Image-Space Modal Bases for Plausible Manipulation of Objects in Video

Abe Davis[1] Justin G. Chen[2] Frédo Durand[1]

[1]MIT CSAIL [2]MIT Dept. of Civil and Environmental Eng.

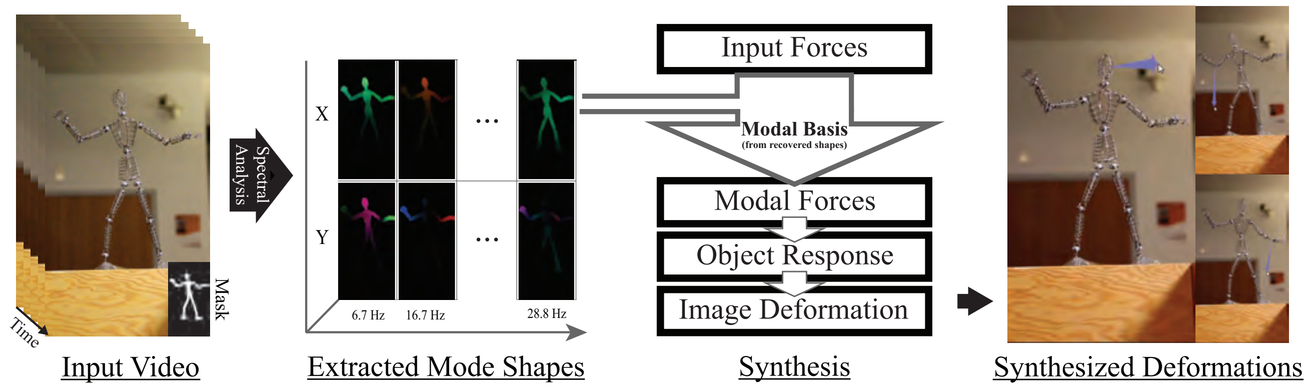


Figure 1: *By extracting the vibration modes of a wire figure from small deformations in a five second video captured with an SLR, we are able to create an interactive 2D simulation of the figure. Left: an image from the input video showing the object at its rest state, with a rough mask shown in the bottom right corner. Middle: deformation modes extracted from the x and y dimensions of the video at different frequencies. Right: synthesized deformations of the object responding to user-defined forces.*

Abstract

We present algorithms for extracting an image-space representation of object structure from video and using it to synthesize physically plausible animations of objects responding to new, previously unseen forces. Our representation of structure is derived from an image-space analysis of modal object deformation: projections of an object's resonant modes are recovered from the temporal spectra of optical flow in a video, and used as a basis for the image-space simulation of object dynamics. We describe how to extract this basis from video, and show that it can be used to create physically-plausible animations of objects without any knowledge of scene geometry or material properties.

CR Categories: I.4.7 [Image Processing and Computer Vision]: Scene Analysis—Time-varying Imagery;

Keywords: Video, Physically based animation, Video Synthesis, Video Textures, Modal Analysis, Animation, Interactive

1 Introduction

Computational photography seeks to capture richer information about the world, and provide new visual experiences. One of the most important ways that we experience our environment is by manipulating it: we push, pull, poke, and prod to test hypotheses about our surroundings. By observing how objects respond to forces that we control, we learn about their dynamics. Unfortunately, video does not afford this type of manipulation - it limits us to observing the dynamics that were recorded. However, in this paper we show that many videos contain enough information to locally predict how recorded objects will respond to new, unseen forces. We use this information to build image-space models of object dynamics around a rest state, letting us turn short video clips into physically-plausible, interactive animations.

Most techniques for physically-based animation derive the properties that govern object dynamics from known virtual models. However, measuring these properties for objects in the real world can be extremely difficult, and estimating them from video alone is severely underconstrained. A key observation of our work is that there is often enough information in video to create a physically plausible model of object dynamics around a rest state in which the object is filmed, even when fundamental ambiguities make recovering a general or fully-accurate model impossible. We show how to extract these physically plausible models from short video clips, and demonstrate their use in two applications.

Interactive Animation: Video makes it easy to capture the appearence of our surroundings, but offers no means of physical interaction with recorded objects. In the real world, such interactions are a crucial part of how we understand the physical properties of objects. By building a model of dynamics around the state in which an object is filmed, we turn videos into interactive animations that users can explore with virtual forces that they control.

Special Effects: In film special effects, where objects often need to respond to virtual forces, it is common to avoid modeling the dynamics of real objects by compositing human performances into virtual environments. Performers act in front of a green screen, and their performance is later composited with computer-generated objects that are easy to simulate. This approach can produce compelling results, but requires considerable effort: virtual objects must be modeled, their lighting and appearence made consistent with any real footage being used, and their dynamics synchronized with a live performance. Our work addresses many of these challenges by making it possible to apply virtual forces directly to objects as they appear in video.

