Data-Driven Simulation Methods in Computer Graphics: Cloth, Tissue and Faces

http://www.gmrv.es/EG13Course

Miguel A. Otaduy
URJC Madrid

Bernd Bickel
Disney Research Zurich

Derek Bradley
Disney Research Zurich

Contents

1 Introduction **3**
- 1.1 Course Structure . . . 3
- 1.2 Course Schedule (Tentative) . . . 5
- 1.3 Course Notes . . . 5
- 1.4 Instructors . . . 5

2 Overview of Data-Driven Simulation Methods **7**
- 2.1 Example 1: Data-Driven Facial Wrinkles . . . 7
- 2.2 Example 2: Data-Driven Soft Tissue . . . 9
- 2.3 Classification of Methods . . . 10

3 Capturing Geometry and Forces in Real Deformation Examples **12**
- 3.1 Cameras and Lights . . . 12
- 3.2 Geometry Reconstruction . . . 13
- 3.3 Temporal Tracking . . . 17
- 3.4 Actuation and Forces . . . 21
- 3.5 Deformation Examples . . . 23

4 Modeling Nonlinear Soft Tissue from Captured Mechanical Data **26**

5 Data-Driven Modeling of Nonlinear Elasticity in Cloth **42**

6 Animation of Faces with Data-Driven Wrinkles **63**

7 Clothing Animation with Wrinkle Synthesis from Examples **79**

8 Outlook **91**

1 Introduction

In recent years, the field of computer animation has witnessed the invention of multiple simulation methods that exploit pre-recorded data to improve the performance and/or realism of dynamic deformations. Various methods have been presented concurrently, and they present differences, but also similarities, that have not yet been analyzed or discussed. This course focuses on the application of data-driven methods to three areas of computer animation, namely dynamic deformation of faces, soft volumetric tissue, and cloth. The course describes the particular challenges tackled in a data-driven manner, classifies the various methods, and also shares insights for the application to other settings.

The explosion of data-driven animation methods and the success of their results make this course extremely timely. Up till now, the proposed methods have remained familiar only at the research context, and have not made their way through computer graphics industry. This course aims to fit two main purposes. First, present a common theory and understanding of data-driven methods for dynamic deformations that may inspire the development of novel solutions, and second, bridge the gap with industry, by making data-driven approaches accessible. The course targets an audience consisting of both researchers and programmers in computer animation.

1.1 Course Structure

Current data-driven methods for dynamic deformation exploit pre-recorded data in one of two ways. Some methods build on traditional mechanical models to simulate deformations of soft tissue [Pai et al. 2001; Lang et al. 2002; Schoner et al. 2004; Schnur and Zabaras 1992; Becker and Teschner 2007; Kauer et al. 2002; Kajberg and Lindkvist 2004; Bickel et al. 2009; Bickel et al. 2010] or cloth [Breen et al. 1994; Eberhardt et al. 1996; Volino et al. 2009; Bhat et al. 2003; Kunitomo et al. 2010; Wang et al. 2011; Miguel et al. 2012], but parameterize those models in a versatile fashion by interpolation of parameter values estimated from real deformation examples. Other methods, on the other hand, interpolate geometric information on cloth [Wang et al. 2010; de Aguiar et al. 2010; Feng et al. 2010; Kavan et al. 2011; Zurdo et al. 2013] or faces [Bickel et al. 2008; Ma et al. 2008b], and define the information, the interpolation domains, and the interpolation functions, from pre-recorded data. To properly describe each method and facilitate the discussion of differences and similarities, the course starts with an overview and classification of the main approaches.

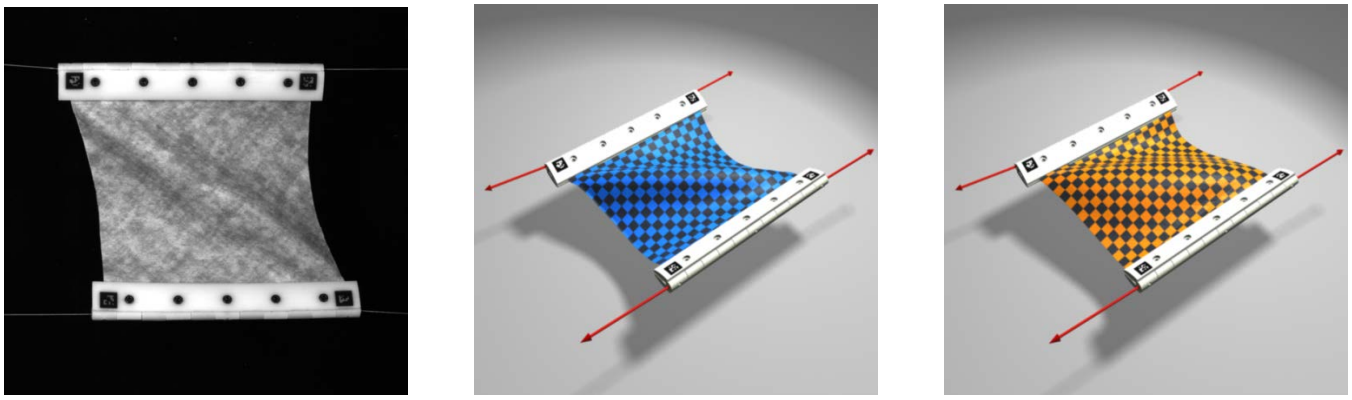


Figure 1: *Example of data-driven cloth simulation with the method of [Miguel et al. 2012]. From left to right, image of the real cloth, reconstructed geometry, and simulation result.*

Then, the course dwells on the description of methods that rely on mechanical data. The course covers two different applications of mechanical-data-driven methods. Fig. 1 shows an example of cloth simulation where cloth deformation models have been estimated from a combination of force-and-deformation information in multiple deformation examples. A similar strategy is followed in the example in Fig. 2, but this time to estimate solid deformation models for soft tissue simulation. The material for the course is combined and adapted from recent publications in

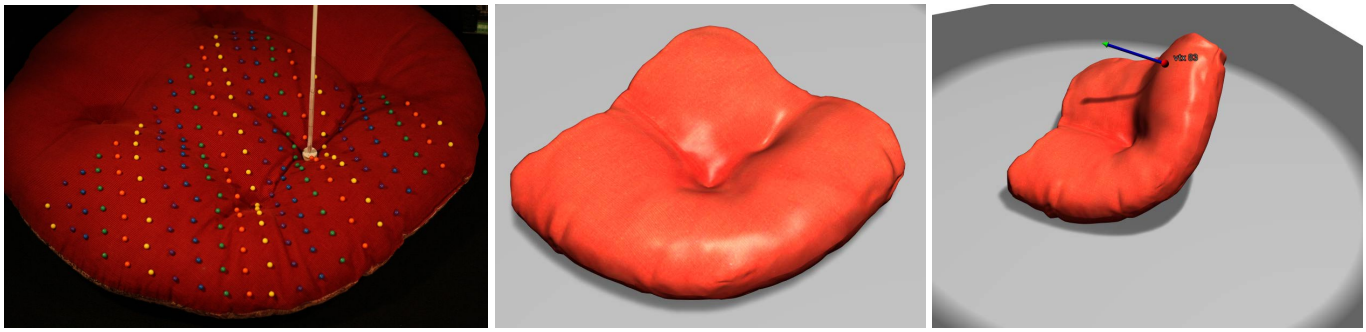


Figure 2: *From left to right: force-and-deformation capture of a non-linear heterogeneous pillow; deformation synthesized with fitted material parameters and the method of [Bickel et al. 2009]; and interactive deformation under different boundary conditions.*

data-driven cloth [Wang et al. 2011; Miguel et al. 2012] and soft-tissue modeling [Bickel et al. 2009; Bickel et al. 2010].

The course also describes animation methods that rely on geometric data. To demonstrate the possibilities of such methods, we present techniques that target two distant applications: animation of highly detailed human faces and cloth animation. Fig. 3 shows facial animation examples where expression wrinkles are synthesized in a data-driven manner. Fig. 4, on the other hand, shows cloth animation examples where folds and wrinkles are synthesized in a data-driven manner. The material for the course is adapted mostly from recent publications in these fields [Bickel et al. 2008; Wang et al. 2010], but we also draw connections with other related methods [Ma et al. 2008b; Kavan et al. 2011], and we discuss the general challenges in defining interpolation functions and domains.

Figure 3: *Facial animation example with the method of [Bickel et al. 2008]. From left to right: large-scale deformation example interpolating mocap markers, full result after example-based fine-scale correction, the same result with full shading, and comparison to the real actor's face.*

One essential component of data-driven simulation methods is data capture, and due to this importance we dedicate a chapter of the course to this problem. It shares challenges with performance capture, but it suffers additional challenges too. Unlike traditional performance capture, which aims at obtaining a reconstruction of arbitrary motion, data capture for data-driven modeling must be designed with the purpose of obtaining a sufficient representation of an object's range of deformations. Therefore, one must design deformation examples that visit the desired range of deformations and suit optimization processes. Moreover, for mechanical-data-driven methods, the capture process must obtain force information in addition to deformation. The material for the course combines and adapts content

Figure 4: *The method of [Wang et al. 2010] uses a precomputed dataset to synthesize cloth wrinkles (a) that are layered onto a coarse base simulation (inset). The precomputed dataset can be used to synthesize wrinkles for a wide range of poses (b and c).*

from several recent publications [Bradley et al. 2008a; Bradley et al. 2008b; Bradley et al. 2010; Bickel et al. 2009; Wang et al. 2010; Wang et al. 2011; Miguel et al. 2012].

1.2 Course Schedule (Tentative)

9:00 am Introduction and overview of methods [Otaduy]

9:20 am Tissue and cloth mechanics [Bickel/Otaduy]

9:50 am Capturing deformation examples [Bradley]

10:30 am Break

10:45 am Facial animation [Bickel]

11:25 am Cloth animation [Otaduy]

12:05 pm Conclusion / Q & A [all]

12:15 pm Close

1.3 Course Notes

The course notes begin with an overview and classification of methods in Chapter 2. Chapter 3 covers solutions for capturing both deformations and forces in an ample set of applications. Then, Chapters 4, 5, 6, and 7 cover, respectively, methods for mechanical-data-driven simulation of soft tissue, mechanical-data-driven simulation of cloth, geometric-data-driven simulation of faces, and geometric-data-driven simulation of cloth. The course notes will be progressively refined, and updated versions will be available on the course web page http://www.gmrv.es/EG13Course. The web page also provides links to publications with supplementary material.

1.4 Instructors

Miguel A. Otaduy is an associate professor in the Department of Computer Science at Universidad Rey Juan Carlos (URJC Madrid). His main research areas are physically based computer animation and haptic rendering. He obtained his BS (2000) on electrical engineering from Mondragón University, and MS (2003) and PhD (2004) on computer science from the University of North Carolina at Chapel Hill. From 2005 to 2008 he was a research associate at ETH Zurich, and then he joined URJC Madrid. He has published over 50 papers in computer graphics and haptics, and has recently co-chaired the program committees for the ACM SIGGRAPH / Eurographics Symposium on Computer

Animation (2010) and the Spanish Computer Graphics Conference (2010). He also leads the ERC Starting Grant *Animetrics*, on measurement-based modeling of complex mechanical phenomena.

Bernd Bickel is a part-time visiting professor at TU Berlin and a post-doctoral researcher at Disney Research Zurich. His research interests include computer graphics and its applications in animation, biomechanics, material science, and computational design for digital fabrication. Recent work includes next generation 3D surface scanner devices, performance capture, measuring and modeling the deformation behavior of soft tissue, and animation tools. Bernd received a M.Sc. in Computer Science in 2006 and spent nine month at Mitsubishi Electric Research Laboratories under the supervision of Prof. Hanspeter Pfister. He wrote his PhD thesis at ETH Zurich in the Computer Graphics Lab headed by Prof. Markus Gross and defended in November 2010.

Derek Bradley is a postdoctoral researcher at Disney Research Zurich. He completed his Bachelor of Computer Science in 2003 and Master of Computer Science in 2005, both at Carleton University in Canada. In 2010, Derek obtained a PhD from the University of British Columbia in Canada, and then started with Disney Research Zurich in September 2010. Derek's main research interest is real-world modeling and animation, primarily through computer vision techniques. He works on various 3D reconstruction projects including multiview stereo, facial performance capture, and data-driven simulation.

2 Overview of Data-Driven Simulation Methods

In the interaction with our surrounding world, mechanical properties play a major role in how we perceive this world. Motion, deformation, flow, fracture or contact, are all mechanical phenomena that allow us to discriminate materials and objects, and to interact with them. Humans have long strived to understand such mechanical phenomena, creating simulation models with which we can replicate or predict the outcome of mechanical processes and events.

It is important to acknowledge that the physical models of the major macroscopic mechanical phenomena are already quite well understood. These models have typically been developed in other disciplines such as physics, mathematics or various engineering fields, and they have made their way through computer animation accompanied by algorithms that are geared to obtaining the desired perceptual stimuli, sometimes incurring in a trade-off between physical realism and interactivity.

Even though the underlying physical models of mechanical phenomena are quite well understood, these phenomena display other inherent sources of complexity that largely limit the applicability of computer animation. Complexity is produced, for example, by nonlinear or anisotropic behaviors, by heterogeneous properties, or by a high dynamic range. These sources of complexity are typically addressed by designing complex nonlinear constitutive models to describe the mechanical behavior. However, these models are implemented using computationally expensive simulation algorithms, which largely limit their applicability. Moreover, their parameters are difficult and tedious to tune, particularly if the properties are heterogeneous. All in all, the animation of complex mechanical phenomena is limited by the domain of effects captured by the underlying physical models and their parameterization accuracy.

Data-driven methods offer an alternative to complex constitutive models, as they turn the modeling metaphor into the knowledge of a system's response under several example conditions. This chapter describes the data-driven modeling metaphor in the context of computer animation, formulates the mathematics of data-driven modeling using two example applications, and introduces a classification of the various existing methods.

2.1 Example 1: Data-Driven Facial Wrinkles

Let us consider a face mesh, consisting of vertices with positions $\mathbf{x} \in \mathbb{R}^3$. Vertex positions can be decomposed into a low-resolution position $\mathbf{x}_0$ and a fine-scale displacement $\Delta\mathbf{x}$, expressed in a local reference system for each vertex (i.e., with orientation $\mathbf{R}$):

$$\mathbf{x} = \mathbf{x}_0 + \mathbf{R}\,\Delta\mathbf{x}. \tag{1}$$

This definition of vertex positions essentially decomposes large-scale facial deformation (i.e., the overall expression of the face) from the small-scale deformation (i.e., expressive wrinkles).

If no assumptions are made, the position of each facial vertex can be defined independently as a function of the facial muscle activations and facial bone configurations. Let us group the muscle and bone configurations in a large vector $\mathbf{u}$. Moreover, due to dynamics, vertex positions are a function of time too. We can write this dependency as $\mathbf{x} = f(\mathbf{u}, t)$. However, due to the repetitive nature of facial expressions, face tissue becomes weaker at certain locations, and expressive wrinkles appear in a deterministic fashion. Moreover, due the viscoelastic nature of facial tissue, the motion of expressive wrinkles appears damped to the human eye. Under these conditions, we can draw the conclusion that fine-scale wrinkle displacements can be defined as a function of some *low-dimensional state* $\mathbf{u}^*$. We can write this dependency as $\Delta\mathbf{x} = f(\mathbf{u}^*)$.

At this point, we have ingredients to define a data-driven model. The potentially high dimensional function of vertex positions has been decomposed into a low-resolution position (inherently low-dimensional), plus a displacement function that can be defined in some low-dimensional state. The remaining open questions are:

- What low-dimensional state $\mathbf{u}^*$ describes best the fine-scale displacement?

- Is the data-driven definition of each vertex completely independent, or can we find relationships across vertices that allow the definition of some global low-dimensional state?

It turns out that, for a vertex, the local strain of the low-resolution face representation serves as a good low-dimensional state. In other words, the existence of expressive wrinkles is closely related to the local strain of the surface. There are multiple choices of strain metrics, and the only major condition for the selection of a strain metric $\mathbf{u}^*$ is that is should be invariant to rigid body motion. Fig. 5 shows the correlation between wrinkles and strain defined through edge deformations.

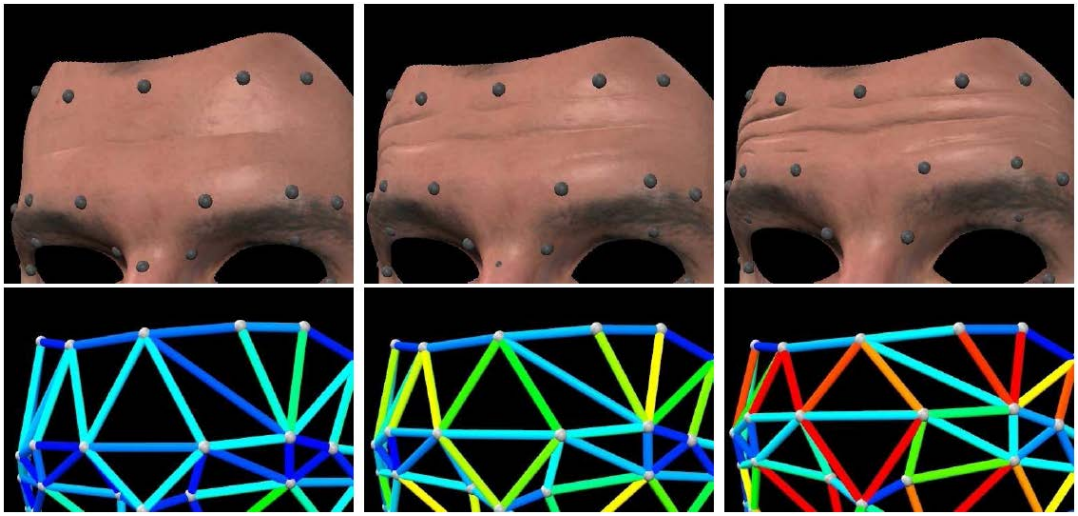


Figure 5: *Correlation between expressive wrinkles and local low-resolution strain (measured through edge deformations). The existence of this correlation enables the definition of a natural low-dimensional domain for interpolating wrinkle displacement data.*

Once a low-dimensional state is selected, the problem is ready for data collection. In our example, the collected data consists of vertex displacement values $\{\Delta\mathbf{x}_i\}$ and local strain values $\{\mathbf{u}_i^*\}$ in correspondence. This data enables the approximation of the function f through *learning methods*. One popular example is scattered-data interpolation based on *radial basis functions*. Then, the approximate function $\bar{f}$ can be formally defined as:

$$\Delta\mathbf{x} \sim \bar{f}(\mathbf{u}^*, \{\Delta\mathbf{x}_i\}, \{\mathbf{u}_i^*\}) = \sum_i \omega_i \, \phi(\mathbf{u}^*, \mathbf{u}_i^*), \tag{2}$$

where ϕ represents a radial basis function, and the weights ω_i are estimated as those that fit best the input data $\{\Delta\mathbf{x}_i\}$.

The approach described so far is successful at describing vertex positions in a data-driven manner, but defines the position of each vertex in a completely independent manner, and may suffer from spatial discontinuities. Ideally, we seek a solution that ensures spatial continuity (and smoothness). The solution is to impose conditions on the captured data and the output of the learning stage, to ensure that vertex displacements are defined based on continuous (and smooth) functions. [Bickel et al. 2008] achieve continuity by building their learning technique as a *weighted pose-space deformation* method. Their approach is described in detail in Chapter 6.

From this example, we can draw several important general conclusions. First, there are certain animation settings that can be modeled efficiently through interpolation of geometric information obtained from representative examples. Second, to find a function that can be described through interpolation, one often successful approach is to decompose the geometric representation in a multi-scale fashion. And third, the definition of an effective interpolation domain can be simplified through the projection of the data to a low-dimensional domain. In the case of expressive wrinkles, local low-resolution strain constitutes a natural low-dimensional domain.

