeuw and K. Van Driessen. Computing LTS Regression for Large Data Sets. Technical report, University of Antwerp, 1999.

[31] H. Rushmeier, F. Bernardini, J. Mittleman, and G. Taubin. Acquiring Input for Rendering at Appropriate Levels of Detail: Digitizing a Pietà. *Eurographics Rendering Workshop 1998*, pages 81–92, 1998.

[32] D.C. Ruspini, K. Kolarov, and O. Khatib. The Haptic Display of Complex Graphical Environments. In *SIGGRAPH 97 Conference Proceedings*, pages 345–352, 1997.

[33] Y. Sato, M.D. Wheeler, and K. Ikeuchi. Object Shape and Reflectance Modeling from Observation. In *SIGGRAPH 97 Conference Proceedings*, pages 379–388, 1997.

[34] H. Spies, B. Jähne, and J.L. Barron. Dense Range Flow from Depth and Intensity Data. In *International Conference on Pattern Recognition*, pages 131–134, 2000.

[35] K. Steiglitz. *A Digital Signal Processing Primer with Applications to Digital Audio and Computer Music*. Addison-Wesley, New York, 1996.

[36] K. Steiglitz and L.E. McBride. A Technique for the Identification of Linear System. *IEEE Trans. Automatic Control*, AC-10:461–464, 1965.

[37] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer. Elastically Deformable Models. In *Computer Graphics (SIGGRAPH 87 Proceedings)*, volume 21, pages 205–214, 1987.

[38] T.R. Thomas. *Rough Surfaces*. Imperial College Press, London, second edition, 1999.

[39] K. van den Doel, P.G. Kry, and D.K. Pai. FoleyAutomatic: Physically-based Sound Effects for Interactive Simulations and Animations. In *SIGGRAPH 2001 Conference Proceedings*, 2001.

[40] K. van den Doel and D.K. Pai. The Sounds of Physical Shapes. *Presence*, 7(4):382–395, 1998.

[41] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade. Three-dimensional Scene Flow. In *International Conference on Computer Vision*, pages 722–729, 1999.

[42] M. Yamamoto, P. Boulanger, J.-A. Beraldin, and M. Rioux. Direct Estimation of Range Flow on Deformable Shape from a Video Rate Range Camera. *PAMI*, 15(1):82–89, 1993.

[43] Y. Zhang and C. Kambhamettu. Integrated 3D Scene Flow and Structure Recovery from Multiview Image Sequences. In *Computer Vision and Pattern Recognition*, volume 2, pages 674–681, 2000.