1.1 Overview

Our approach is based on the same linear modal analysis behind many techniques in physically-based animation. However, unlike most of these techniques, we do not assume any knowledge of object geometry or material properties, and therefore cannot rely on finite element model (FEM) methods to derive a modal basis for simulation. Instead, we observe non-orthogonal projections of an object's vibration modes directly in video. For this we derive a relationship between projected modes and the temporal spectra of optical flow. We then show that, while non-orthogonal, these projections can still be used as a basis to simulate image-space object dynamics.

Recovering accurate physical models of objects in video is severely underconstrained. To deal with this ambiguity, we make a few key assumptions, which we analyze in Section 3.3.

2 Related Work

Physically-based Animation: Many techniques in physically-based animation use modal analysis to reduce the degrees of freedom in deformable body simulations [Pentland and Williams 1989; James and Pai 2002; James and Pai 2003; Pai et al. 2001; Huang et al. 2011; Li et al. 2014]. These techniques work by first deriving orthogonal vibration modes from known geometry using FEM approaches. As high frequency modes generally contribute less to an object's deformation, they can often be discarded to obtain a lower-dimensional basis for faster simulation. We use a similar reduced modal basis to simulate objects in video, but assume no knowledge of scene geometry and cannot therefore use FEM approaches to compute vibration modes. Instead, we observe projections of these modes directly in video and show that, while non-orthogonal, these projections can still be used as a basis to simulate the dynamics of objects in image-space.

Observing Vibration Modes The problem of directly observing vibration modes has been explored in several engineering disciplines, where the structure of objects must be carefully validated in the real world, even when a virtual model is available. The general approach is to relate the spectrum of surface motion, typically measured with accelerometers, to mode shapes. [Helfrick et al. 2011] applied this analysis to motion estimated with a stereo rig, which they used to recover mode shapes for shell-like structures.

Recent work in graphics and vision has used narrow-band phase-based motion magnification to visualize the modal vibrations of objects in video [Wadhwa et al. 2013; Wadhwa et al. 2014; Chen et al. 2015]. [Davis et al. 2014; Davis et al. 2015] proposed an alternative visualization based on the temporal spectra of weighted optical flow. However, both approaches focus on providing a visualization tool, and neither has been used to recover a basis for simulation. We show that a similar algorithm, borrowing aspects of each of these visualization techniques, can be used to recover mode shapes that are suitable for simulation.

Motion Synthesis in Video: Several works in computer graphics and vision have focused on synthesizing plausible animations of quasi-periodic phenomena based on a video exemplar [Doretto et al. 2003; Szummer and Picard 1996; Chuang et al. 2005; Schödl et al. 2000; Pentland and Sclaroff 1991; Tao and Huang 1998]. In most of these applications, video synthesis is formulated as a stochastic process with parameters that can be fit to the exemplar. Such approaches work especially well for animating phenomena like rippling water or smoke, and with skeletal information provided by a user have even been extended to model the motion of structures caused by stochastic forces like wind [Stam 1996; Sun et al. 2003]. The applications we address are similar to many of these works in spirit, but, to our knowledge, we are the first to build image-space simulations based on a modal bases extracted directly from video.

Motion Magnification Like recent publications in motion magnification [Wadhwa et al. 2013; Wadhwa et al. 2014; Chen et al. 2015], our work can be used to magnify and visualize small vibrations of an object. However, our work is different from motion magnification in several key ways. First, while motion magnification is a time-varrying representation of motion, our technique extracts a static representation of each vibration mode, and can therefore average over the entire input video to reduce noise at each mode. Second, while phase-based methods for Eulerian motion magnification rely on expensive pyramid decompositions of video at render time, our approach to synthesis is Lagrangian and can be implemented efficiently on the GPU, allowing for real-time synthesis of motion composed of many vibration modes. Finally, while motion magnification can only magnify motion already present in a captured video, our technique can synthesize responses to new combinations of forces that were never observed in the input.

3 Modal Analysis

Here we connect the image-space deformations of an object to established modal analysis. Section 3.1 reviews some of the relevant theory from linear modal analysis (more detail can be found in [Shabana 1991; Bathe 2006]). Section 3.2 connects this theory to the observed deformations of an object in video and provides a theoretical basis for the algorithms described in Section 4.

3.1 Object Motion

The dynamics of most solid objects under small deformations are well approximated by a finite element model representing a system of masses, dampers, and springs. We assume that objects undergo small deformations around a fixed rest state. The matrices $\mathbf{M}$, $\mathbf{C}$, and $\mathbf{K}$ represent mass, damping, and stiffness relationships between an object's degrees of freedom, and the equation of motion in response to a force $\mathbf{f}(t)$ is given by

$$\mathbf{M}\ddot{\mathbf{u}}(t) + \mathbf{C}\dot{\mathbf{u}}(t) + \mathbf{K}\mathbf{u}(t) = \mathbf{f}(t), \tag{1}$$

where $\ddot{\mathbf{u}}(t)$, $\dot{\mathbf{u}}(t)$, and $\mathbf{u}(t)$ are vectors for acceleration, velocity, and displacement. Assuming sinusoidal solutions to Equation 1, the eigenmodes of this system are the orthogonal solutions to the generalized eigenvalue problem given by $\mathbf{K}\phi_i = \omega_i^2\mathbf{M}\phi_i$. The set of eigenvectors or eigenmodes $\phi_1 ... \phi_N$ define a modal matrix $\mathbf{\Phi}$ shown in Equation 2 which diagonalizes the mass and stiffness matrices into modal masses $\mathbf{m}_i$ and modal stiffnesses $\mathbf{k}_i$.