2.2 Example 2: Data-Driven Soft Tissue

Let us consider a deformable solid discretized with tetrahedra. A vector $\mathbf{x}$ concatenates the positions of all nodes in the solid, and a vector $\mathbf{F}$ concatenates the internal forces (due to elastic deformation) acting on the nodes. Under linear elasticity theory, the internal forces are simply proportional to the amount of deformation, measured as the deviation from the rest configuration $\mathbf{x}_0$. The linear relationship between deformation and forces is called the stiffness matrix $\mathbf{K}$, and can be computed using the Finite Element Method. For a homogeneous material, this stiffness matrix depends solely on the structure of the tetrahedral mesh and two material parameters: Young modulus (E) and Poisson ratio (ν). Then, we can formally define the internal forces of the solid as:

$$\mathbf{F} = -\mathbf{K}(E, \nu)\,(\mathbf{x} - \mathbf{x}_0). \tag{3}$$

Unfortunately, real materials are nonlinear, and two parameters hardly describe real elastic behavior. The traditional solution to tackle this problem is to turn to more complex constitutive models, not just linear. However, in a local neighborhood of a given deformation state, a linear model is typically a good descriptor of the material. The complete deformation state of the solid can be described by concatenating the strain tensors of all tetrahedra in a large vector $\mathbf{u}$. Then, the local linear behavior of the material can be defined as a function of the deformation state, $[E, \nu] = f(\mathbf{u})$. It turns out that, by describing separately the material parameters of each tetrahedron, the local linear behavior is well described as a function of the local strain of the tetrahedron itself, which constitutes a considerably lower-dimensional domain. Then, if we define the strain of just one tetrahedron with a vector $\mathbf{u}^*$, the local linear behavior of that particular tetrahedron can be defined as a function $[E, \nu] = f(\mathbf{u}^*)$. Fig. 6 shows two example distributions of Young modulus under different strains.

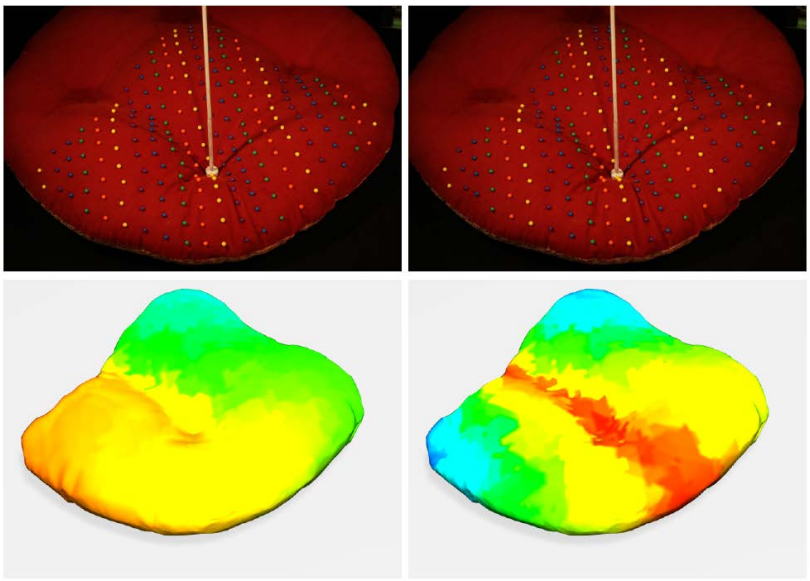


Figure 6: *The top row shows a deformable pillow under two different external forces. The bottom row compares the distribution of values of Young modulus that best describe the behavior of the pillow under these two forces.*

Now, we have reached the ingredients for a data-driven method. Following Example 1, the collected data should consist of material parameter values $\{[E_i, \nu_i]\}$ and local strain values $\{\mathbf{u}_i^*\}$ in correspondence. Unfortunately, material parameter values are difficult to be directly measured on real solid objects. Instead, we opt to collect measurable data, in particular external force values $\{\mathbf{F}_{\texttt{ext},i}\}$ and position values $\{\mathbf{x}_i\}$ in correspondence.

If the data is collected under equilibrium conditions, we can relate nodal positions and applied forces through a quasi-static deformation problem,

$$\mathbf{x} = \mathbf{K}(E, \nu)^{-1}\,(\mathbf{F}_{\texttt{ext}} + \mathbf{F}_{\texttt{other}}) + \mathbf{x}_0. \tag{4}$$

This relationship should hold for all collected pairs of force and deformation data, and this fact will help us estimate the function f that relates tetrahedral strain to material parameter values. To make the problem specific, and similar to Example 1, we can define an approximate function $\bar{f}$ through radial-basis-function interpolation:

$$[E, \nu] \sim \bar{f}(\mathbf{u}^*, \{[E_i, \nu_i]\}, \{\mathbf{u}_i^*\}) = \sum_i \omega_i \, \phi(\mathbf{u}^*, \mathbf{u}_i^*), \tag{5}$$

But recall that, unlike Example 1, in this case the parameter values $\{[E_i, \nu_i]\}$ are unknown, and the strains $\{\mathbf{u}_i^*\}$ cannot be directly measured either. Instead, the strains $\{\mathbf{u}_i^*\}$ will be sampled to sufficiently cover the range of strains in the collected data, and, most importantly, the unknowns of the problem, i.e., the weights of the radial basis functions, ω_i, will be estimated by solving an *optimization* problem. The error function for this optimization problem can be defined as the Euclidean norm between measured positions $\{\mathbf{x}_j\}$ and positions estimated using the data-driven method,

$$\epsilon = \sum_j \left| \mathbf{K} \left(\sum_i \omega_i \, \phi(\mathbf{u}^*, \mathbf{u}_i^*) \right)^{-1} (\mathbf{F}_{\mathtt{ext},j} + \mathbf{F}_{\mathtt{other}}) + \mathbf{x}_0 - \mathbf{x}_j \right|^2 . \tag{6}$$

[Bickel et al. 2009] build on a similar data-driven approach to model nonlinear heterogeneous soft tissue, but they select different material parameters that simplify the optimization problem. Chapter 4 describes their approach in detail.

From this example, we can draw several important general conclusions. First, there are certain animation settings where mechanical parameters can be modeled efficiently through data-driven interpolation. In the case of nonlinear elasticity for soft-tissue deformation, the nonlinear behavior can be modeled through interpolation of local linear models. However, unlike the previous example, parameter data may not be directly measured from examples, which brings us to the second conclusion. By collecting force and deformation data from examples, interpolation weights for the model of mechanical parameters can be estimated through numerical optimization.

2.3 Classification of Methods

To classify data-driven simulation methods in computer animation, we assume that their final output consists of the (deformed) geometry of simulated objects. Then, this geometry is used in the context of rendering algorithms to generate synthetic images of the simulated scene. Drawing from the two examples described above, we can draw a clear classification of data-driven simulation methods into two major categories. One category, represented by Example 1, models in a data-driven manner the geometry itself. The other category, represented by Example 2, models in a data-driven manner some mechanical parameters, and the geometry is obtained as a result of a mechanical model.

Then, we distinguish between *geometric-data-driven methods* and *mechanical-data-driven methods*. In both cases, the data collected in examples includes geometric information (i.e., deformation), but in mechanical-data-driven methods this data should be augmented with force information. Both categories of methods may share techniques for learning, interpolation, or subspace projection. But in mechanical-data-driven methods, the optimization procedures for model fitting require objective functions that account for the mechanical process that relates model parameters to deformation.

Based on our dichotomy of methods, a representative (although not exhaustive) list of data-driven simulation methods in computer graphics (for cloth, tissue and faces) can be classified as follows:

- Geometric-data-driven methods for cloth [Wang et al. 2010; Feng et al. 2010; Kavan et al. 2011; Zurdo et al. 2013].

- Geometric-data-driven methods for faces [Bickel et al. 2008; Ma et al. 2008b].
- Mechanical-data-driven methods for solid tissue [Pai et al. 2001; Lang et al. 2002; Schoner et al. 2004; Schnur and Zabaras 1992; Becker and Teschner 2007; Kauer et al. 2002; Kajberg and Lindkvist 2004; Bickel et al. 2009].
- Mechanical-data-driven methods for cloth [Breen et al. 1994; Eberhardt et al. 1996; Volino et al. 2009; Bhat et al. 2003; Kunitomo et al. 2010; Wang et al. 2011; Miguel et al. 2012].

3 Capturing Geometry and Forces in Real Deformation Examples

In this section we will discuss the process of capturing deformation examples for data-driven simulation. Recently, many different techniques have emerged for capturing the 3D deformation of real surfaces such as cloth [Scholz et al. 2005; White et al. 2007; Bradley et al. 2008b; Furukawa and Ponce 2008] and faces [Furukawa and Ponce 2009; Bradley et al. 2010; Beeler et al. 2011]. These methods primarily use vision-based approaches to acquire both the time-varying shape and corresponding motion of the surface. When capturing deformation examples for simulation we can make use of these general methods, however recovering only shape and motion is typically not enough. In the simulation setting, we must also reconstruct the forces that act on the surface and measure the complete answer that should be predicted by a simulator. This additional challenge often leads to additional capture hardware and specialized reconstruction algorithms. Another point to consider is that the choice of deformation examples can be more critical when considering that the reconstructions will be used in a simulation setting. Often we wish to explore the full range of a material's strain space, possibly exciting different subsets of strain independently. As an example, we may wish to separate the weft strain from the warp strain when deforming a piece of cloth, or actuate different face muscles independently in order to isolate specific facial expressions.

Several recent methods have successfully combined traditional reconstruction algorithms with novel capture techniques for data-driven simulation [Bickel et al. 2009; Wang et al. 2011; Miguel et al. 2012]. These methods form the main focus of our discussion in this course. Here we will give an overview of the related capture setups and reconstruction algorithms, starting with the basics of *Cameras and Lights* (3.1), algorithms for *Geometry Reconstruction* (3.2), computing deformation through *Temporal Tracking* (3.3), obtaining the complete picture of *Actuation and Forces* (3.4), and finally concluding with some hints on which *Deformation Examples* (3.5) might make sense to capture.

3.1 Cameras and Lights

When designing a capture setup, some thought should go into the choice of cameras to use. The first question is whether you need video or still cameras, and this is depends on the examples you wish to capture. Wang et al. [2011] showed that different cloth strains can be isolated in a static way, in which case still cameras such as digital SLRs are sufficient. Still cameras are often used for capturing isolated facial expressions as well [Beeler et al. 2010]. In many cases, however, you will want to capture moving surfaces using video cameras.

The choice of video cameras depends less on the capture application and more on budget. Two options are scientific machine vision cameras or off-the-shelf consumer camcorders. In addition to cost, other factors to consider are camera synchronization, rolling shutter distortions, and system portability. Fig. 7 outlines the tradeoffs between the two. The primary benefit of machine vision cameras is that they can be perfectly synchronized using a hardware trigger. They also provide raw, uncompressed images captured with a global shutter model (i.e. every pixel is exposed at the exact same time). If budget is not an issue then machine vision cameras are the recommended way to go.

On a stricter budget, consumer camcorders are evolving as promising alternatives to scientific cameras in many computer vision applications [Bradley et al. 2008b; Bradley et al. 2010; Atcheson et al. 2008]. They offer high resolution and guaranteed high frame rates at a significantly reduced cost. Also, integrated hard drives or other storage media eliminate the need to transfer video sequences in real-time to a computer, making multi-camera setups more portable. There are two challenges that currently limit the use of such camcorders, especially in multi-camera and camera array applications. First, consumer camcorders typically do not have support for hardware synchronization. Second, in contrast to the global shutter model of scientific cameras, most consumer cameras employ a rolling shutter, in which the individual scanlines use a slightly different temporal offset for the exposure interval (see, e.g. Wilburn et al. [2004]). An illustration of this camera model is shown in Fig. 8. The resulting frames represent a sheared slice of the spatio-temporal video volume that cannot be used directly for many computer vision applications. Bradley et al. [2009] have proposed a solution to the synchronization and rolling shutter problem by

Figure 7: *Trade-offs between machine vision cameras and consumer camcorders.*

capturing under stroboscopic illumination. Strobe lights provide short pulses of illumination, exposing the scene to all cameras at the same time. Even in the rolling shutter model, this approach will *optically* synchronize all scanlines across all cameras. Fig. 9 illustrates this idea and shows experimental results of synchronizing consumer camcorders. Beeler et al. [2010] also use triggered flashes to synchronize multiple digital SLRs for face reconstruction. The tradeoff of these techniques is that more sophisticated lighting hardware is required, and capture must occur in a dimly-lit indoor environment.

A final point on cameras is calibration. Simple white-balancing is often sufficient for radiometric calibration, but more sophisticated color calibration can also be achieved by photographing a color calibration chart. For geometric calibration we must determine the intrinsic parameters, which define how the camera forms an image, and a set of extrinsic parameters, which define the position and orientation of the camera in the world. In most cases, the common calibration technique of Zhang [1999] will suffice. This method is widely used and an implementation is readily available in the OpenCV library [ope]. The basic idea is to capture a number of images of a planar calibration target with known proportions, and then solve for all the camera parameters such that the reprojection error of the target is minimized. Some camera setups, such as a hemispherical camera array, require more sophisticated calibration techniques. We refer to examples such as Beeler et al. [2010] and Bradley and Heidrich [2010].

3.2 Geometry Reconstruction

Reconstructing a deforming surface is often a two-step process. First, the geometry of the surface is acquired, capturing the changing *shape* of the surface over time. Second, the motion of the surface is extracted, recovering the full 3D *deformation*. This section describes methods for recovering shape.

There exists a large body of computer vision literature on reconstructing shape from images. A good survey can be found in Seitz et al. [2006]. One approach is to keep things simple, if your simulation environment allows it. Wang

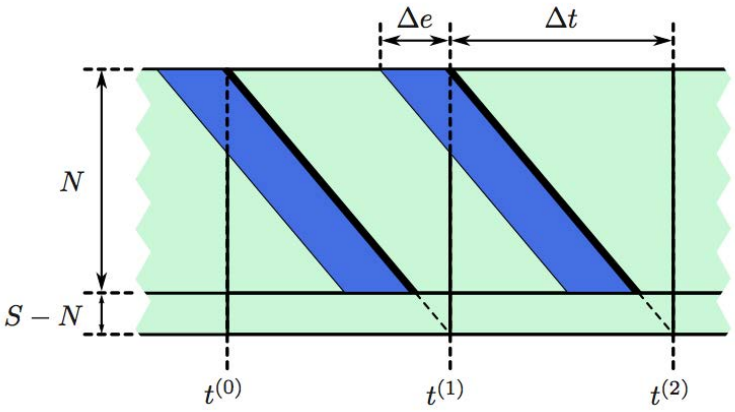


Figure 8: *Rolling shutter camera model, with time as the horizontal axis and scanlines as the vertical axis. The blue region indicates the exposure. Δe is the exposure time, Δt is the frame duration (one over the frame rate), S is the total number of scanlines per frame, and $t^{(j)}$ is the read-out time of the topmost (visible) scanline in frame j. The just-in-time exposure and readout of the individual scanlines creates a shear along the time axis.*

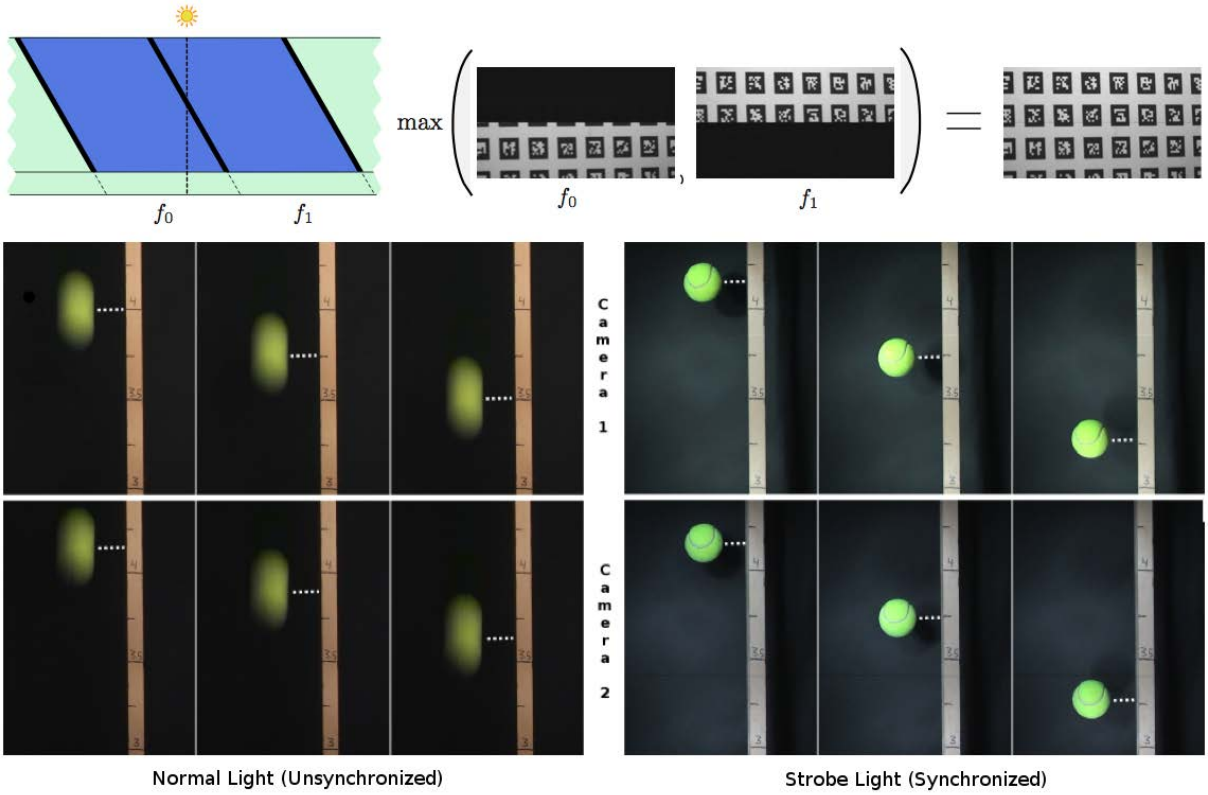
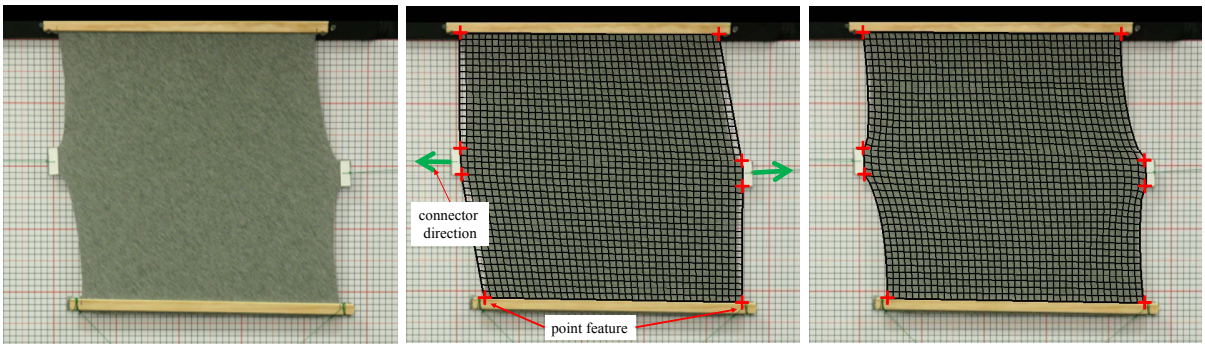


Figure 9: *Stroboscopic Illumination. Top: A flash of light in a dark room exposes all scanlines simultaneously, removing the rolling shutter distortion. The image is split across two consecutive frames, but can be combined in a post-process. Bottom: Strobe lighting synchronizes two consumer camcorders observing a falling ball. As a side effect, motion blur is also removed.*

et al. [2011] show that in-plane cloth deformation can be reconstructed in image-space from a single view and a few labelled feature points (see Fig. 10). More complicated 3D shape recovery typically requires several cameras and multi-view reconstruction algorithms (for example, Fig. 11).