$$\mathbf{\Phi} = \begin{bmatrix}\phi_1 & \phi_2 & \ldots & \phi_N\end{bmatrix} \tag{2}$$

$$\mathbf{\Phi}^T\mathbf{M}\mathbf{\Phi} = diag(\mathbf{m}_i) \tag{3}$$

$$\mathbf{\Phi}^T\mathbf{K}\mathbf{\Phi} = diag(\mathbf{k}_i) \tag{4}$$

The matrix $\mathbf{\Phi}$ defines modal coordinates $\mathbf{q}(t)$ where $\mathbf{u}(t) = \mathbf{\Phi}\mathbf{q}(t)$. In these modal coordinates the equations of motion are decoupled into single degree of freedom systems defined by modal masses $\mathbf{m}_i$, damping $\mathbf{c}_i$, stiffnesses $\mathbf{k}_i$, and forces $\mathbf{f}_i(t) = \phi_i^T\mathbf{f}(t)$. Under the common assumption of Rayleigh damping, modal damping can be expressed by $\mathbf{c}_i = \alpha\mathbf{m}_i + \beta\mathbf{k}_i$ giving the decoupled equation of motion for each mode

$$\ddot{\mathbf{q}}(t) + 2\xi_i\omega_i\dot{\mathbf{q}}(t) + \omega_i^2\mathbf{q} = \frac{\mathbf{f}_i}{\mathbf{m}_i} \tag{5}$$

where the undamped natural frequency is $\omega_i = \sqrt{\frac{\mathbf{k}_i}{\mathbf{m}_i}}$, giving the modal damping factor

$$\xi_i = \frac{\mathbf{c}_i}{2\mathbf{m}_i\omega_i} = \frac{1}{2}\Big(\frac{\alpha}{\omega_i} + \beta\omega_i\Big) \tag{6}$$

We can then obtain the unit impulse response for the i^{th} mode by solving Equation 5

$$h_i(t) = \Big(\frac{e^{-\xi_i\omega_i t}}{\mathbf{m}_i\omega_{di}}\Big)sin(\omega_{di}t) \tag{7}$$

where the damped natural frequency is $\omega_{di} = \omega_i\sqrt{1-\xi_i^2}$. Given Equation 7 we can construct the response of an object to an arbitrary impulse as the superposition of that object's 1D modal responses.

Taking the Fourier transform of the unit impulse response $h_i(t)$ the product in Equation 7 becomes the convolution

$$H_i(\omega) = \Big(\frac{1}{\mathbf{m}_i\omega_{di}}\frac{\xi_i\omega_i}{\xi_i^2\omega_i^2 + \omega^2}\Big) * \Big(\frac{\delta(\omega - \omega_{di}) - \delta(\omega + \omega_{di})}{i}\Big) \tag{8}$$

which convolves the Fourier transform of the decaying exponential, a Lorentzian distribution; and a pair of delta functions. In other words, the transfer function of a single mode is the convolution of a spike at its resonant frequency and a Lorentzian with a width that depends on modal frequency and damping.

3.2 Eigenmodes in Image-Space

In this section we relate deformations observed in video to projections of the mode shapes ϕ_i and show that these projections can be used as a basis for representing image-space dynamics. We first consider the dynamics of a single degree of freedom, which we later relate to the motion of a visible point in video.

An excitation force $\mathbf{f}$ given in modal coordinates can be decomposed into a set of impulses $\mathbf{f}_i = A_i\delta(t)$ where A_i is the amplitude of the impulse at mode ϕ_i. Applying Equation 7, the response of the object at one degrees of freedom $u_p(t)$ is given by

$$u_p(t) = \sum_{i=1}^{N} A_i h_i(t)\phi_i(p) \tag{9}$$

where $\phi_i(p)$ is the mode shape coefficient of the degree of freedom p of the object for mode i. Using Equations 8 and 9 we can construct the Fourier transform of Equation 9 as

$$U_p(\omega) = \sum_{i=1}^{N} A_i H_i(\omega)\phi_i(p) \tag{10}$$

Here we make an assumption that is common in engineering modal analysis [De Roeck et al. 2000; Brincker et al. 2003], but not necessary in FEM-based applications of modal analysis for simulation: that modes are well spaced, or non-overlapping in the frequency domain. Under this assumption, we can represent the frequency response of a single degree of freedom at ω_{di} as

$$U_p(\omega_{di}) = A_i H_i(\omega_{di})\phi_i(p). \tag{11}$$

Our next assumption is weak perspective - a common approximation in computer vision, but one that is also not necessary when modes are derived from known models. Using this approximation we align our object's coordinate system with the image plane of an input video, giving us observable degrees of freedom for each pixel's motion in the x and y dimensions of our image. For the purpose of derivation, we represent visibilty across all degrees of freedom with the unknown, binary, diagonal matrix $\mathbf{V}$, which multiplies the visible degrees of freedom in a mode by 1 and all other degrees of freedom by 0. The projection of a mode shape ϕ_i into the image plane is then $\mathbf{V}\phi_i$.

By taking Fourier transforms of all local motions $\mathbf{Vu}$ observed in video we obtain $\mathbf{VU}$, the Fourier spectra for visible degrees of freedom, which, evaluated at resonant frequencies ω_{di}, is

$$\mathbf{VU}(\omega_{di}) = A_i H_i(\omega_{di})\mathbf{V}\phi_i. \tag{12}$$

Here, A_i and $H_i(\omega_{di})$ are constant across all degrees of freedom p, meaning that $\mathbf{VU}(\omega_{di}) \propto \mathbf{V}\phi_i$. Therefore we can treat the set of complex ϕ_i', the values of $\mathbf{VU}(\omega_{di})$ measured in video, as a basis for the motion of the object in the image plane.

3.3 Assumptions and Limitations

While linear motion is a standard assumption of linear modal analysis that usually applies to the type of small motion we are analyzing, our derivation makes a few key approximations that are not typical of modal analysis applied to simulation:

- *Weak Perspective* - We assume that linear motion in 3D projects to linear motion in the image plane. This can be violated by large motion in the z-plane.
- *Well-spaced modes* - We rely on separation in the frequency domain to decouple independent modes. This can fail in objects with strong symmetries, high damping, or independent moving parts.
- *Broad-Spectrum Forcing* - By using observed modes as a basis for the motion of an object in the image plane, we make an implicit assumption about the ratio of modal masses to observed modal forces. Allowing for an ambiguity of global scale, this assumption is still violated when observed forces are much stronger at some modes than others.

Because we deal with small motion around a rest state, weak perspective is generally a safe approximation. However, there are many cases where our remaining two assumptions could fail. Fortunately, the consequences of these failures tend to affect the accuracy more than the plausability of simulation. Consider the failure cases of each approxiation. Overlapping modes will cause independent objects to appear coupled in simulation - in other words, the response of an object to one force will incorrectly be an otherwise appropriate response to multiple forces. Similarly, when broad-spectrum forcing is violated, the response of a object to one force will be the appropriate response to a differently scaled, but equally valid set of forces. In both cases, the failure results in inaccurate, but still plausible deformations of the object.

4 Algorithm

Our algorithms first extracts a volume of candidate vibration modes from an input video. We then provide a user interface for selecting a subset of these candidate modes to use as a basis for simulation.

4.1 Extracting Candidate Modes

We measure optical flow in the x and y dimensions of an input video using phase variations of a complex steerable pyramid [Simoncelli et al. 1992]. This approach has been shown to work well for small motion in several recent works [Wadhwa et al. 2013; Wadhwa et al. 2014; Davis et al. 2014; Davis et al. 2015], though Lagrangian flow algorithms may be equally well suited to our application. To filter local displacements, we employ the weighted gaussian filtering used in [Wadhwa et al. 2013]. Local displacements are first given weights proportional to local contrast. The weighted displacements and the weights are both blurred spatially, then the filtered displacements are normalized by their filtered weights. This denoises displacement signals by causing regions with low image contrast, and therefore noisy displacements, to take on the values of nearby regions with high image contrast.

Next we compute the temporal FFT of our filtered displacement signals as in Davis [2014; 2015]. Each spatial slice of the resulting temporal frequency spectra forms a candidate shape for a possible mode at that frequency.

4.2 Mode Selection:

Under ideal conditions, the observed candidate modes ϕ'_ω at each frequency ω would be zero everywhere but at real mode shapes. However, real video contains unintended motion from a variety of sources (e.g., camera shake, noise, moving background). To distinguish between object deformations and unintended motion from other sources, we first ask users to provide a rough mask of the content they are interested in. We then present them with a graphical interface to help select mode shapes.

Our mode selection interface (shown in Figure 2) displays a representative image from the input video, a power spectrum showing the magnitude of motion observed at each frequency, and a visualization of the current selected candidate mode, chosen by the user. The power spectrum shows the average amplitude of unmasked coefficients in each candidate mode shape. In very recent work, [Davis et al. 2015] showed that resonant modes of an object can be identified as peaks in a similar power spectrum (though the spectra they use are based on the motion signals described in [Davis et al. 2014]). When a user clicks on the spectrum in our interface, we find the frequency with maximum energy in a small window around the user's mouse, and display the corresponding candidate mode in our shape window. We use the same visualization of candidate mode shapes described in [Davis et al. 2014] - phases are mapped to hue, and magnitudes are mapped to intensity. Users can select either a set or a range of modes by clicking on different peaks in the power spectrum. This selection process is similar to peak-picking methods that have been used for modal identification of structures in engineering [De Roeck et al. 2000]. Informed users are generally able to select a suitable set of mode shapes in less than a minute, though some training to know how to identify 'good' mode shapes is necessary. For a video of mode selection refer to our supplemental material.