(a) Original photograph (b) Point features (in red) and initialized cloth mesh (a) Final cloth mesh

Figure 10: *Reconstructing cloth deformation from a single view and a few labelled feature points. The resulting cloth mesh is overlaid on the image for comparison.*

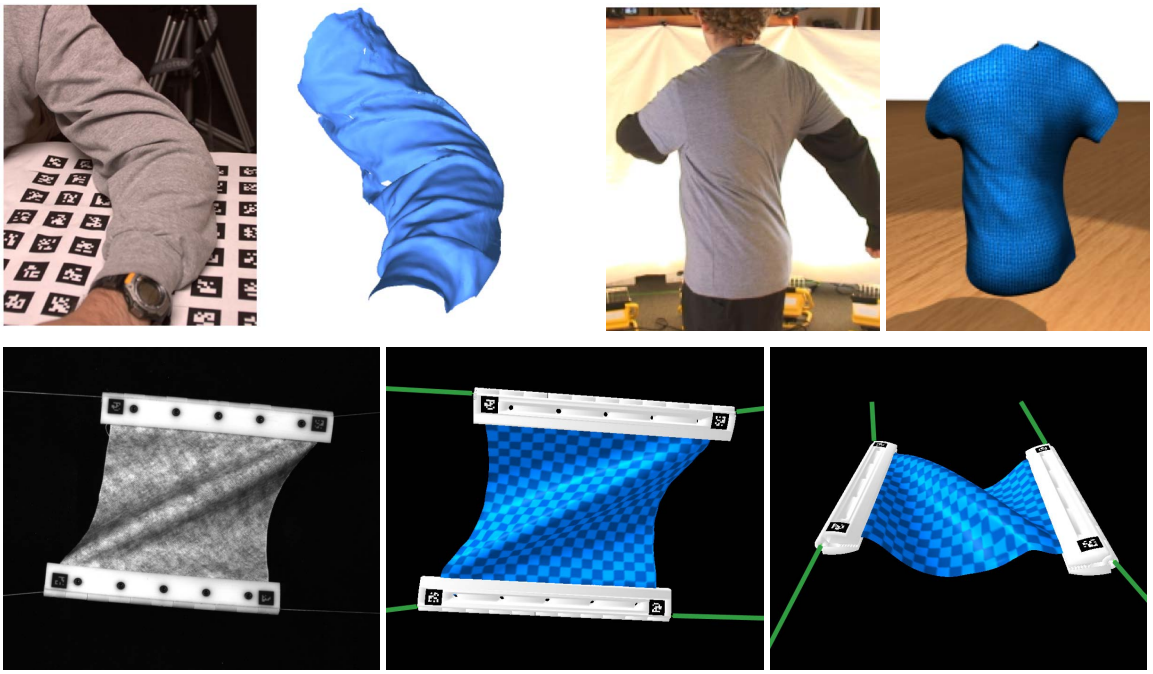


Figure 11: *Multi-view cloth reconstruction based on the technique of Bradley et al. [2008a] (top left), was used for garment capture in Bradley et al. [2008b] (top right), and Miguel et al. [2012] (bottom).*

A good starting point for multi-view reconstruction algorithms is the Patch-based Multi-View Stereo (PMVS) approach of Furukawa and Ponce [2010]. Their method begins by matching features across multiple pictures to obtain a sparse set of corresponding patches, which are then repeatedly expanded to spread the initial matches to nearby pixels until a dense set of correspondences are found. This method performs well on benchmark datasets [Seitz et al. 2006; mview], and the software is available online (http://grail.cs.washington.edu/software/pmvs/). The authors have also extended this approach to be usable for dense motion capture from video streams of garments [Furukawa and Ponce 2008] and faces [Furukawa and Ponce 2009].

An alternative approach is the method of Bradley et al. [2008a], which aims for both accuracy and efficiency. When reconstructing many frames of a deforming surface, efficient runtimes are favorable. This technique has been used for several applications of reconstructing deforming surfaces [Bradley et al. 2008b; Bradley et al. 2010; Miguel et al. 2012], and so it is a good choice for creating deformation examples for simulation. The method is performed in two steps: binocular stereo on image pairs, followed by surface reconstruction. Since software is not available, in the

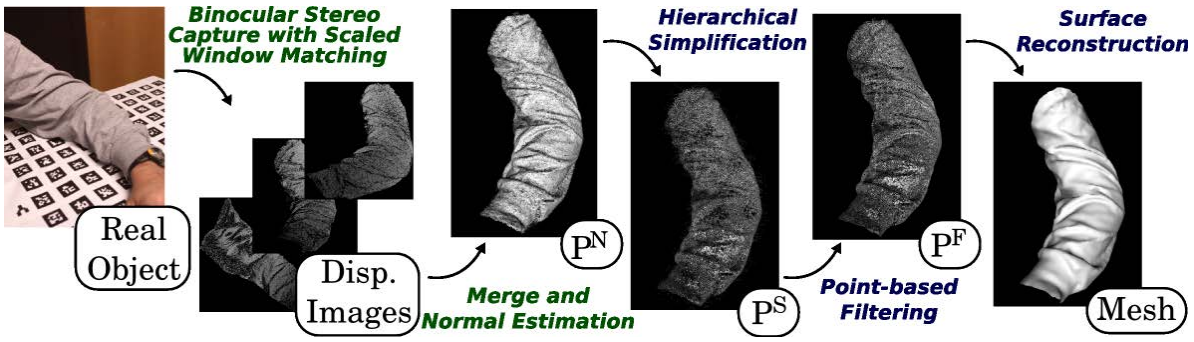


Figure 12: *Overview of the multi-view reconstruction algorithm of Bradley et al. [2008a].*

following we provide more details for the implementation of this technique.

Like most reconstruction algorithms, the input is a set of calibrated images, captured from different viewpoints around the object to be reconstructed. A segmentation of the object from the background should be provided, so that the visual hull is represented as a set of silhouette images. This is easy to achieve if you can capture in front of a green screen or dark background. As we mentioned, the method is performed in two steps, binocular stereo and surface reconstruction. Each step is broken down into individual stages, as illustrated in Fig. 12.

The binocular stereo part of the algorithm creates depth maps from pairs of adjacent viewpoints. First, image pairs are rectified [Fusiello et al. 2000] so that each scanline in one image corresponds to exactly one scanline in the other image. The depth of each pixel in one image is then computed by finding the corresponding pixel along the scanline in the other image and then triangulating. Matching individual pixels can lead to many errors, so a common approach is to match local neighborhoods instead, known as window-matching. Two local neighborhoods of N pixels v_0 and v_1 can be matched using Normalized Cross Correlation (NCC):

$$NCC(v_0, v_1) = \frac{\sum_{j=1}^{N^2}(v_0(j) - \overline{v_0}) \cdot (v_1(j) - \overline{v_1})}{\sqrt{\sum_{j=1}^{N^2}(v_0(j) - \overline{v_0})^2 \cdot \sum_{j=1}^{N^2}(v_1(j) - \overline{v_1})^2}}, \tag{7}$$

where $\overline{v_0}$ and $\overline{v_1}$ represent intensity averages over the neighborhoods. An NCC value of 1 indicates a perfect match, and -1 is the worst possible match. Bradley et al. [2008a] use NCC in a robust window-matching procedure that compensates for perspective distortions by matching under various non-uniform window scales. This feature improves quality in high curvature regions (like the buckling of cloth) and for large camera base-lines (which allows for setups with fewer cameras). The depth images from the binocular stereo pairs are converted to 3D points through triangulation and simply merged into a single dense point cloud. The second part of the algorithm aims at reconstructing a triangular mesh from the point cloud. It consists of three steps:

1. **Downsampling**: The point cloud is usually much denser than required for reproducing the amount of actual detail present in the data. The first step is thus to downsample the data using *hierarchical vertex clustering* [Boubekeur et al. 2006].

2. **Cleaning**: The simplified point cloud remains noisy. While some methods integrate the noise removal in the meshing algorithm [Kazhdan et al. 2006], others feel that this important data modification must be controlled explicitly, prior to any decision concerning the mesh connectivity. In this reconstruction algorithm, the problem is addressed at the point level using *point-based filtering* tools (see [Alexa et al. 2004; Gross and Pfister 2007] for an introduction), producing a filtered point set.

3. **Meshing**: The final step is to generate a triangle mesh without introducing excessive smoothing. Building on

lower dimensional triangulation methods [Boubekeur et al. 2005; Gopi et al. 2000], triangle mesh patches are created in 2D and then ”lifted” to 3D as mini-heightfields. This approach is fast and runs locally, ensuring scalability and good memory-computational complexity.

Here we have briefly described only two out of the multitude of multi-view stereo algorithms that have been published (currently over 50 are evaluated at http://vision.middlebury.edu/mview/eval/). Choosing the right method for your application can be a challenging and time-consuming process. Our hope is to provide enough background and resources to find the best reconstruction algorithm that suits your needs.

3.3 Temporal Tracking

The geometry reconstruction algorithms from the previous section can be used to compute a triangle mesh per-frame of the deforming surface. While it is important to obtain the time-varying shape, the full 3D deformation must also include surface tracking, such that the 3D motion of each surface point is reconstructed. A convenient way to represent the deformation is a triangle mesh with constant connectivity over time and varying vertex positions. In this section we discuss different ways to perform temporal tracking and couple the result with the per-frame geometry to obtain reconstructed surface deformations.

In this course we will focus on optical tracking, where it is assumed we have images of the deforming surface. When images are not available, a complementary form of tracking can be used, which relies entirely on the deforming geometry. For example, non-rigid shape registration can generate dense surface correspondences over time. A good overview of these techniques is given in the recent Eurographics Tutorial by Wand et al. [2012]. In the remainder of this section, we focus our discussion on image-based tracking methods.

Early work in tracking deforming surfaces was to use hand-placed markers which can be identified and tracked with ease [Williams 1990; Guenter et al. 1998]. This idea has led to great success in marker-based facial performance capture [Lin and Ouhyoung 2005; Bickel et al. 2008; Ma et al. 2008a], which currently drives facial animation in the entertainment industry. For cloth, some of the first research in capturing garment motion from video has also employed marker-based techniques [Scholz et al. 2005; White et al. 2007]. These methods use a unique encoding of color marker arrays to locate specific points on a garment over time. Fig. 13 shows a few examples of marker-based reconstruction and motion capture for cloth.

Figure 13: *Marker-based motion capture methods for cloth. Left: Scholz et al. [2005], Right: White et al. [2007].*

More recent research has shown that dense motion capture can be achieved in a markerless setting, if the surface can be painted with a high-frequency texture [Furukawa and Ponce 2009; Miguel et al. 2012] or with high enough image resolution to use fine details as surface texture [Bradley et al. 2010; Beeler et al. 2011]. Furukawa and Ponce track face motion starting with their previous work on dense 3D motion capture [Furukawa and Ponce 2008], which assumes tangentially rigid motion, and then introducing a new tangential regularization method capable of dealing with the stretching, shrinking and shearing of deformable surfaces such as skin [Furukawa and Ponce 2009].

The methods of Bradley et al. [2010], Beeler et al. [2011] and Miguel et al. [2012] all rely on dense 2D optical flow in order to compute the 3D surface motion. Optical flow is an image-space vector field that encodes the motion of the pixels from one frame to the next in a video sequence (see Baker et al. [2011] for a survey of techniques).

Although each reconstruction approach varies slightly in the use of optical flow for 3D tracking, the general ideas are similar. For the purpose of notation, lets call the per-frame geometry reconstructions G^t, where t corresponds to the frame number or time. These meshes can be the raw result of the reconstruction algorithms described in the previous section. Given G^t and the optical flow fields of each input video, we would like to generate a set of compatible meshes M^t that have the same connectivity as well as explicit vertex correspondence. That is to say, we desire one mesh that deforms over time. Without loss of generality, we can choose M^0 to represent the global topology, and then the goal is to track M^0 forward (and possibly backwards) in time to establish the mesh sequence M^t. The basic tracking approach is illustrated on face meshes in Fig. 14, and it proceeds as follows. For each vertex v_i^{t-1} of M^{t-1} we project the vertex onto each camera c in which it is visible (i.e. inside the field of view of and not occluded). Let $p_{i,c}$ be this projected pixel. We then look up the 2D flow vector that corresponds to $p_{i,c}$ and add the flow to get a new pixel location $p'_{i,c}$. Back-projecting from $p'_{i,c}$ onto G^t gives us a guess for the new vertex location, which we call $\bar{v}_{i,c}^t$.

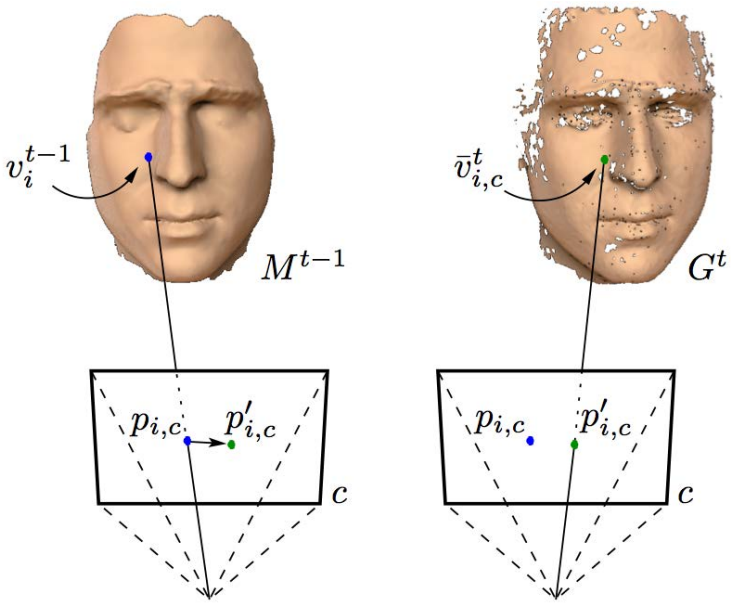


Figure 14: *Basic mesh tracking using per-camera optical flow.*

The figure illustrates the 3D motion estimation for vertex v_i according to one camera, c. The estimates from all cameras can be combined in a weighted average, giving more influence to the cameras that have a better view of the surface point:

$$\bar{v}_i^t = \sum_{c=1}^{n} w_{i,c}^t \cdot \bar{v}_{i,c}^t, \tag{8}$$

where $w_{i,c}^t$ is the dot product between the surface normal at $\bar{v}_{i,c}^t$ and the vector from there to c. Since each vertex is updated independently, a regularization step avoids possible triangle-flips and removes any unwanted artifacts that may have been present in the initial reconstruction. A common regularization approach for meshes stems from *Laplacian surface editing* [Sorkine et al. 2004]. Following de Aguiar et al. [2008], we solve a least-squares Laplacian system using cotangent weights and the current positional constraints $\bar{v}_i^t$. Thus, we generate the final mesh M^t by minimizing

$$\arg\min_{v^t}\{\| v_i^t - \bar{v}_i^t \|^2 + \alpha \| Lv^t - Lv^0 \|^2\}, \tag{9}$$

where L is the cotangent Laplacian matrix. The parameter α controls the amount of regularization.

Repeatedly propagating the mesh through time using optical flow can lead to several unpleasant artifacts (illustrated in Fig. 15). First, optical flow tracks can be lost due to occlusion. Second, it is generally well-known that optical-flow based tracking methods suffer from accumulation of error, known as drift. Lets first consider drift. Although the error from one frame to the next is usually small and imperceptible, the error can accumulate over time, resulting in incorrect motion estimation. Drift typically occurs because optical flow is computed between successive video frames only. If it were possible to accurately compute flow between the first video image and every other frame, there would be no accumulation of error. Unfortunately, most temporally distant video images in a capture sequence are usually too dissimilar to consider this option. Bradley et al. [2010] and Beeler et al. [2011] present two different solutions to this problem.

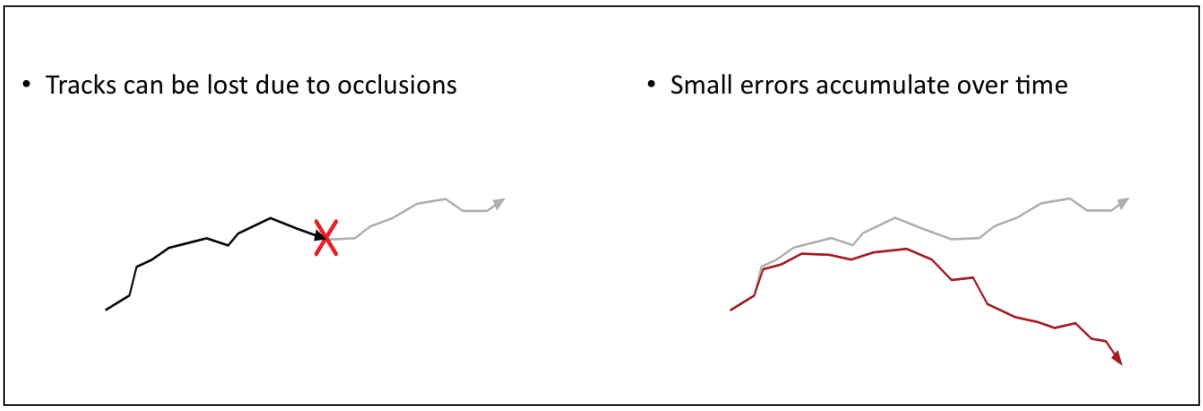


Figure 15: *Two problems that can happen when using optical flow with sequential propagation. 1) Tracks can be lost due to occlusions, and 2) Small errors can accumulate over time and cause drift.*

Bradley et al. [2010] compute a 2D parameterization of the surface (or a UV-map) and then build per-frame texture images from the input videos. Two example texture images are given in Fig. 16. Every vertex of the mesh has unique 2D coordinates in the parameter domain, yielding a one-to-one mapping between 2D and 3D mesh triangles. Their main observation is that the texture domain of the mesh remains constant over time, which means that the computed per-frame texture images are all very similar. Any temporal drift in the 3D geometry appears as a small 2D shift in the texture images, which can easily be detected, again by optical flow. Automatic drift correction is then implemented as follows. After computing the geometry M^t and texture T^t for a given frame, optical flow is computed between the textures T^0 and T^t . This flow (if any is detected) is then used to update M^t on a per-vertex basis using the direct mapping between the geometry and the texture. Any shift in texture space becomes a 3D shift along the mesh surface. After updating the vertices to account for drift, Laplacian regularization is applied to avoid possible triangle flips.

Beeler et al. [2011] take a different approach to eliminating drift in the reconstructed sequence. Leveraging the fact that facial performances often contain repetitive subsequences, their method identifies so-called *anchor frames* as those which contain similar facial expressions to a manually chosen reference expression. Anchor frames are automatically computed over one or even multiple performances. This method introduces a robust image-space tracking method that computes pixel matches directly from the reference frame to all anchor frames, and thereby to the remaining frames in the sequence via both forward and backward sequential matching. This allows the propagation of one reconstructed frame to an entire sequence in parallel, in contrast to the previous sequential methods. This anchored reconstruction approach limits tracking drift, since every anchor frame brings the tracking error back to (nearly) zero. The idea of using anchor frames also helps to overcome additional problems with sequential motion tracking, and that is occlusion and motion blur. Sequential tracking methods would fail during an occlusion or blurred frame, thus losing track of the surface and would not be able to recover. As a result, the motion

Figure 16: *Example face textures from Bradley et al. [2010]. Notice that the parameterization domain is constant even though the 3D face and expression changes. The only change in the textures is due to wrinkles and appearance.*

for the remainder of the performance would not be reconstructed. The anchor frame reconstruction framework can recover from such tracking failure, as long as an anchor frame exists later on in the sequence. The main drawback of the anchor frame approach is that it is designed specifically for deformation sequences that return to a similar pose often throughout the sequence. With arbitrary non-cyclic deformations, you may not experience the benefits of this algorithm.

Resulting facial performance reconstructions using the methods of Bradley et al. [2010] and Beeler et al. [2011] are shown in Fig. 17 and Fig. 18, respectively. Each figure shows a reference image, the reconstructed geometry, and the reconstructed motion illustrated with a grid overlaid on the deforming surface.

Figure 17: *Deforming face reconstructions from Bradley et al. [2010]. The top row is a reference image, middle row shows the resulting geometry, and bottom row shows the reconstructed motion using an overlaid checkerboard pattern.*

While these tracking methods have been described in the context of facial performance capture, similar tracking methods can be applied to any smooth deforming surface. The deformation examples in the data-driven cloth simulation work of Miguel et al. [2012] were reconstructed using the same techniques.