4.3 Complex Mode Shapes:

Note that the set of mode shape solutions ϕ_i to Equation 1 are real-valued, i.e. they only have binary phase relationships. Similarly, the mode shapes derived using FEM in typical simulation applications are also real-valued. In contrast, the mode shapes we recover may have non-binary phases. This can happen for a number of reasons, including noise or a violation of one of our assumptions. We could force mode shapes to be real-valued by projecting them onto their dominant axis in the complex plane, however, we found that allowing non-binary phases actually improves results. Visually, such mode shapes allow for features like traveling waves and partial coupling that might otherwise require much higher-order modes to represent. By allowing these shapes, we effectively let our representation fit the motion in a video more closely. In this sense, our technique is allowed to behave a little more like methods for exemplar-based motion texture synthesis in situations where motion cannot be explained well with sparse, low-frequency modes.

To ensure that the behavior of our simulation reduces to one using only real mode shapes when observed modes contain only binary phase relationships, we calculate the dominant orientation of each selected mode shapes on the complex plane, and rotate all phases so that this orientation aligns with the real axis.

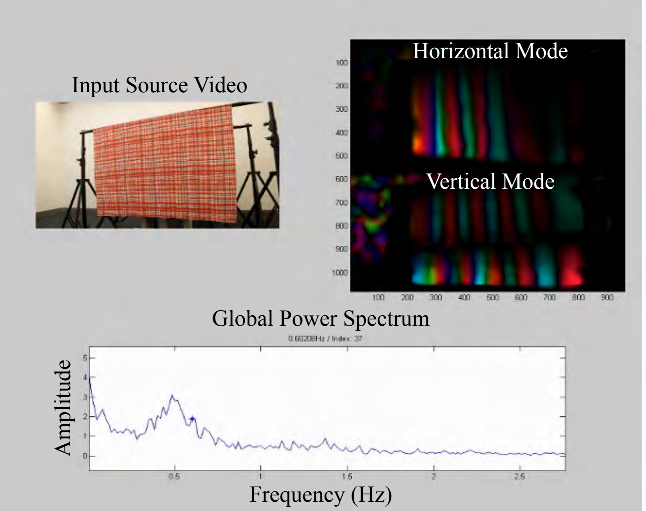


Figure 2: *To use our mode selection interface, users click on a frequency in the video's motion spectrum (bottom) and are shown a visualization at the corresponding candidate mode shape (right). Using this interface users can select either an individual, or a range of candidate modes to use as a basis for simulation.*

4.4 Simulation

Our simulation works on the state of an object in modal coordinates. The key components are a way to evolve the state of an object over time, and a way to translate user input into forces, displacements, and velocities.

Given Equation 5, we can define a state space model per modal coordinate to simulate the the object over time. We define the state vector $\mathbf{y_i}$ that describes the system for a single modal coordinate $\mathbf{y_i} = [\varrho_\mathbf{i} \;\; \dot{\varrho}_\mathbf{i}]^\intercal$, where $\varrho_\mathbf{i}$ and $\dot{\varrho}_\mathbf{i}$ are the modal displacement and velocity vectors respectively which relate to the complex modal coordinate by $\mathbf{q_i} = \varrho_\mathbf{i} - i\dot{\varrho}_\mathbf{i}/\omega_\mathbf{i}$. We evolve the state to $\mathbf{y}[n+1]$ given

$\mathbf{y}[n]$ and a modal force $\mathbf{f_i}$ using the equation[1]:

$$\mathbf{y}[n+1] = \begin{bmatrix} 1 & h \\ -\omega_i^2 h & 1 - 2\xi_i\omega_i h \end{bmatrix} \mathbf{y}[n] + \begin{bmatrix} 0 \\ h/\mathbf{m}_i \end{bmatrix} \mathbf{f}_i[n], \quad (13)$$

and set h, the amount of time passed in the simulation, to be small enough to ensure that this equation is stable.

4.5 User Input

We provide users with modes of interaction that can be divided into two categories: forcing interactions and direct manipulations. Forcing interactions affect state indirectly by changing the force $\mathbf{f}_i$ applied to an object. Direct manipulations translate user input directly into intantaneous state $\mathbf{y}$.

Forcing Interactions: Forcing interactions translate user input into a force to be applied at a specified point. In the simplest forcing interaction, a user clicks at a point $\mathbf{p}$ on the object, and drags their mouse in a direction $\mathbf{d}$. We interpret this as specifying a force $\mathbf{f}$ to be applied at the point $\mathbf{p}$ in the direction $\mathbf{d}$. The scalar modal force $\mathbf{f_i}$ applied to each mode is computed by taking the magnitude of the dot product of $\mathbf{d}$ with the value of that mode shape ϕ'_i at point $\mathbf{p}$:

$$\mathbf{f_i} = \|\mathbf{d}^\intercal \phi'_i(\mathbf{p})\|\alpha \quad (14)$$

where α is used to control the strength of the force, and can be set by the user with a slider. Note that we take the magnitude here because the mode shape ϕ'_i is complex.

Direct Manipulation: Real objects are often found in configurations that are difficult or impossible to achieve through forces applied to one point at a time. However, fully specifying shaped forces is a difficult user interaction problem. We instead offer a mode of interaction that lets users directly manipulate the position or velocity of a single point. This lets users explore states with greater contributions from higher-order modes that are difficult to achieve without shaped forces. We accomplished this by explicitly setting the state of the object whenever the user's mouse is pressed, and only letting the state evolve once the mouse is released. As with forcing interactions, the user specifies a point $\mathbf{p}$ and direction $\mathbf{d}$ with a mouse. We then compute the magnitude of each modal coordinate in the same way that we computed the magnitude of modal forces before:

$$\|\mathbf{q_i}\| = \|\mathbf{d}^\intercal \phi'_i(\mathbf{p})\|\alpha \quad (15)$$

where α is used to control the strength of the manipulation, and can be set by the user with a slider. However, in this case we set the phase of the modal coordinate to maximize either the displacement or velocity of $\mathbf{p}$ in the direction $\mathbf{d}$. This is accomplished by setting the phase $Arg(\mathbf{q_i})$ to

$$\text{Max Displacement:} \quad Arg(\mathbf{q_i}) = -Arg(\mathbf{d}^\intercal \phi'_i(\mathbf{p})) \quad (16)$$

$$\text{Max Velocity:} \quad Arg(\mathbf{q_i}) = -Arg(\mathbf{d}^\intercal \phi'_i(\mathbf{p})) + \frac{\pi}{2} \quad (17)$$

For objects with real mode shapes, velocity is maximized when displacements are zero, and displacement is maximized when velocities are zero. Intuitively, maximizing displacement lets users 'pull' a point around the screen and see how the the object deforms in response, while maximizing velocity specifies an impulse to be applied when the mouse is released.

4.6 Rendering Deformations

We render the object in a given state by warping a single color image, representing the object's rest state, by a displacement field $\mathbf{D}(t)$. $\mathbf{D}(t)$ is calculated as a superposition of mode shapes weighted by their respective modal coordinates:

$$\mathbf{D}(t) = \sum_i^N \mathbf{Re}\{\phi'_i q_i(t)\} \quad (18)$$

This can be evaluated efficiently on the GPU by representing each ϕ'_i as an RGBA texture storing two complex numbers per pixel, corresponding to the coupled image-space x and y displacements of ϕ'_i. Each $\phi'_i q_i(t)$ term is computed in a single rendering pass, accumulating $\mathbf{D}_t$ in a framebuffer that can be applied as a displacement map to the color image in a final pass. Our implementation uses depth culling and assigns pixels depth values that are inversely proportional to the magnitude of their displacement, causing parts of the image that move more to occlude parts that move less. This tends to work better than blending pixel values in practice, as objects closer to the camera usually exhibit larger screen space motion due to foreshortening.

Note that rendering deformations with our algorithm is substantially faster than previous work on motion magnification, and can run in realtime. This is because, unlike previous work on motion magnification, we do not rely on the complex steerable pyramid at render time.

4.7 Implementation Details

Our mode extraction and selection interface are written in MATLAB. Once modes have been selected, they are exported as 8-bit RGBA TIFF images, and loaded into our simulation software, which is written in C++ and uses Qt, OpenGL, and GLSL.

The slowest part of our algorithm is building a complex steerable pyramid on the input video. Using the MATLAB implementation from [Simoncelli et al. 1992] this takes less than two minutes on shorter videos like the Wireman, but can take 2-3 hours on longer, or high-speed videos like the Ukulele. The only parameter we set for this is the standard deviation of the gaussian used for filtering local motion signals. Our strategy for setting this parameter is to effectively test out 4 values at once - we pick a standard deviation that is 5-10% of the larger image dimension, filter with this standard deviation at all scales, and use the highest-resolution scale that does not appear noisy. Mode selection can then usually be done in less than a minute, but users may choose to spend more time exploring the recovered spectra with our selection interface.

In the Playground, YoutubeBridge, and ForceTree examples we use inpainting to fill disoccluded parts of the image.

5 Results

We tested our method on several different examples. Thumbnails showing the rest statea of several examples can be found in Table 2 along with additional details about the corresponding input video.

All of the input videos that we captured were recorded with a tripod. The input video for YoutubeBridge was downloaded from Youtube user KOCEDWindCenter (link).

Note that motion blur is not typically a problem for our method for several reasons. First, we deal with small motions, so motion blur is not common. Second, we only need one sharp frame of the

[1] A derivation of this equation can be found in [Shabana 1991]

object in its resting state. In practice, motion blur tends to help mode recovery by acting as a pre-filter to prevent temporal aliasing.

Our simulations plausibly reproduce the behavior observed in most input videos. Our method works well with regular cameras operating at 30 frames per second. While higher-frequency modes exist in most objects, their fast temporal dynamics are not usually visible in input videos. Our Ukulele example explores the use of a high speed camera to recover modes that are not visible at normal framerates.

Our supplemental material also includes a metal beam example, captured with a cell phone camera, where we compare motion from our interactive simulation to a real video of the beam being struck by a hammer.