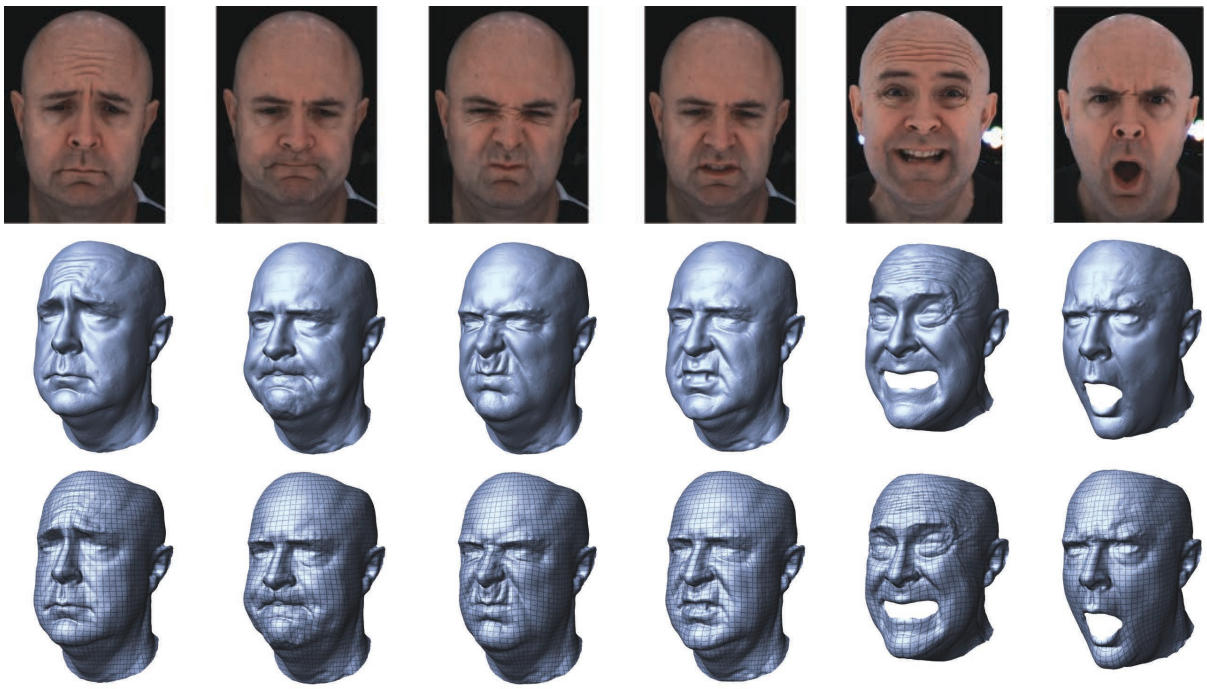


Figure 18: *Deforming face reconstructions from Beeler et al. [2011]. The top row is a reference image, middle row shows the resulting geometry, and bottom row shows the reconstructed motion using an overlaid grid pattern.*

3.4 Actuation and Forces

So far we have discussed different physical capture setups and various methods for 3D reconstruction of shape and motion tracking for deforming surfaces. In order to capture examples for simulation, we often have additional requirements. First, we should have complete control of the surface in order to actuate very specific deformations. Secondly, we must reconstruct the forces that act on the surface and measure the complete answer that should be predicted by a simulator. These additional challenges often lead to additional capture hardware and specialized reconstruction algorithms. In this section we will outline some of the physical setups and algorithmic approaches that have been used to acquire this additional information.

Bickel et al. [2009] acquire a set of example deformations of real objects, such as soft pillows and faces, including force information using a simple capture system. Their acquisition system consists of force probes and a marker-based reconstruction (see Fig. 19). Deformations are induced by physical interaction with the object, meeting the requirement of having complete control of the surface. The second requirement (reconstructing forces), is met by building contact probes with arbitrary shapes and circular disks of different diameters attached to the tip of a long screwdriver. The position and orientation of the contact probe is estimated using two makers on the white shaft of the screwdriver. To measure the magnitude of the contact forces, a force sensing resistor is used.

Wang et al. [2011] devise a setup for measuring in-plane cloth deformations using weights to create specific forces (see Fig. 20). A piece of cloth is mounted vertically, with the top and bottom edges sandwiched between a pair of wooden slats to constrain the motion in a controlled way. The left and right edges of the cloth are attached to clips in the middle of each edge. These locations are treated as boundary conditions and their positions can be easily measured using a calibrated camera. Wires are attached to the clips and then weights are attached to the other end of the wires over pulleys in order to apply tension. The top edge of the cloth sample is attached to the top of the testing board, while the other three edges have freedom to move. Different weights are applied on these three edges in order to drag the cloth sample into different shapes. The left and right sides are loaded with the same weights so that the sample does not lose its balance during the experiment. This is a simple mechanism to control the magnitude of the

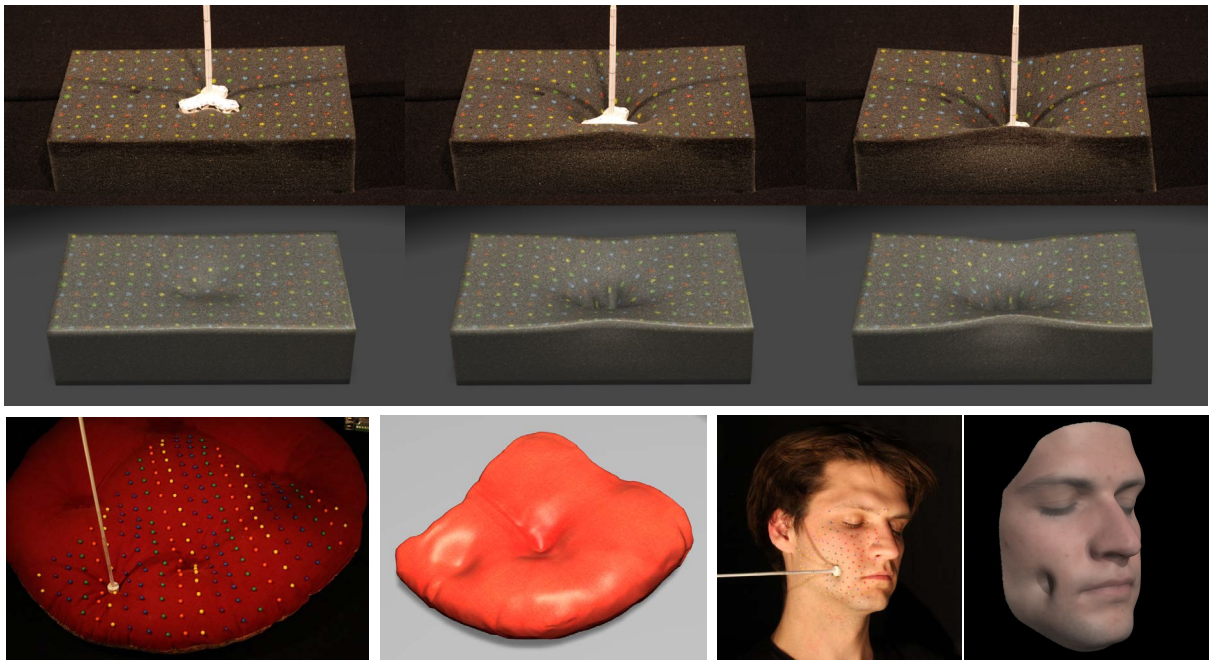


Figure 19: *The capture setup of Bickel et al. [2009] uses force probes and marker-based reconstruction to acquire example deformations of real objects.*

force applied to the cloth, and is attractive since the setup is inexpensive and no specialized hardware is required.

Miguel et al. [2012] designed a more elaborate acquisition system for cloth that explores a substantial range of the materials strain space, and records complete information about the forces applied to the cloth and the deformation that it undergoes. Like Wang et al. (and previous work), this setup focuses primarily on tensile forces. Tests are performed on 100 mm square cloth samples using two kinds of plastic clips: small, rounded clips that grab a localized area, and long clips that grip one whole side of the sample. Forces are applied to the clips by fine wire cords that are pulled to defined displacements by eight linear actuators, and the tension in the cords is monitored by miniature load cells located at the actuator ends (see Fig. 21). Using the reconstruction methods described previously, the geometry and motion of the cloth is captured. The location and orientation of the cords attached to the clips (which reveal the direction of the applied force) are also reconstructed, by fitting 3D lines to reconstructed points along each cord (see Fig. 22). This system, although more complex to construct than that of Wang et al., is able to produce

- The 3D configuration of the cloth sample, represented as a deformed mesh with 10K regularly sampled vertices.
- The 3D positions and orientations of all clips attached to the cloth, including a list of clamped cloth vertices.
- The 3D forces applied to all clips. The magnitudes are determined by the tension measurements, and the directions are determined by the observed directions of the cords.

Note that the actuator positions themselves are not part of the output, since they are superseded by the displacements measured at the clips. This prevents stretching of the cord, or other factors affecting the distance between the clip and the actuator, from affecting displacement accuracy.

These are just a few examples of physical setups that have been used to generate controlled deformation and measure the forces in addition to the deforming geometry.

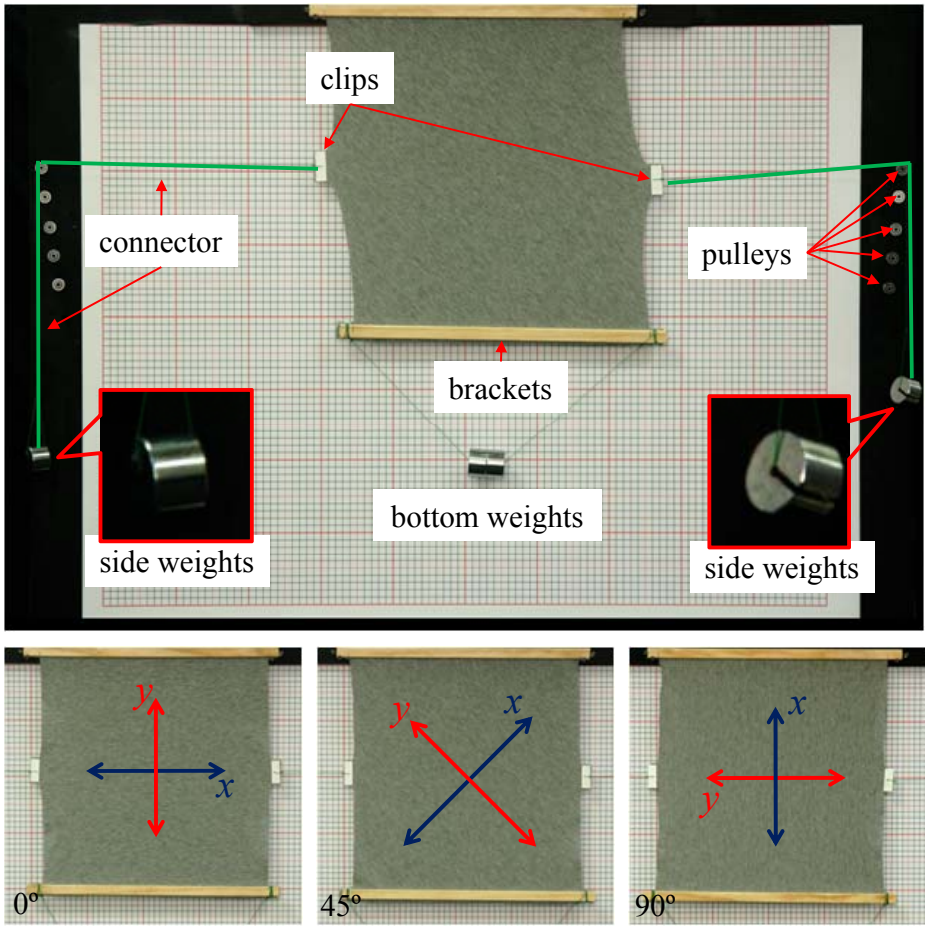


Figure 20: *The capture setup of Wang et al. [2011] uses weights to control cloth deformations by applying specific forces to a cloth sample at various attachment points.*

3.5 Deformation Examples

We conclude this section with some hints on what deformation examples might make sense to capture, and point to data that is already available online.

The types of interesting deformations depends on the object you are capturing, and of course on the application you have in mind. For cloth, it has been a popular idea to isolate specific stretching and bending deformations. Wang et al. [2011] follow the biaxial tensile method in the textile literature, which tests the cloth sample by stretching it simultaneously in both warp and weft directions. Using the setup described previously (recall Fig. 20), a number of stretching tests are performed. For cloth materials with symmetric properties to their warp and weft directions, they create three 400mm x 400mm cloth samples with bias angles 0°, 45° and 90° respectively. The bias angle is defined as the rotational angle from the warp-weft coordinate system to the samples local coordinate system counterclockwise. Warp and weft directions can be recognized from thread directions in the weaving structure for most cloth materials. Each sample is typically tested by seven different weights at the bottom, going from 0g to 600g, and five weights on both sides, from 0g to 400g. In total, there are 35 tests for each sample and 105 tests for each cloth material. This test set covers the range of forces typically experienced by the cloth in clothing when it is worn. Cloth bending is also measured, by clamping a sample in a bent position. A sample is incrementally advanced so that progressively more of it protrudes from the clamp, and the sample drapes into different curves under its own weight. These curves are captured from a side view and the trajectory of each curve is manually labeled using point

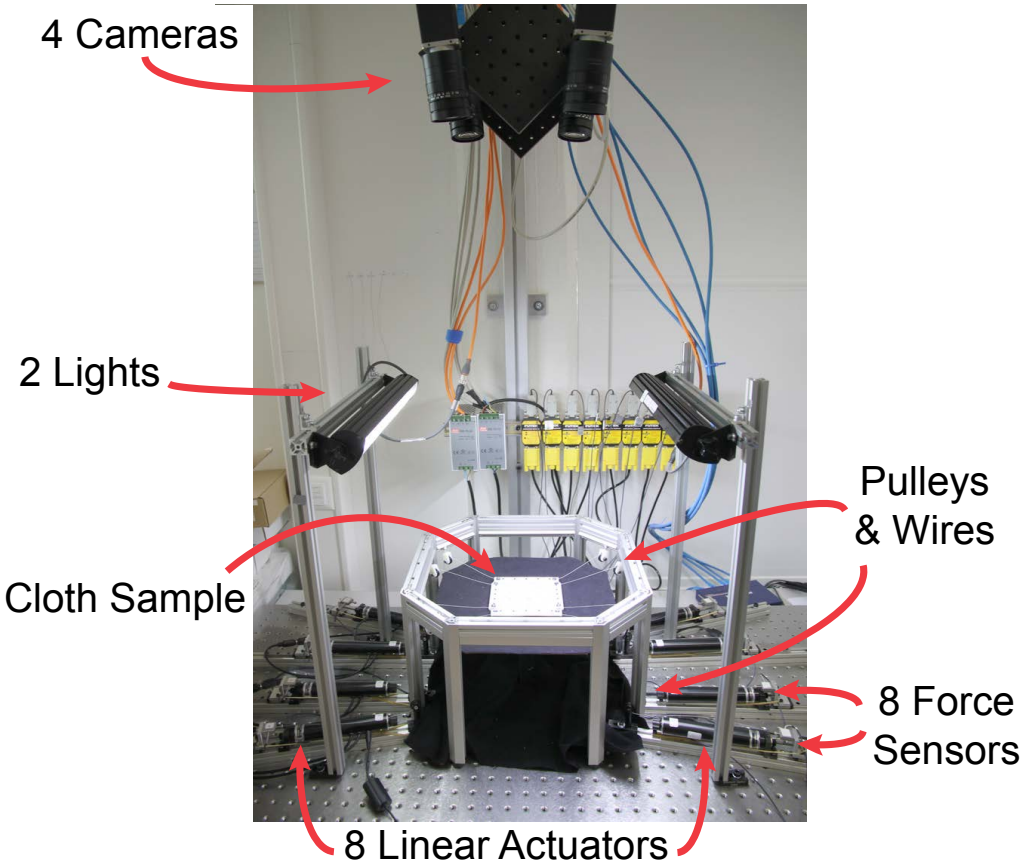


Figure 21: *The acquisition setup of Miguel et al. [2012] is designed to produce a variety of controlled deformations through the use of actuators, wires, pulleys and force sensors.*

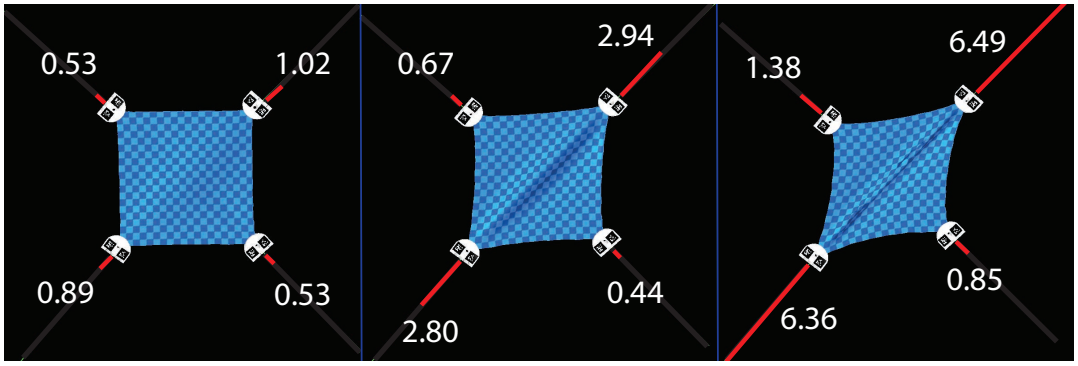


Figure 22: *Force measurements from the setup of Miguel et al. [2012].*

features.

Motivated by the goal of parameter fitting, Miguel et al. [2012] designed deformation sequences of cloth that produce near-isolated strains, and allow estimating stretch, shear and bending properties in a separate and incremental manner. Unlike standard textile evaluation practices, they relax the requirement of uniform strains. Using the setup described previously (recall Fig: 21), stretching is isolated by performing a uni-axial tension experiment, with forces applied to two long bar clips attached to either side of the cloth. The cloth is slowly stretched until a maximum

force is reached and then slowly released back. The process is repeated three times, in both weft and warp directions separately. Shearing is captured using an approximate picture-frame experiment, where four long clips fix the cloth boundaries and shear stress is applied as the cords pull on opposite corners. Similar to Wang et al., to isolate bending deformation they slowly push the flat cloth sample off the edge of a table and measure its shape as it bends under its own weight, for both weft and warp directions. However, here Miguel et al. measure 3D bending rather than a 2D curve. This gives a total of five measurements per cloth sample that are used for parameter fitting (two stretch, one shear, and two bending). Additional, more complex, deformations are also captured for validating their algorithm.

These experiments on cloth all aim to capture the full strain-space of the material. If instead you are working with faces, it is not as easy to isolate the different modes of deformation. However, a lot of work has focused on defining a representative set of facial expressions, from which many new expressions can be formed. A standard set of expressions has been defined in the Facial Action Coding System (FACS) of Ekman [1978].

Finally, since it can be a difficult task to capture real deformation examples for simulation, here is a short list of datasets that are already available online:

- Cloth datasets of Wang et al. [2011] are available at
  http://graphics.berkeley.edu/papers/Wang-DDE-2011-08/
- Example garment capture sequences of Bradley et al. [2008b] are available at
  http://www.cs.ubc.ca/labs/imager/tr/2008/MarkerlessGarmentCapture/data.html
- Facial performance capture data of Bradley et al. [2010] is available at
  http://www.cs.ubc.ca/labs/imager/tr/2010/Bradley_SIG2010_FaceCapture/
- An example facial performance from Beeler et al. [2011] can be requested from
  http://graphics.ethz.ch/publications/papers/paperBee11.php
- Datasets of articulated mesh animations of people from the work of Vlasic et al. [2008] can be obtained from
  http://people.csail.mit.edu/drdaniel/mesh_animation/index.html
- Finally, similar datasets from the work of de Aguiar et al. [de Aguiar et al. 2008] can be requested at
  http://www.mpi-inf.mpg.de/resources/perfcap/

4 Modeling Nonlinear Soft Tissue from Captured Mechanical Data

Motivation

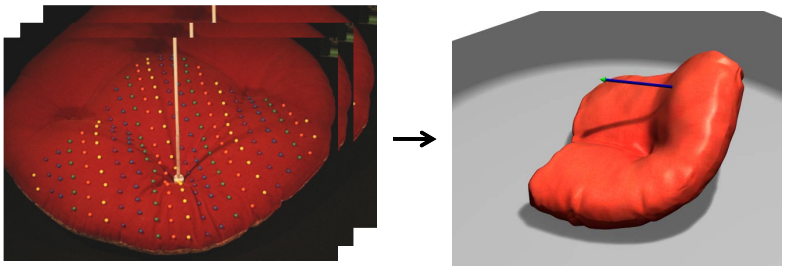
Challenges

Material
Heterogeneity

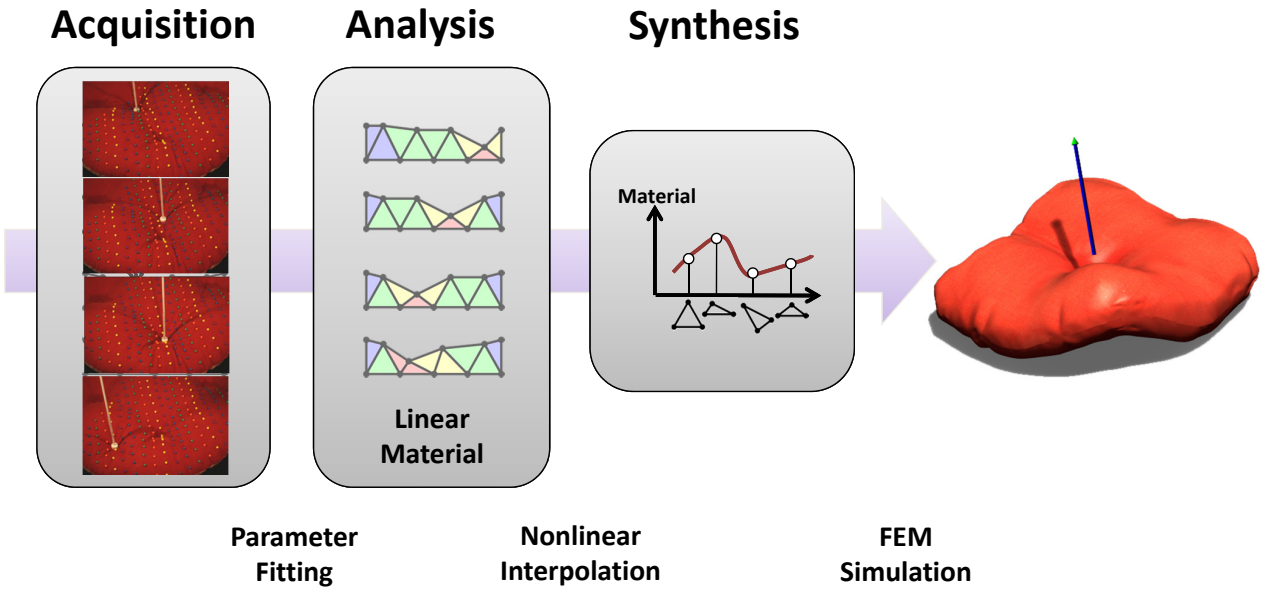
Material
Non-linearity

Goals

- Non-linear heterogeneous soft tissue
  - Simple method
  - Easy and robust fit
  - Reproduce deformation behavior
  - Interactive performance



Our Approach

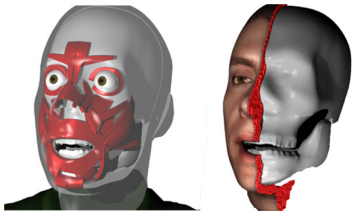


Related Work (I)

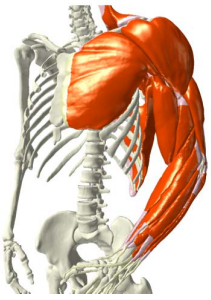
- Bio-mechanical Models

  Neo-Hookean, Mooney-Rivlin, Ogden, Rubin-Bodner, …

  ✓ Heterogeneous
  ✓ Non-linear
  ✗ Model and parameters are difficult to choose



[Sifakis et al. 2005]

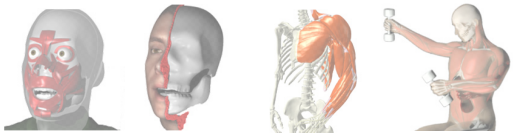


[Teran et al. 05]

[Lee et al. 09]

Related Work (II)

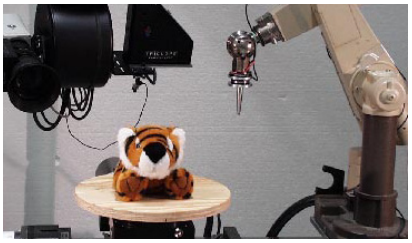
- Bio-mechanical Models



- Measurement-based

  Green's function [Pai et al. 01], [Lang et al. 02], [Lang et al. 03]
  Viscoelasticity [Schoner et al. 04]

  ✓ Heterogeneous
  ✗ Linear

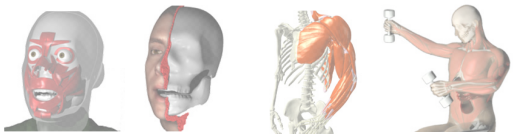


[Pai et al. 01]

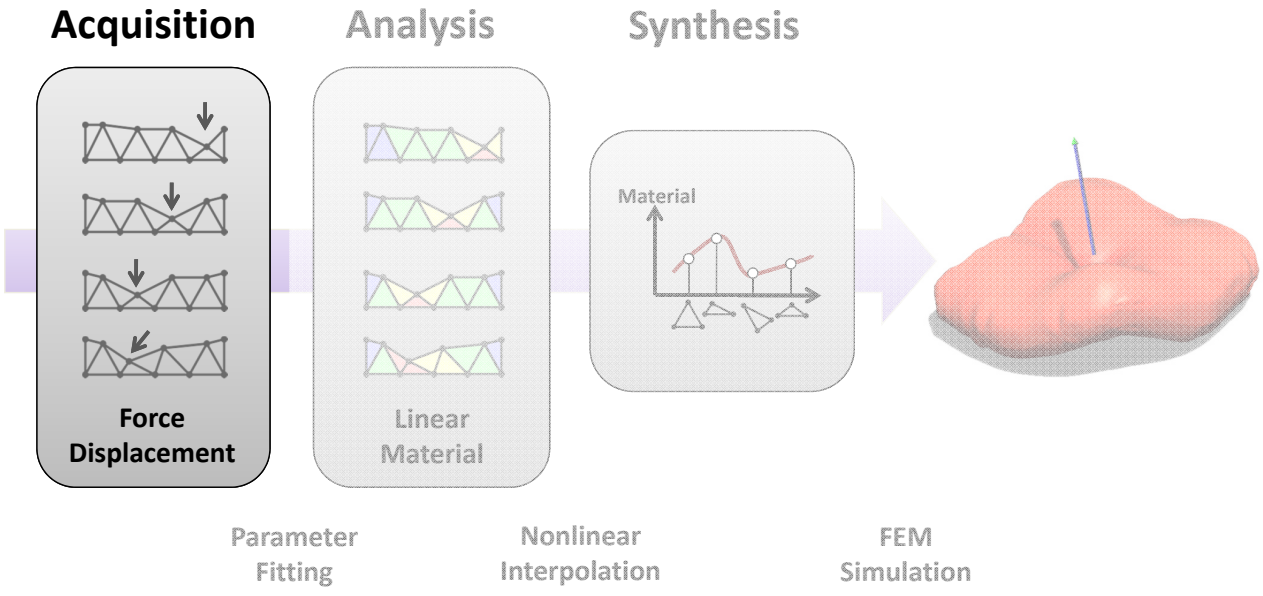
[Schoner et al. 04]

Related Work (III)

- Bio-mechanical Models
- Measurement-based
- Constitutive model fitting
  [Schnur and Zabaras 92],
  [Kauer at al. 02],
  [Becker and Teschner 07]



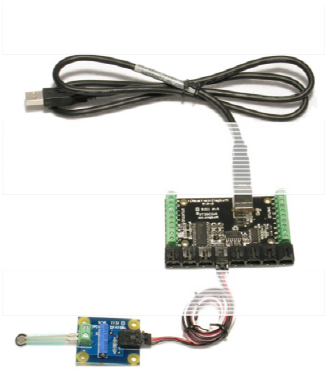
Overview



Data Acquisition

3 Canon 40D cameras
External trigger for sync

Contact probe

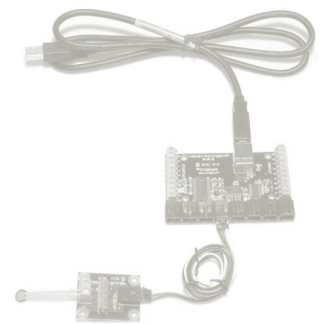


PhigetInterfaceKit with
force sensing resistors

Data Acquisition

3 Canon 40D cameras
External trigger for sync

Contact probe

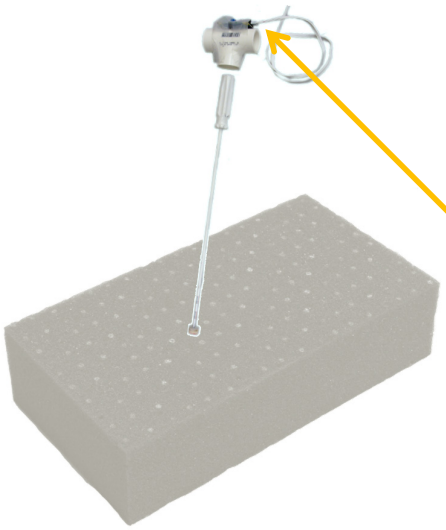


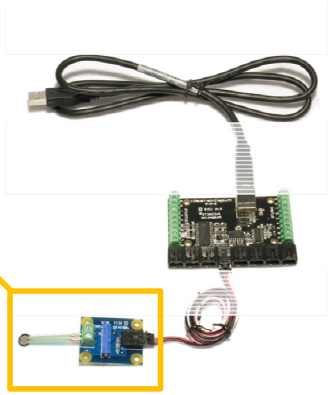
PhigetInterfaceKit with
force sensing resistors

Data Acquisition

3 Canon 40D cameras
External trigger for sync

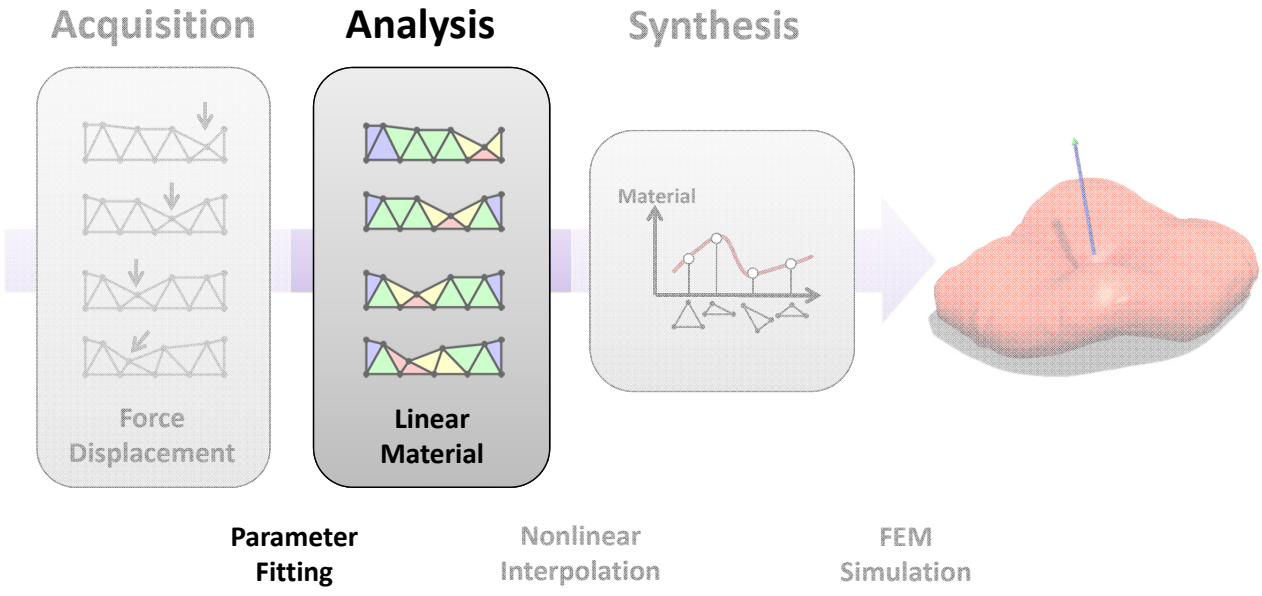
Contact probe





PhigetInterfaceKit with
force sensing resistors

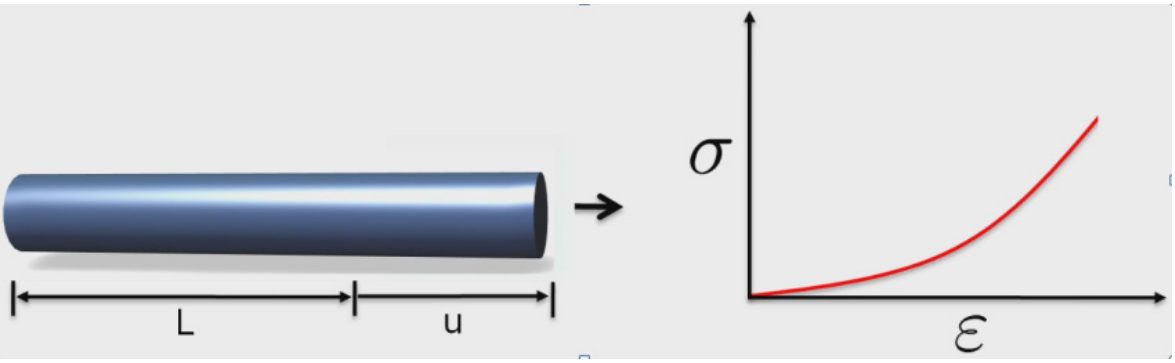
Overview



Material Representation

Stress-Strain relationship 1D

- Strain $\varepsilon = u/L$
- Stress $\sigma = F/A$

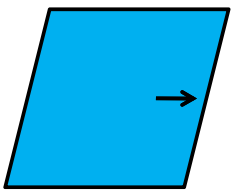


Material Representation

Strain in 3D

Cauchy's linear strain tensor

$$\varepsilon(\mathbf{u}) = \tfrac{1}{2}\left(\nabla\mathbf{u} + (\nabla\mathbf{u})^T\right) = \begin{bmatrix} \varepsilon_{xx} & \varepsilon_{xy} & \varepsilon_{xz} \\ \varepsilon_{yx} & \varepsilon_{yy} & \varepsilon_{yz} \\ \varepsilon_{zx} & \varepsilon_{zy} & \varepsilon_{zz} \end{bmatrix}$$



Material Representation

Strain in 3D

Cauchy's linear strain tensor

$$\varepsilon(\mathbf{u}) = \tfrac{1}{2}\left(\nabla\mathbf{u} + (\nabla\mathbf{u})^T\right) = \begin{bmatrix} \varepsilon_{xx} & \varepsilon_{xy} & \varepsilon_{xz} \\ \varepsilon_{yx} & \varepsilon_{yy} & \varepsilon_{yz} \\ \varepsilon_{zx} & \varepsilon_{zy} & \varepsilon_{zz} \end{bmatrix}$$

Stress in 3D

Hooke's Law

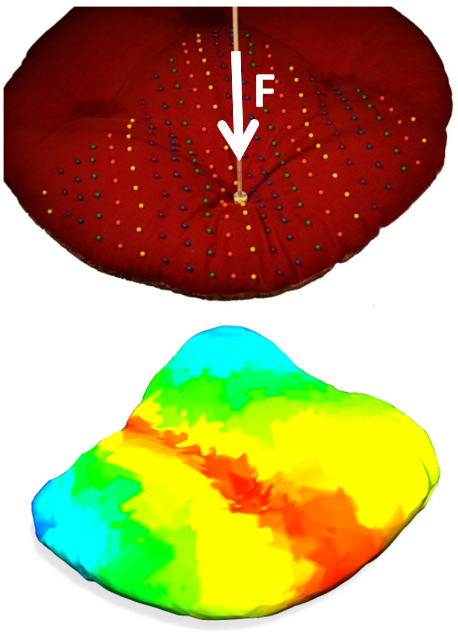
$$\sigma(\mathbf{u}) = \mathbf{E}\begin{bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \\ \varepsilon_{xy} \\ \varepsilon_{xz} \\ \varepsilon_{yz} \end{bmatrix}$$

Material Representation

Linear isotropic material

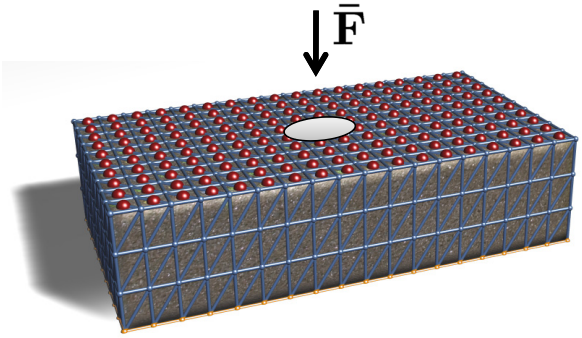
- Young's Modulus E
- Poisson's ratio ν

$$\mathbf{E} = \frac{E}{(1+\nu)(1-2\nu)}(\mathbf{G} + \nu\mathbf{H})$$



Young's Modulus

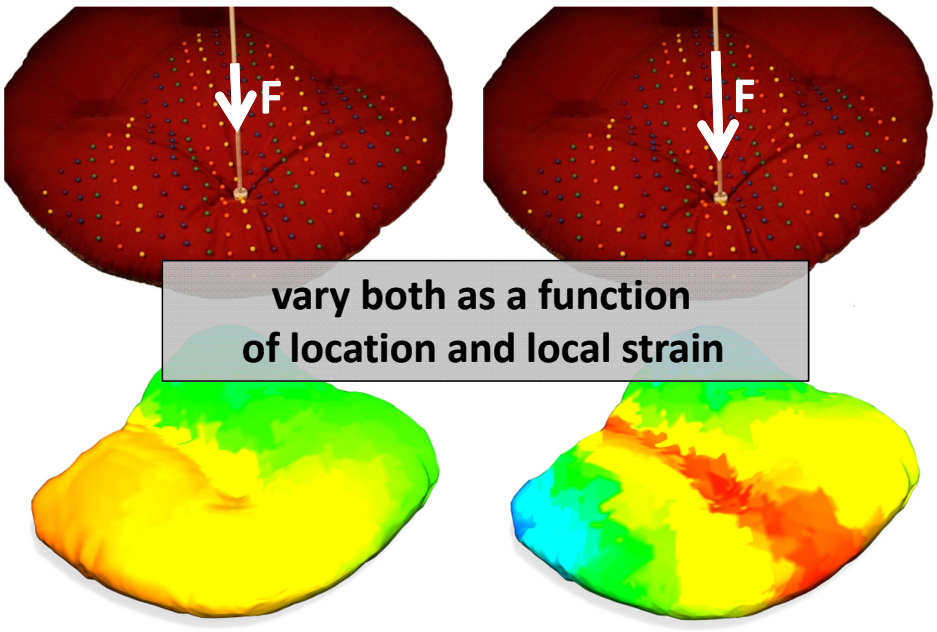
Parameter Fitting



$$\hat{\mathbf{p}} = \arg\min_{\mathbf{p}} \left\{ \sum_{i=1}^{n} ||\mathbf{x}_i(\mathbf{p}, \bar{\mathbf{F}}) - \bar{\mathbf{x}}_i||^2 + \gamma ||\mathbf{L}\mathbf{p}||^2 \right\}$$

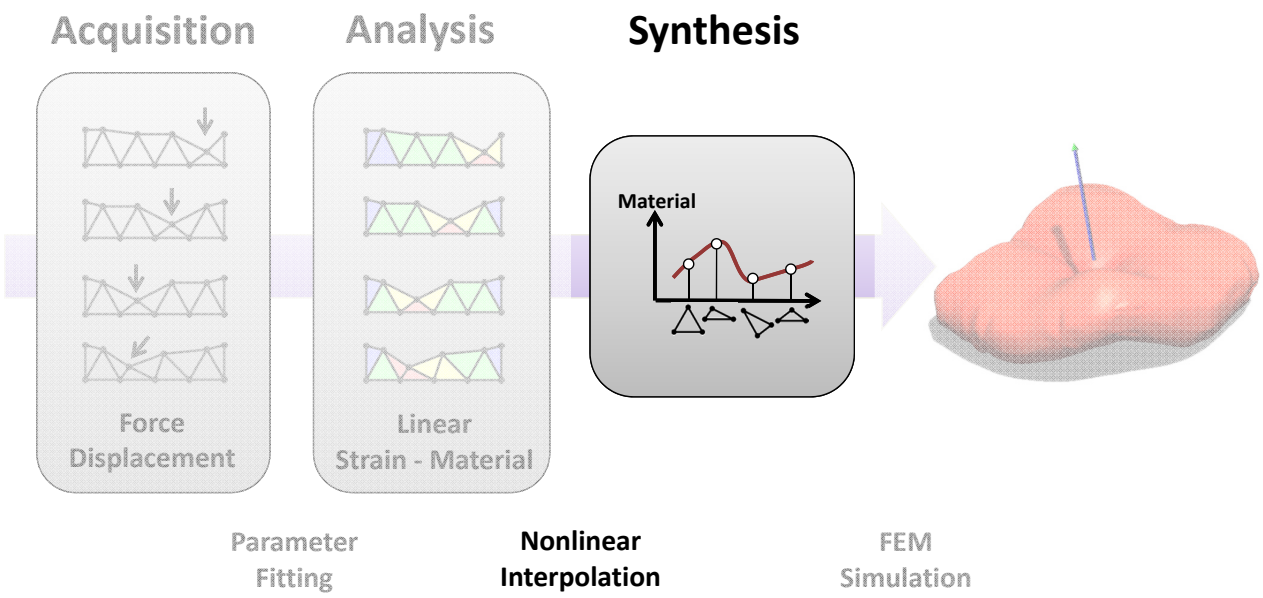
- Levenberg-Marquardt
- Typical fitting time per example deformation ~20min
- Details in the paper…