Interactive Animations Video showing interactive sessions with our examples can be found in the supplemental material. In each interactive session, an arrow is rendered to indicate where users click and drag. The head of the arrow points to the current mouse location, and the tail of the arrow ends at the displaced point $\mathbf{p}$ where the user initially clicked.

For the most part, interactive animations are quite compelling. However, in some cases where our non-overlapping modes assumption is violated, independent parts of a scene appear coupled. This effect is subtle in most of our results, so we include an additional failure case designed to violate this assumption in our supplemental material (labeled 'dinos1'). The example shows two dinosaur toys with similar motion spectra resting on the same surface. When a user interacts with one of the toys, this causes some motion in the other toy as well. This problem could be addressed in the future by asking users to provide multiple masks, indicating independent parts of the scene.

Special Effects A variety of visual effects can be be achieved by specifying forces in different ways. We explore the possibility of using this to create low-cost special effects. For example, by using forcing interactions and setting $\mathbf{d}$ to be a vector pointing down, we can simulate the effect of increased weight at the point $\mathbf{p}$. In our supplemental video we use this to simulate a small robot rolling along the surface of different objects. When the robot 'lands' on a point $\mathbf{p}$ of the object, we fix the robot to $\mathbf{p}$ by applying the time-varying displacement at $\mathbf{p}$ to the image of the robot at each frame. By moving $\mathbf{p}$ along a trajectory specified in the object rest state, we cause the robot to 'roll' along the object's surface in a way that couples their dynamics.

In another example, ForceTree, we control the force $\mathbf{d}$ applied to branches of a tree so that the branches appear to be controlled by a moving hand elsewhere in the video. In this way, we make it appear as though the leaves of the tree are coupled (or controlled through some supernatural force) by the hand. This is substantially simpler that modeling a synthetic tree and matching its appearence to the filmed scene.

6 Conclusion

We have shown that, with minimal user input, we can extract a modal basis for image-space deformations of an object from video and use this basis to synthesize animations with physically plausible dynamics. We believe that the techniques in this paper can be a valuable tool for video analysis and synthesis. The interactive animations we create bring a sense of physical responsiveness to regular videos. Our work could also lead to low-cost methods for special effects by enabling the direct manipulation of objects in video.

Acknowledgements

This work was partially supported by the National Science Foundation under IIS-1420122 and by the Qatar Computing Research Institute. Justin was supported by Shell Research through the MIT Energy Initiative.

References

BATHE, K.-J. 2006. *Finite element procedures*. Klaus-Jurgen Bathe.

BRINCKER, R., VENTURA, C., AND ANDERSEN, P. 2003. Why output-only modal testing is a desirable tool for a wide range of practical applications. In *Proc. Of the International Modal Analysis Conference (IMAC) XXI, paper*, vol. 265.

CHEN, J. G., WADHWA, N., CHA, Y.-J., DURAND, F., FREEMAN, W. T., AND BUYUKOZTURK, O. 2015. Modal identification of simple structures with high-speed video using motion magnification. *Journal of Sound and Vibration 345*, 58–71.

CHUANG, Y.-Y., GOLDMAN, D. B., ZHENG, K. C., CURLESS, B., SALESIN, D. H., AND SZELISKI, R. 2005. Animating pictures with stochastic motion textures. *ACM Trans. Graph. 24*, 3 (July), 853–860.

DAVIS, A., RUBINSTEIN, M., WADHWA, N., MYSORE, G., DURAND, F., AND FREEMAN, W. T. 2014. The visual microphone: Passive recovery of sound from video. *ACM Transactions on Graphics (Proc. SIGGRAPH) 33*, 4, 79:1–79:10.

DAVIS, A., BOUMAN, K. L., CHEN, J. G., RUBINSTEIN, M., DURAND, F., AND FREEMAN, W. T. 2015. Visual vibrometry: Estimating material properties from small motion in video. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June).

DE ROECK, G., PEETERS, B., AND REN, W.-X. 2000. Benchmark study on system identification through ambient vibration measurements. In *Proceedings of IMAC-XVIII, the 18th International Modal Analysis Conference, San Antonio, Texas*, 1106–1112.

DORETTO, G., CHIUSO, A., WU, Y., AND SOATTO, S. 2003. Dynamic textures. *International Journal of Computer Vision 51*, 2, 91–109.

HELFRICK, M. N., NIEZRECKI, C., AVITABILE, P., AND SCHMIDT, T. 2011. 3d digital image correlation methods for full-field vibration measurement. *Mechanical Systems and Signal Processing 25*, 3, 917–927.

HUANG, J., TONG, Y., ZHOU, K., BAO, H., AND DESBRUN, M. 2011. Interactive shape interpolation through controllable dynamic deformation. *Visualization and Computer Graphics, IEEE Transactions on 17*, 7, 983–992.

JAMES, D. L., AND PAI, D. K. 2002. Dyrt: dynamic response textures for real time deformation simulation with graphics hardware. *ACM Transactions on Graphics (TOG) 21*, 3, 582–585.