Fitted Parameters

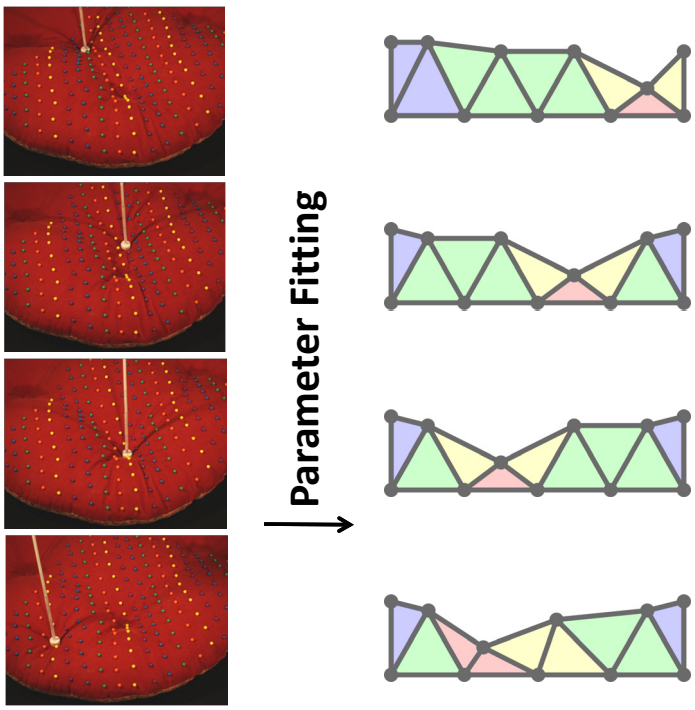


color coded
Young's Modulus

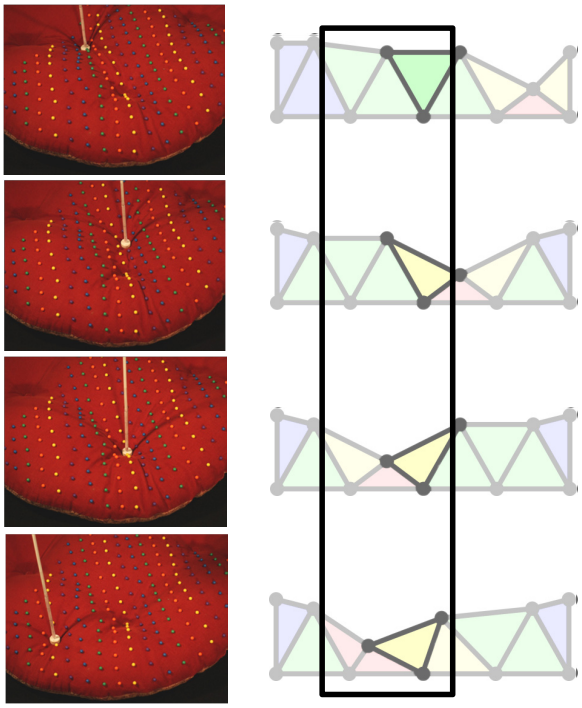
Overview



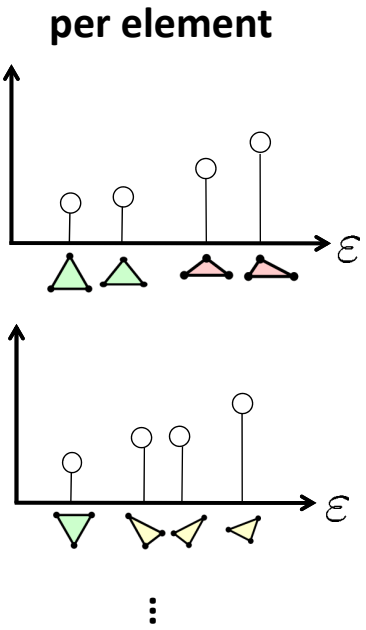
Strain-Space Interpolation



Strain-Space Interpolation



local



Strain-Space Interpolation

$$\mathbf{p}(\varepsilon) : \mathbb{R}^6 \rightarrow \mathbb{R}^2$$

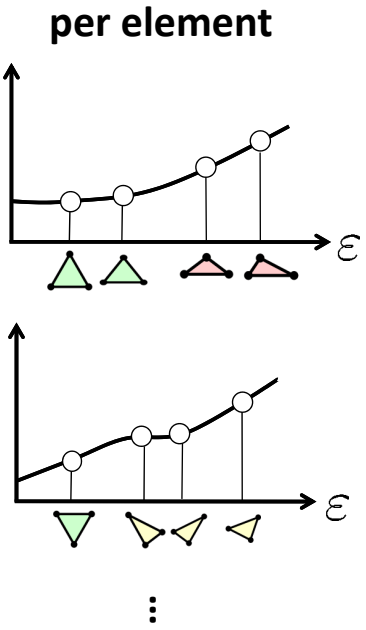
$$\mathbf{p}(\varepsilon) = \sum_{i=1}^{M} \mathbf{w}_i \cdot \varphi(\|\varepsilon - \varepsilon_i\|)$$

biharmonic RBF kernel $\varphi(r) = r$

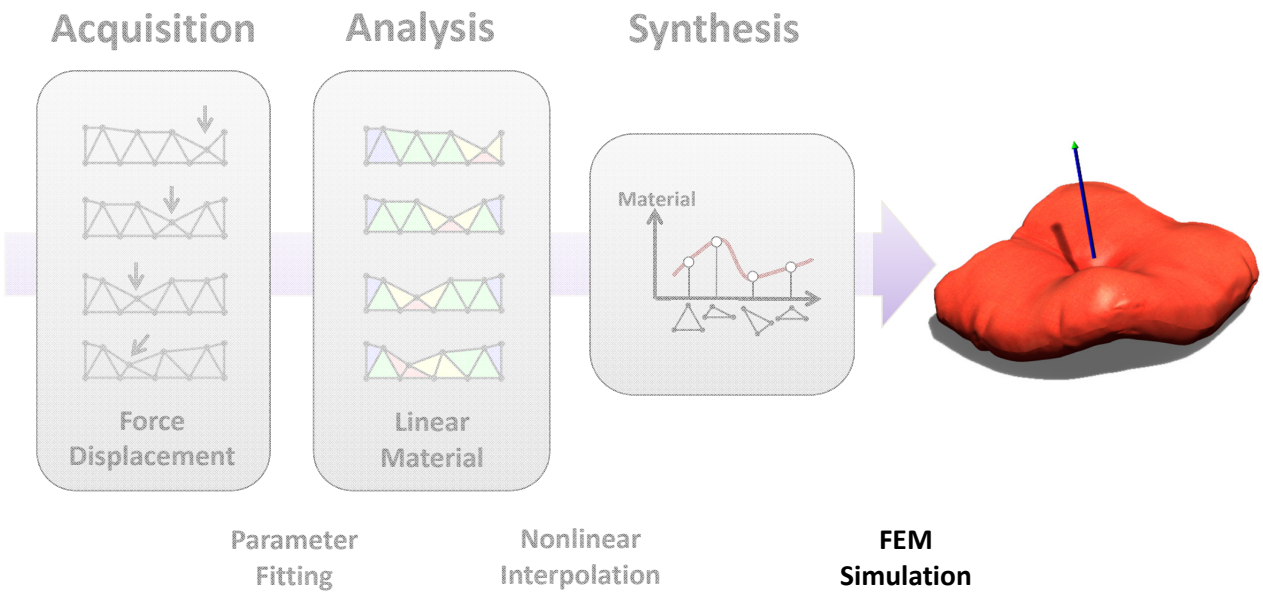
[Carr et al. 01]

$$\begin{pmatrix} \varphi_{1,1} & \varphi_{1,2} & \cdots & \varphi_{1,M} \\ \varphi_{2,1} & \varphi_{2,2} & \cdots & \varphi_{2,M} \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_{M,1} & \varphi_{M,2} & \cdots & \varphi_{M,M} \end{pmatrix} \cdot \begin{pmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \\ \vdots \\ \mathbf{w}_M \end{pmatrix} = \begin{pmatrix} \mathbf{p}(\varepsilon_1) \\ \mathbf{p}(\varepsilon_2) \\ \vdots \\ \mathbf{p}(\varepsilon_M) \end{pmatrix}$$

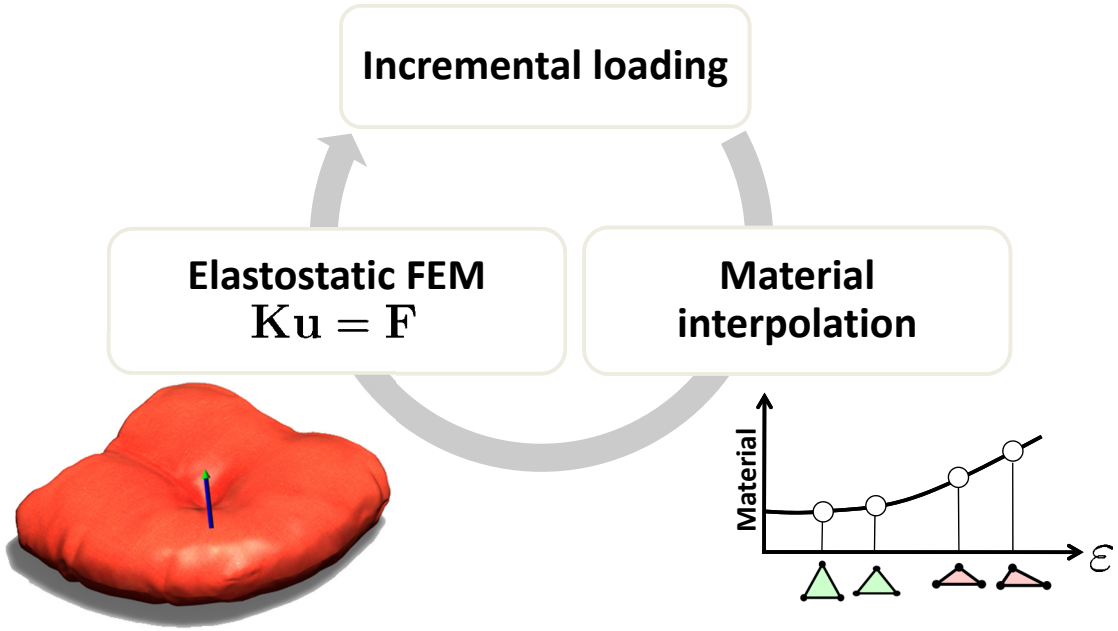
(solve M×M system per element)



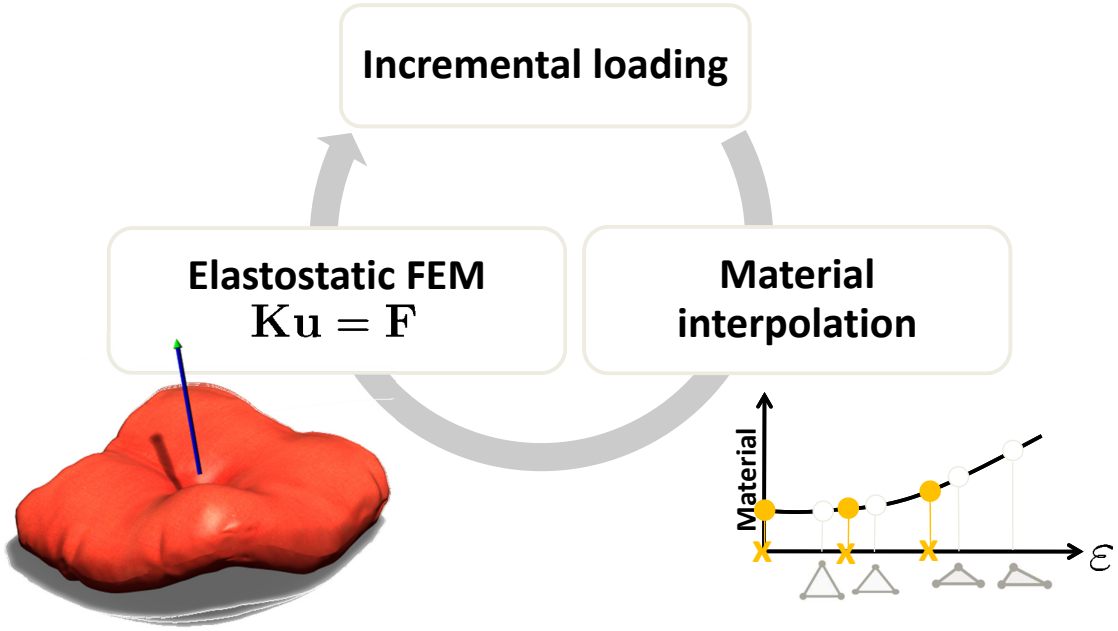
Overview



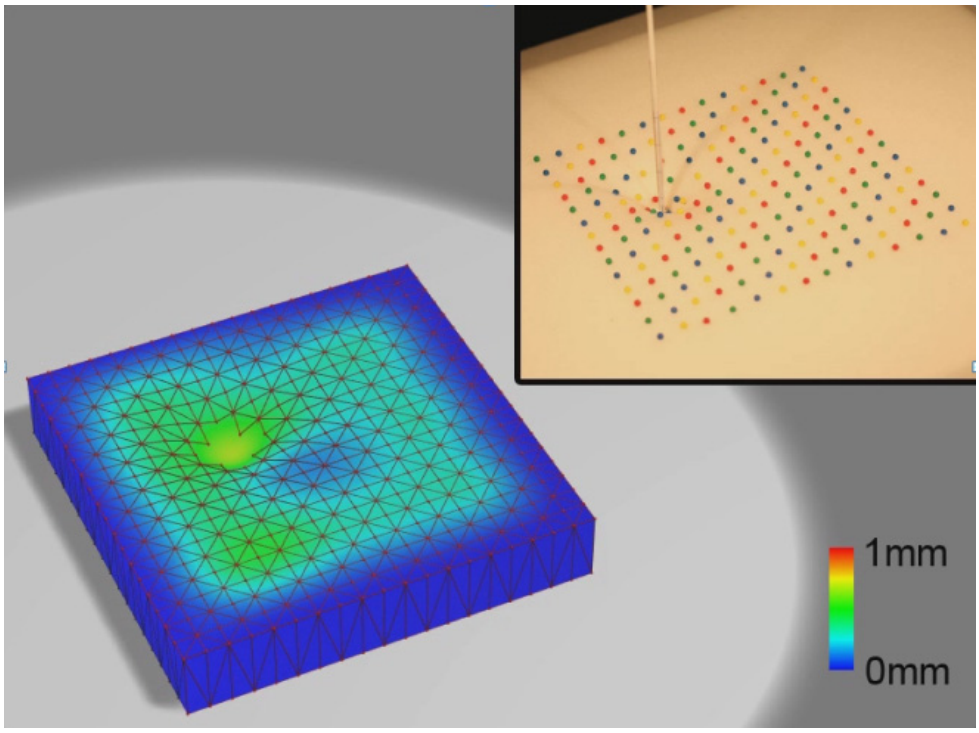
Elastostatic FEM Simulation



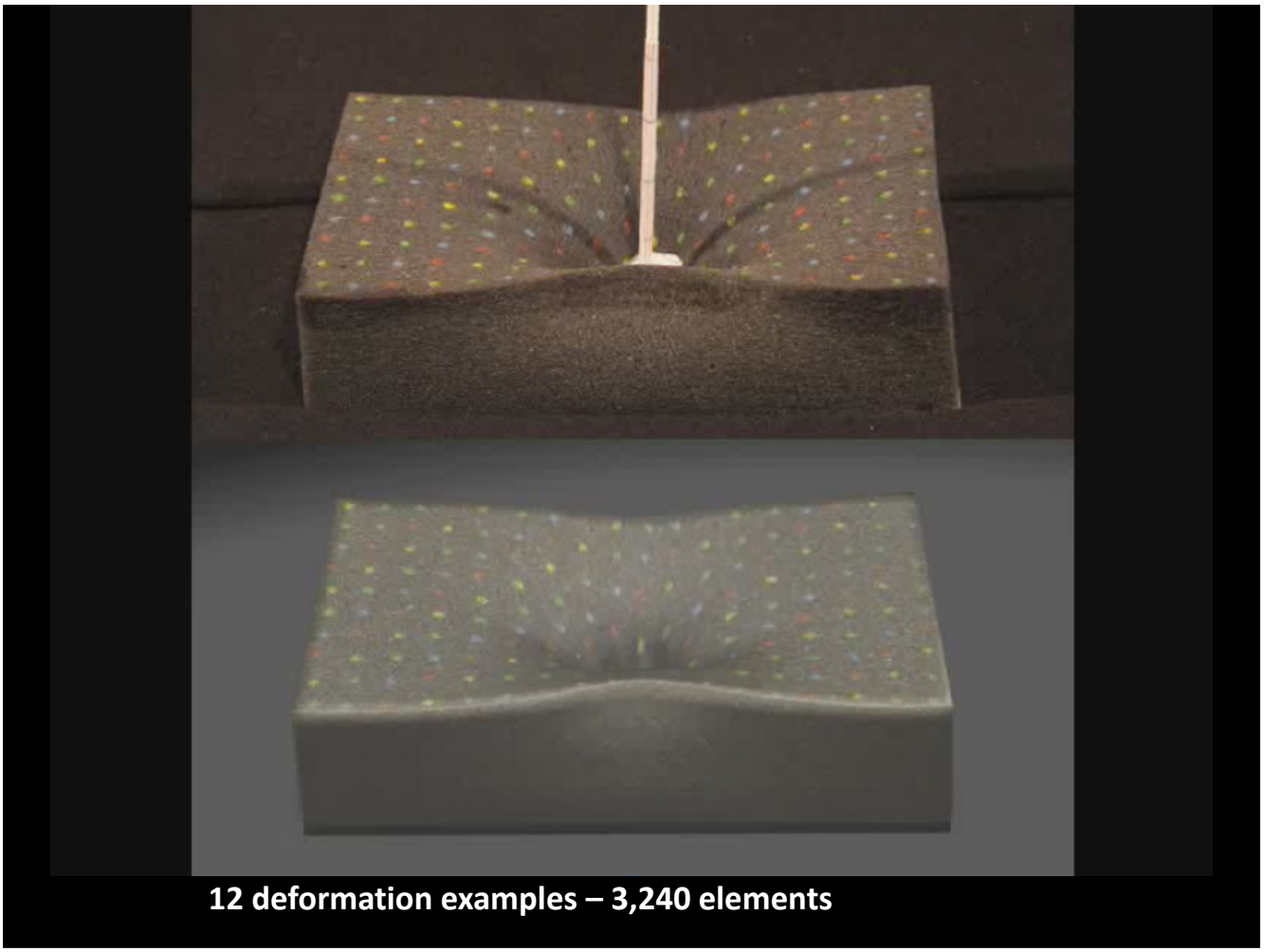
Elastostatic FEM Simulation



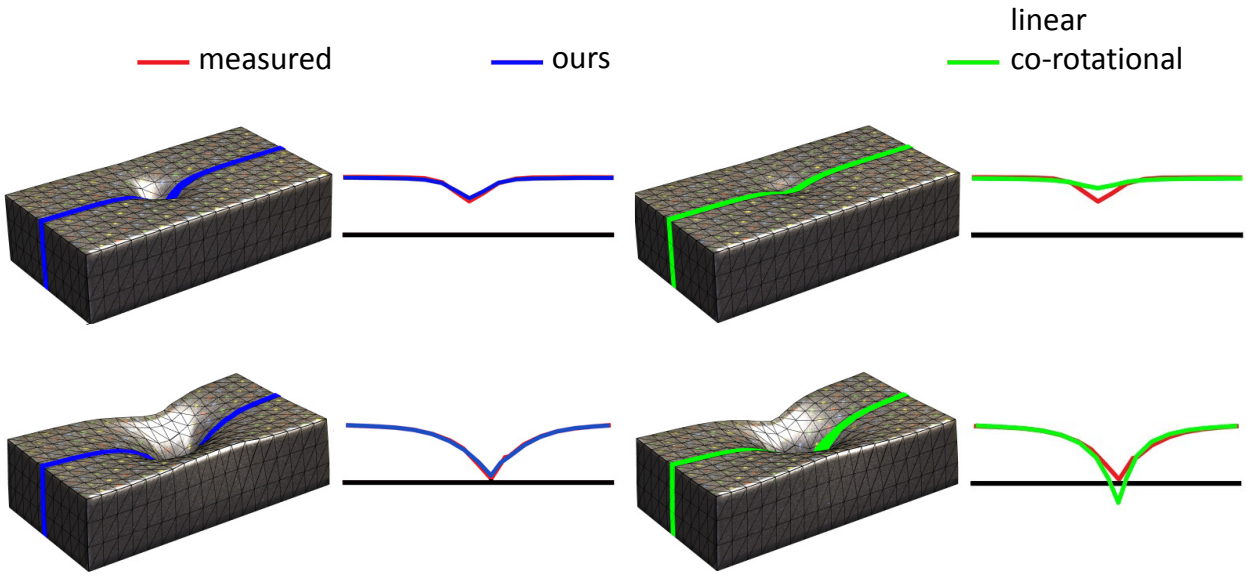
Results – Validation



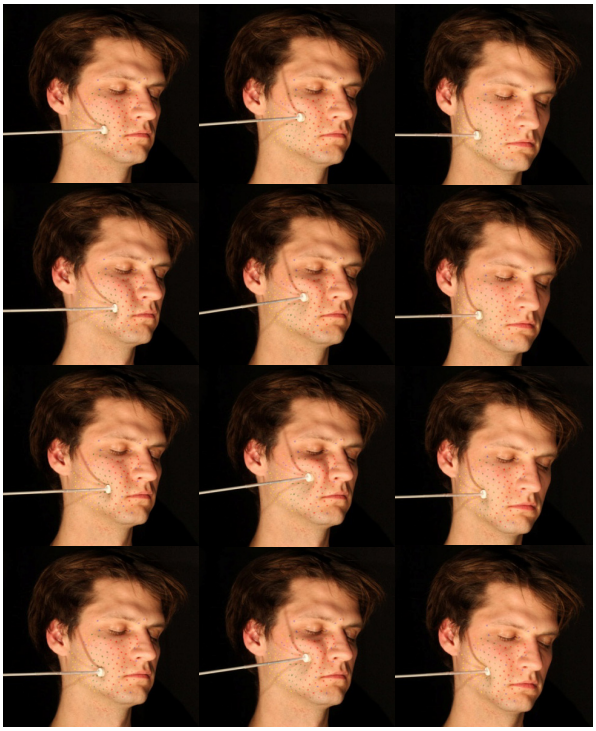
48 deformation examples – 1,805 elements