JAMES, D. L., AND PAI, D. K. 2003. Multiresolution green's function methods for interactive simulation of large-scale elastostatic objects. *ACM Transactions on Graphics (TOG) 22*, 1, 47–82.

LI, S., HUANG, J., DE GOES, F., JIN, X., BAO, H., AND DESBRUN, M. 2014. Space-time editing of elastic motion through material optimization and reduction. *ACM Transactions on Graphics 33*, 4, Art–No.

Table 1: *This table gives a summary of several experimental results. The first row contains the names of examples. The middle row contains an image from the input video representing the rest state of each object, and the bottom row is an example of a synthesized deformation.*

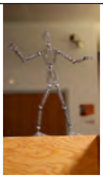
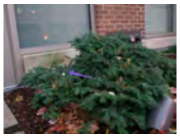
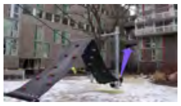
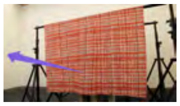
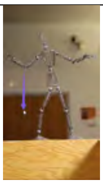
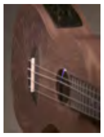
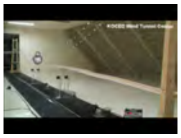
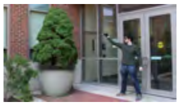
Example Name						
Bush	Playground	Cloth	Wireman	Ukulele	YoutubeBridge	ForceTree
Input Video Image						
						
Synthesized Deformation						
						

Table 2: *This table gives a summary of the parameters of several experimental results. We give the source, length, framerate, and resolution of the source video. The excitation column describes the type of excitation used to excite the object in the input video where: ambient/wind means natural outdoor excitations mostly due to wind, impulse means that the object or its support was manually tapped, and sound means that a ramp of frequencies was played from 20 Hz to the Nyquist rate of the recorded video. We give the number of mode shapes identifed from the input video local motion spectra that are used to simulate the object response and in the final column, the frequency range of these mode shapes.*

Example	**Source**	**Source Length (s)**	**Framerate (fps)**	**Resolution**	**Excitation**	**Number of Modes**	**Frequency Range**
Bush	SLR	80.18	60	640 × 480	Ambient/Wind	77†	1.3 - 4.2 Hz
Playground	SLR	53.85	60	1280 × 720	Impulse	34†	0.8 - 22 Hz
Cloth	SLR	59.77	30	1920 × 1080	Ambient/Wind	147†	0.3 - 0.8 Hz
Wireman	SLR	5.82	60	720 × 1280	Impulse	6	5 - 20 Hz
Ukulele	High-speed camera	8.87	1400	432 × 576	Sound	13	219 - 670 Hz
YoutubeBridge	Youtube (link)	50	30	640 × 480	Wind	18	0.25 - 11 Hz
ForceTree	SLR	35	60	1280 × 720	Impulse	13	0.6 - 9 Hz

† Range of frequencies selected

PAI, D. K., DOEL, K. V. D., JAMES, D. L., LANG, J., LLOYD, J. E., RICHMOND, J. L., AND YAU, S. H. 2001. Scanning physical interaction behavior of 3d objects. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, ACM, New York, NY, USA, SIGGRAPH '01, 87–96.

PENTLAND, A., AND SCLAROFF, S. 1991. Closed-form solutions for physically based shape modeling and recognition. 715–729.

PENTLAND, A., AND WILLIAMS, J. 1989. *Good vibrations: Modal dynamics for graphics and animation*, vol. 23. ACM.

SCHÖDL, A., SZELISKI, R., SALESIN, D. H., AND ESSA, I. 2000. Video textures. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, SIGGRAPH '00, 489–498.

SHABANA, A. A. 1991. *Theory of vibration*, vol. 2. Springer.

SIMONCELLI, E. P., FREEMAN, W. T., ADELSON, E. H., AND HEEGER, D. J. 1992. Shiftable multi-scale transforms. *IEEE Trans. Info. Theory 2*, 38, 587–607.

STAM, J., 1996. Stochastic dynamics: Simulating the effects of turbulence on flexible structures.

SUN, M., JEPSON, A. D., AND FIUME, E. 2003. Video input driven animation (vida). In *Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2*, IEEE Computer Society, Washington, DC, USA, ICCV '03, 96–.

SZUMMER, M., AND PICARD, R. W. 1996. Temporal texture modeling. In *IEEE Intl. Conf. Image Processing*, vol. 3, 823–826.

TAO, H., AND HUANG, T. S. 1998. Connected vibrations: A modal analysis approach for non-rigid motion tracking. In *CVPR*, IEEE Computer Society, 735–740.

WADHWA, N., RUBINSTEIN, M., DURAND, F., AND FREEMAN, W. T. 2013. Phase-based video motion processing. *ACM Trans. Graph. (Proceedings SIGGRAPH 2013) 32*, 4.

WADHWA, N., RUBINSTEIN, M., DURAND, F., AND FREEMAN, W. T. 2014. Riesz pyramid for fast phase-based video magnification. In *Computational Photography (ICCP), 2014 IEEE International Conference on*, IEEE.