Results - Comparison



Results - Face Deformation

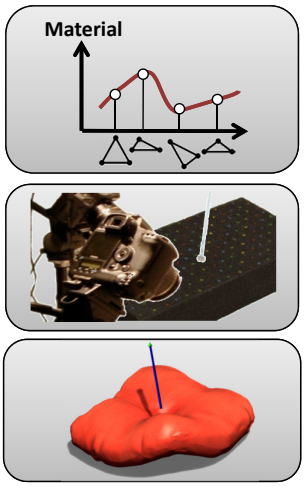


- **Simulation domain 8,261 elements**
- **Sliding contact between tissue and skull**
- **Smooth embedding based on MLS**

Summary

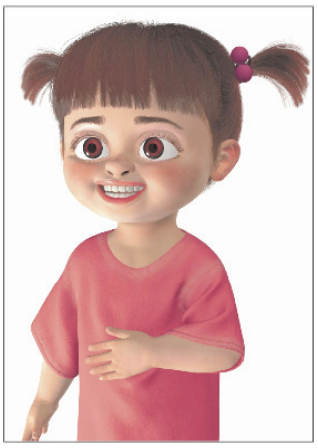
Data-driven method for soft tissue simulation

- Novel representation
  - Non-linear deformations
  - Heterogeneity
- Simple data acquisition
- Efficient deformation synthesis

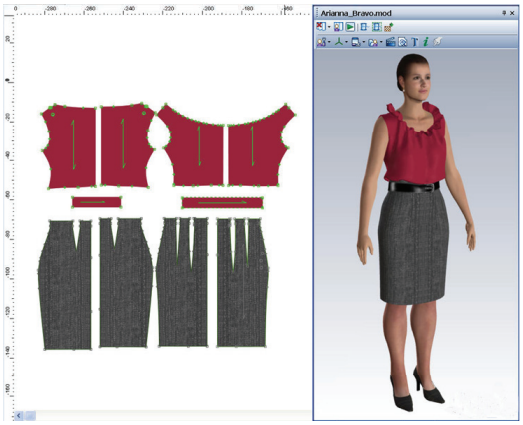


5 Data-Driven Modeling of Nonlinear Elasticity in Cloth

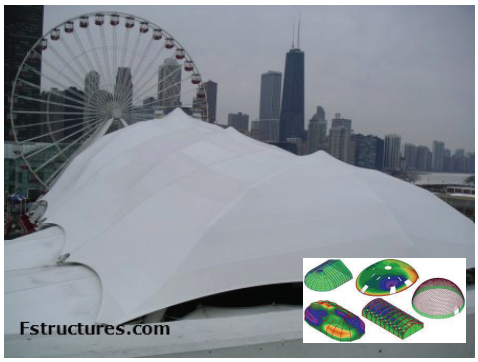
Motivation



[Baraff et al.]



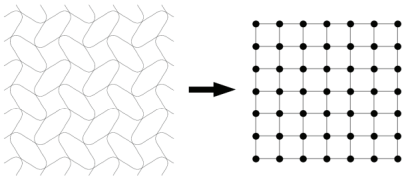
[www.optitex.com]



[www.fstructures.com]

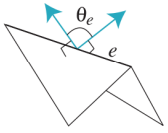
Cloth simulation is of great interest in computer graphics, the textile industry, and mechanical engineering

Motivation



[Particle System -- Breen]

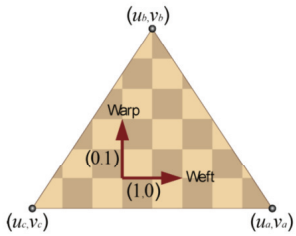
$$k_{ij}\left(\|\mathbf{x}_i - \mathbf{x}_j\| - l_{ij}^0\right)\frac{\mathbf{x}_i - \mathbf{x}_j}{\|\mathbf{x}_i - \mathbf{x}_j\|}$$

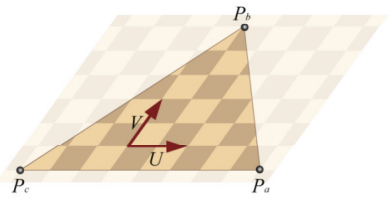


[Discrete Shells – Grinspun et al.]

$$W_B(\mathbf{x}) = \sum_e (\theta_e - \bar{\theta}_e)^2 \|\bar{e}\|/\bar{h}_e$$

$$W = W_M + k_B W_B$$



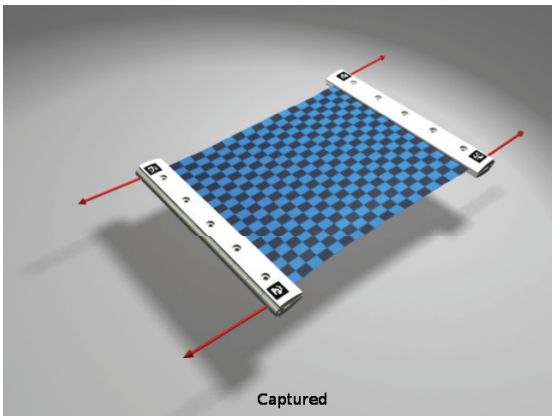


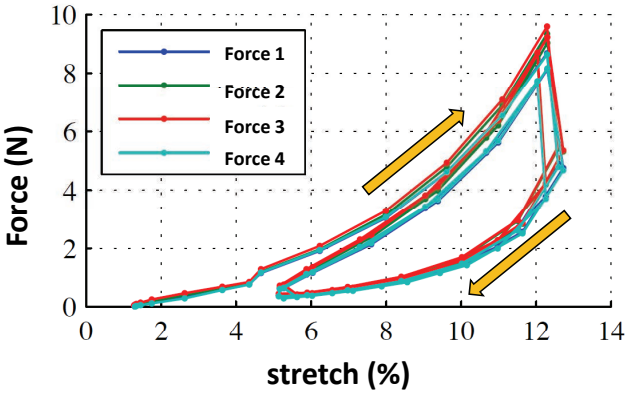
[StVK – Volino et al.]

$$\begin{bmatrix}\sigma_{uu}\\ \sigma_{vv}\\ \sigma_{uv}\end{bmatrix} = E\begin{bmatrix}\varepsilon_{uu}\\ \varepsilon_{vv}\\ \varepsilon_{uv}\end{bmatrix} + E'\begin{bmatrix}\varepsilon'_{uu}\\ \varepsilon'_{vv}\\ \varepsilon'_{uv}\end{bmatrix} \qquad E = \frac{e}{1-\nu^2}\begin{bmatrix}1 & \nu & 0\\ \nu & 1 & 0\\ 0 & 0 & \frac{1-\nu}{2}\end{bmatrix}$$

State-of-the-art cloth models rely on parameters, but parameters tuning is a tedious trial-and-error task

Motivation

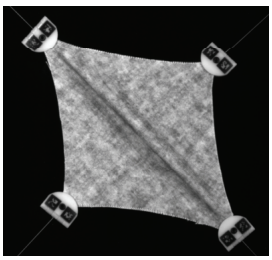




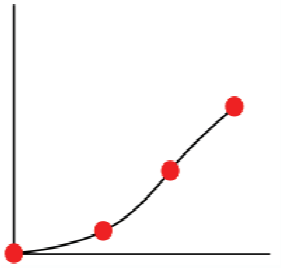
Real cloth exhibits strongly nonlinear behavior, including large hysteresis between loading and unloading cycles

Automatic Parameter Tuning

1. Capture real cloth behavior.



2. Estimate parameters automatically.



3. Evaluate performance of different cloth models.

Spring

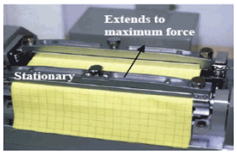
St. VK

Soft Constraints

Classic Approaches

Kawabata'80
Volino'09
…

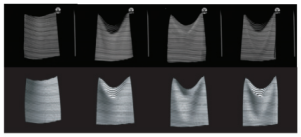
FORCE-BASED MEASUREMENTS



- Force vs displacement measurements
- Isolate individual deformation modes
- Uniform deformation

Bhat'03
Kunitomo'10
Stoll'10
…

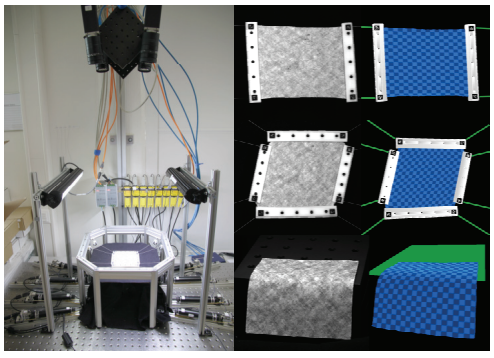
DYNAMIC CAPTURED VIDEOS



- Reduced controlled conditions
- Impossible to separate internal and external parameters
- No force information is available

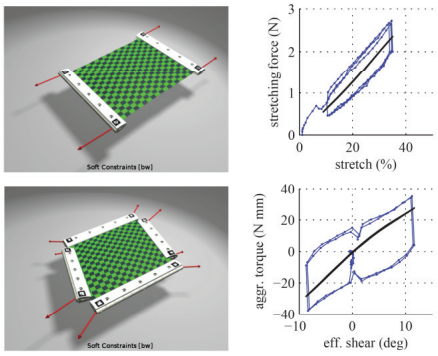
Approach 1 [Miguel et al. 2012]

MEASUREMENT



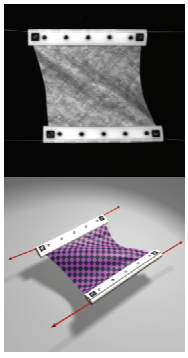
- **Capture system**
- **Detailed 3D geometry**
- **Force measurements**

MODEL FITTING



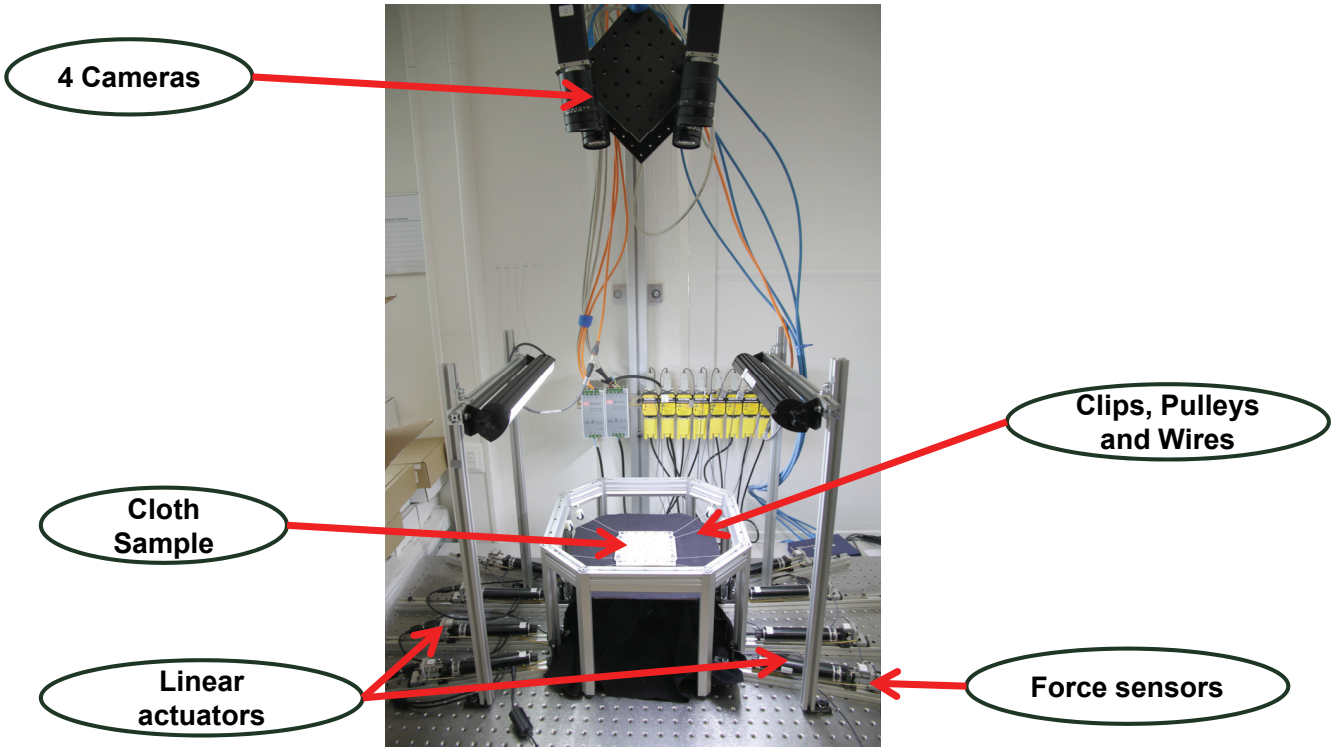
- **Fitting method**
- **Estimated parameters**

VALIDATION

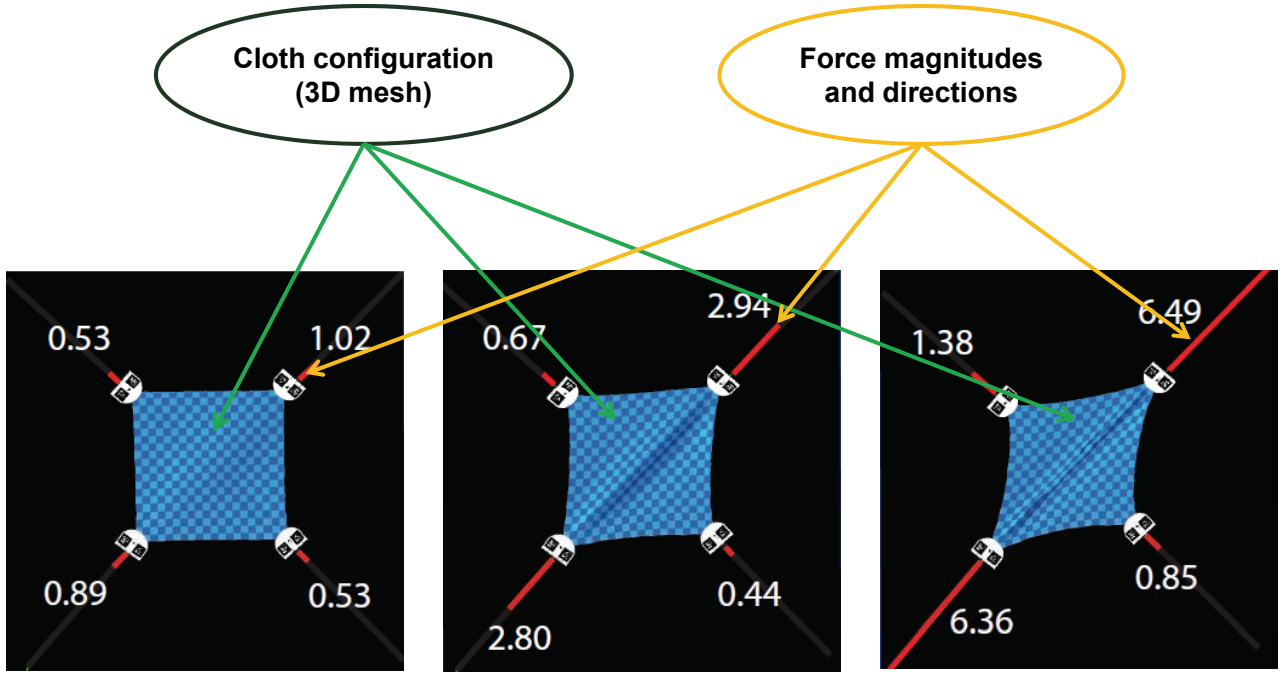


- **Insight on evaluated models**

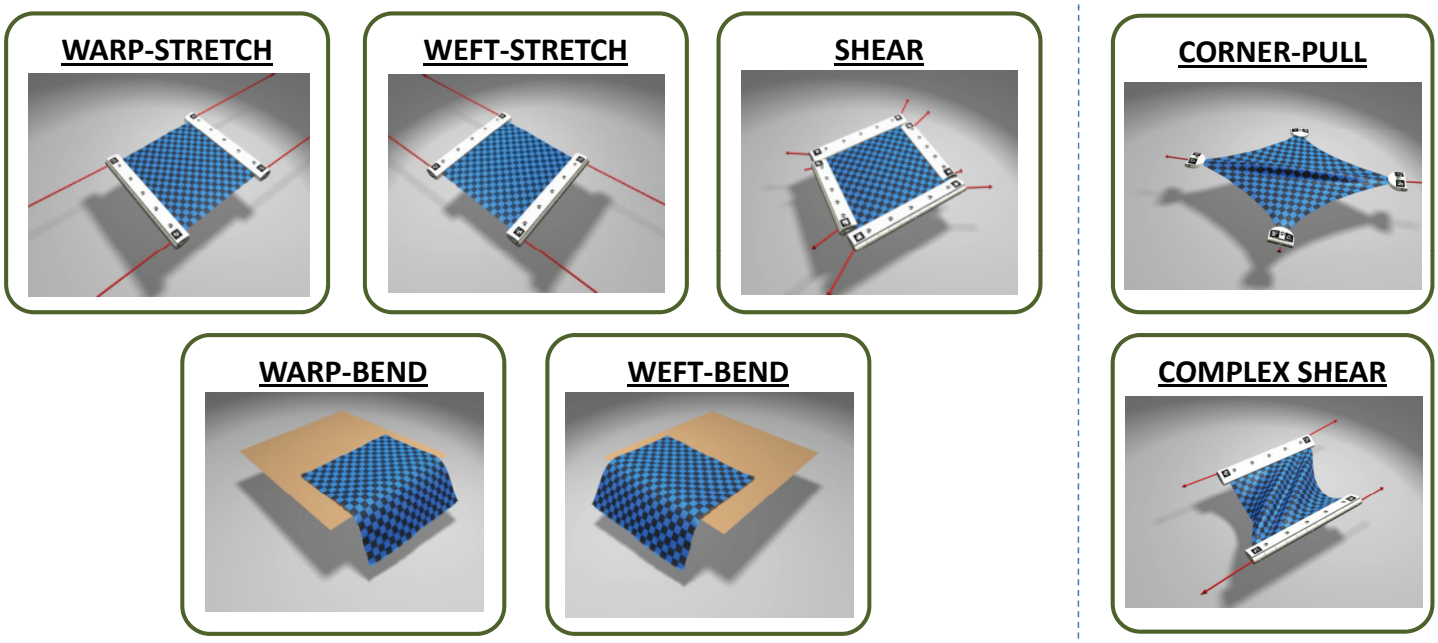
Measurement: Hardware Overview



Measurement: Measured Data



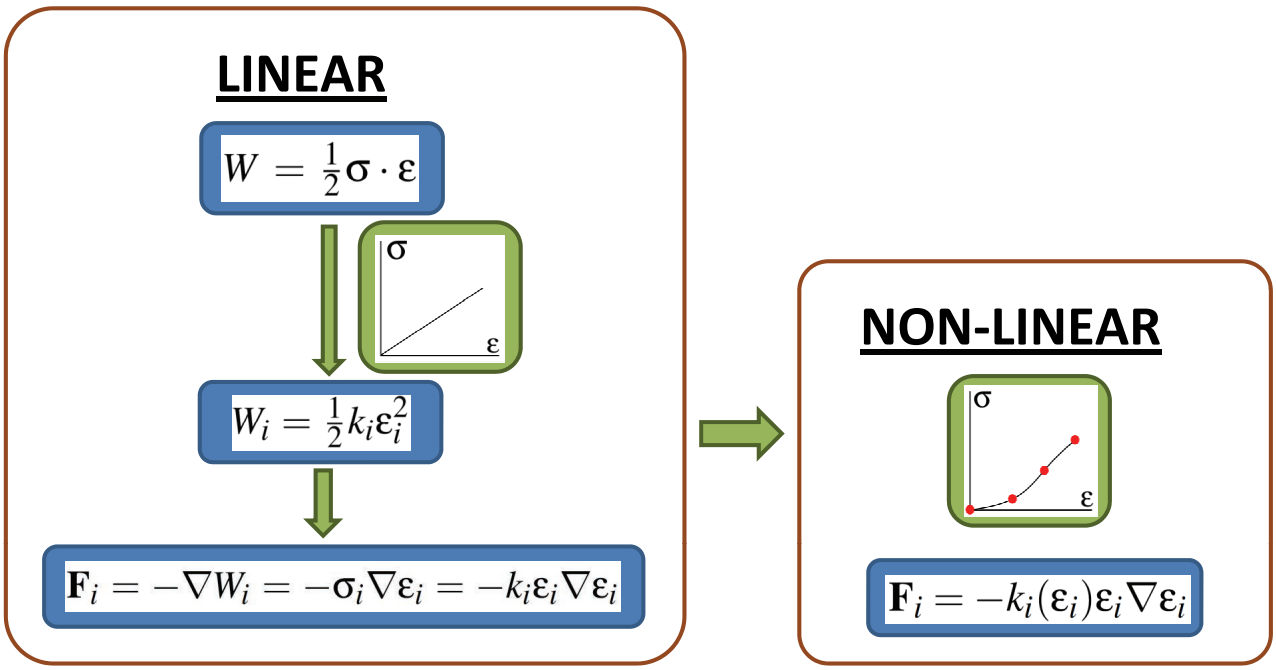
Measurement: Experiments



5 experiments with near-isolated strain, used for data fitting

2 experiments with complex strain, used for validation

Model Fitting: Cloth Models



Energy is defined as the stress-strain product. The left block derives the force for each deformation mode for a linear, separable model. On the right, the model is extended to account for nonlinear elasticity, by interpolating stiffness from control point values.

Model Fitting: Cloth Models

MEMBRANE

- Spring
- Volino'09
- Baraff & Witkin'98
- …

BENDING

- Springs
- Grinspun'03
- Bridson'03
- Bergou'06
- Garg'07
- …

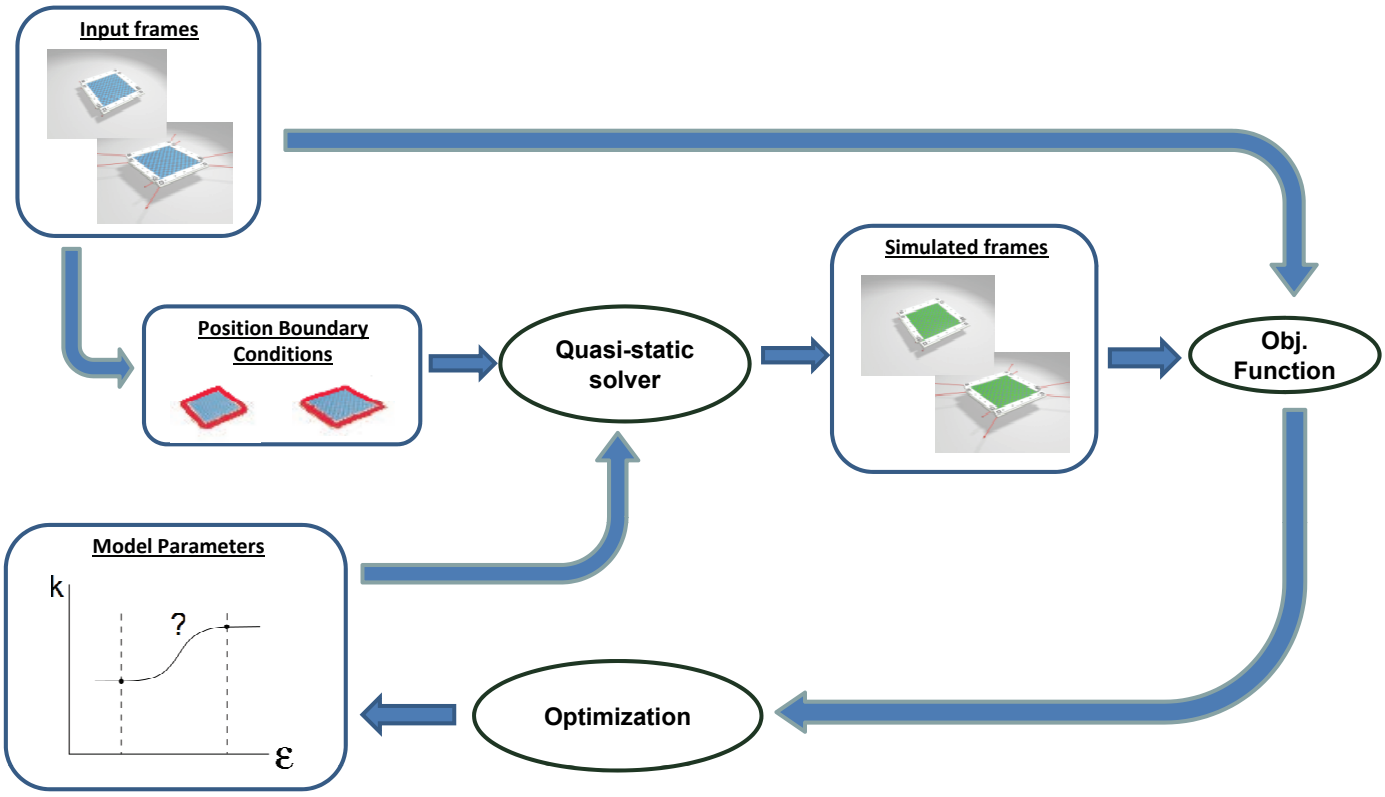
Many popular models can be written in this fashion, and some of them are picked to define three example membrane-bending models:

Spring
Spring Membrane
Spring Bending

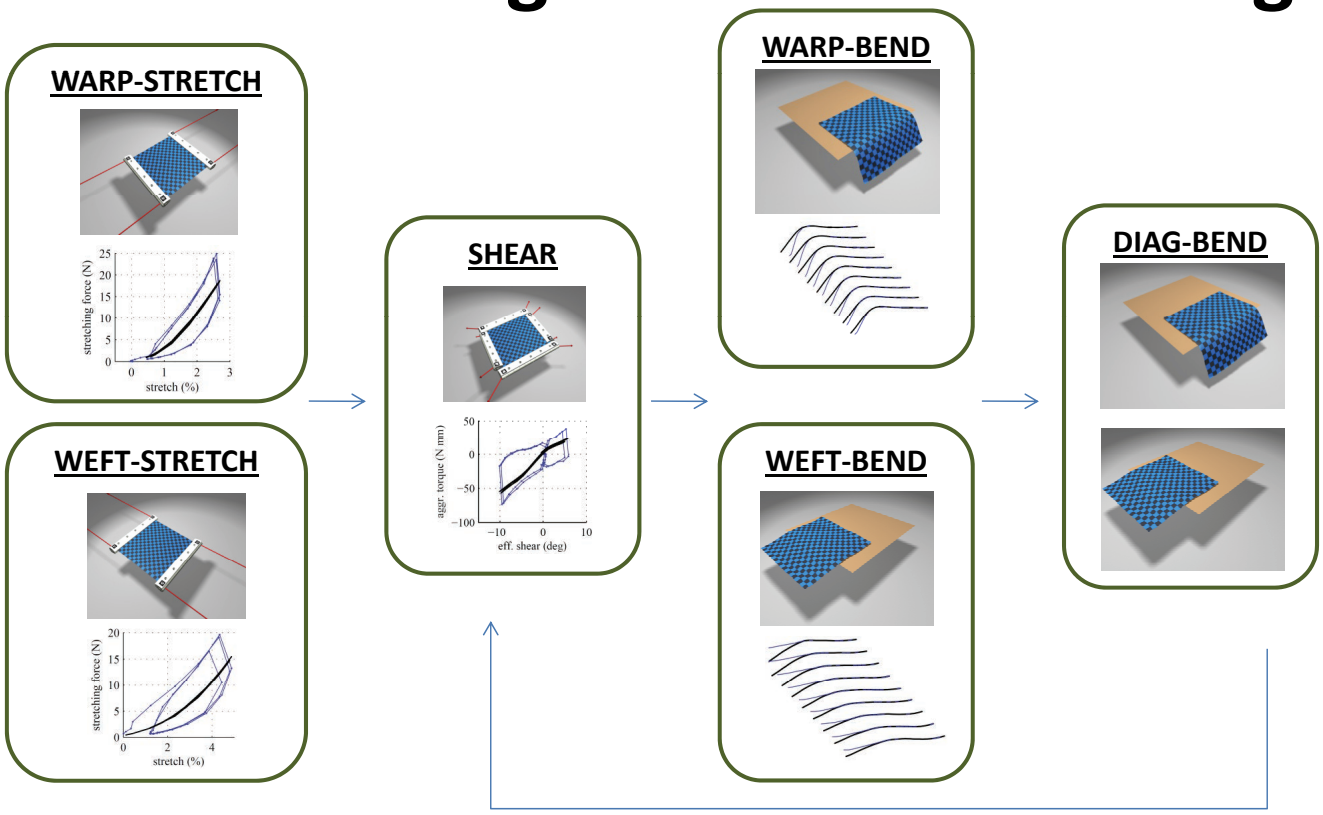
Soft Constraints
Baraff & Witkin
Discrete Shells Bending

St. VK
Diagonalized St. VK
Discrete Shells Bending

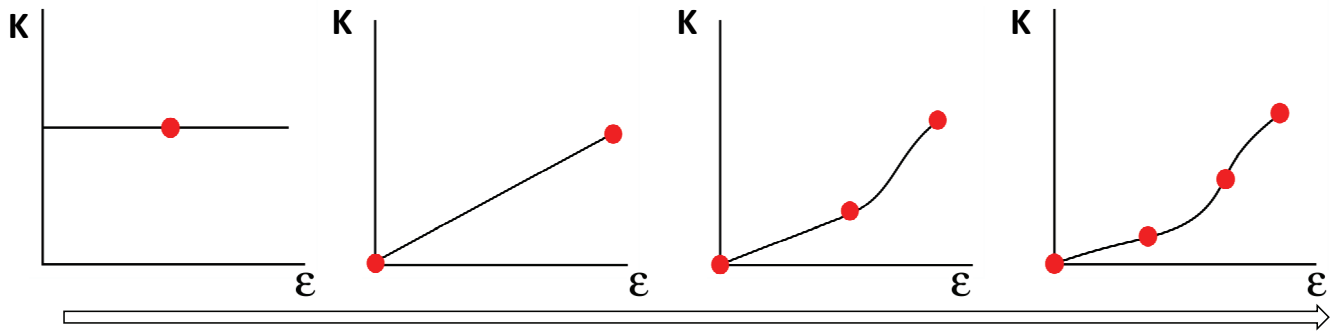
Model Fitting: Optimization Loop

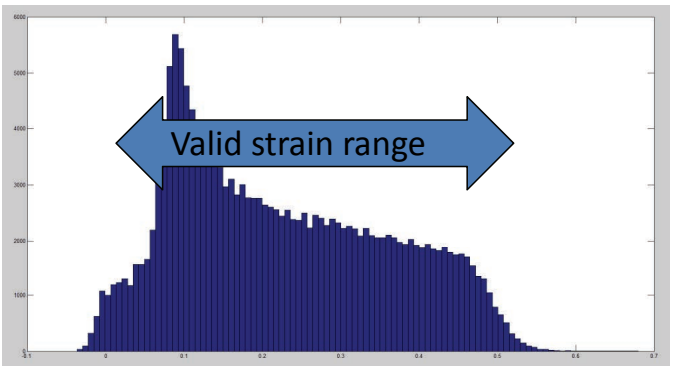


Model Fitting: Incremental Fitting



Model Fitting: Control Point Insertion

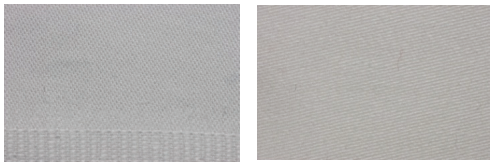




Validation: Real Cloth Samples

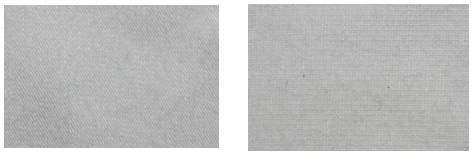
Cotton satin

- Very stiff in stretch
- Compliant in bending



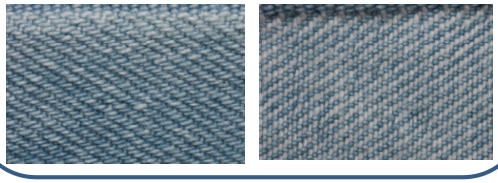
Rayon/spandex knit

- Isotropic
- Compliant in stretch and bending



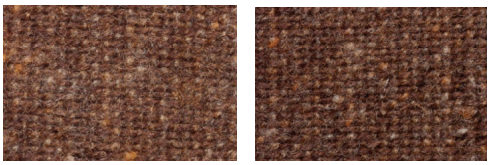
Cotton denim

- Stiff and quite isotropic in stretch
- Extremely anisotropic in bending



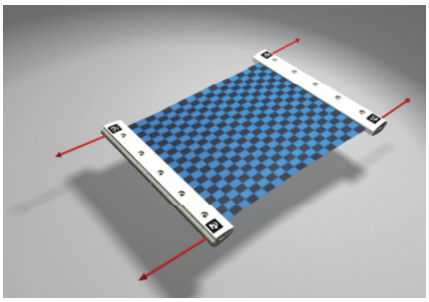
Wool/cotton blend

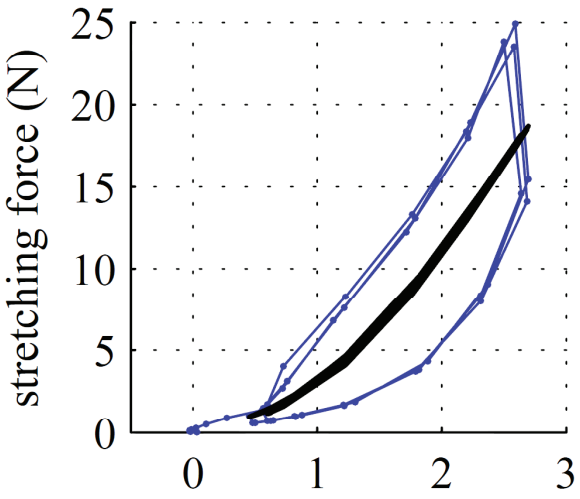
- Anisotropic in stretch and bending



Validation: Stretch

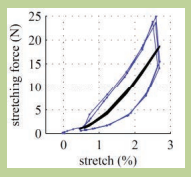
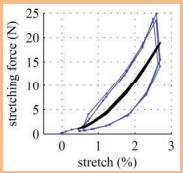
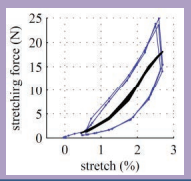
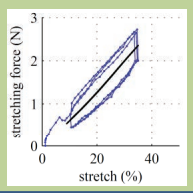
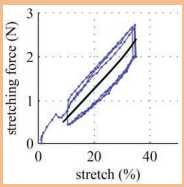
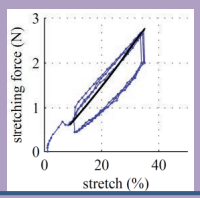
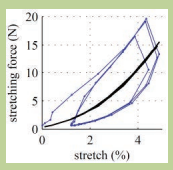
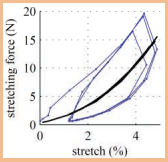
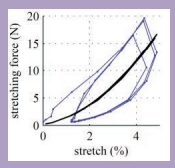
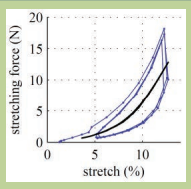
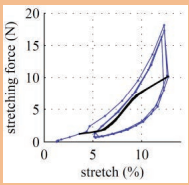
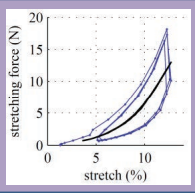
WARP-STRETCH





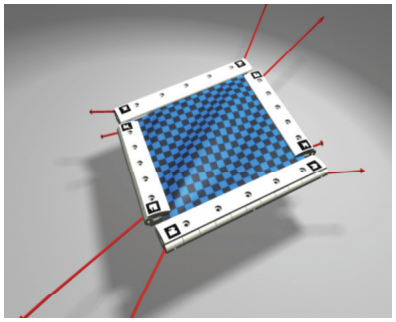
A plot compares measured force-displacement in loading-unloading cycles to simulated force

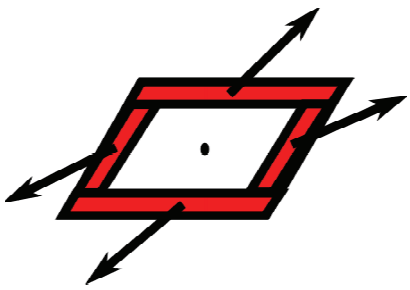
Validation: Stretch

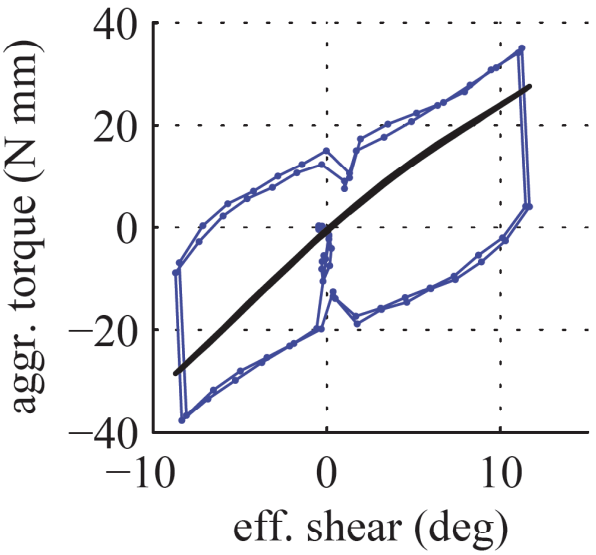
	Soft Constraints	St.VK	Spring
Cotton satin			
Rayon/spandex knit			
Cotton denim			
Wool/cotton blend			

Validation: Shear

SHEAR

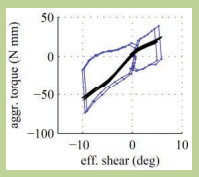
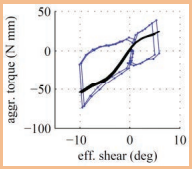
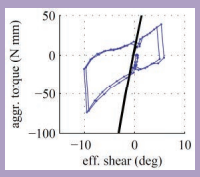
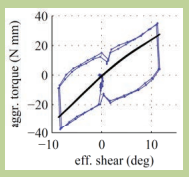
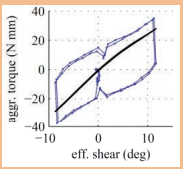
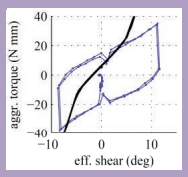
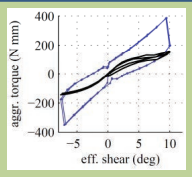
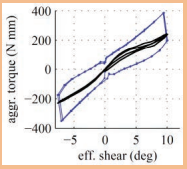
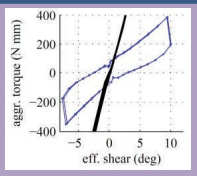
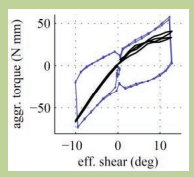
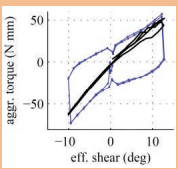
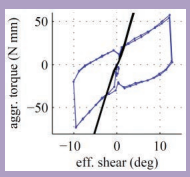






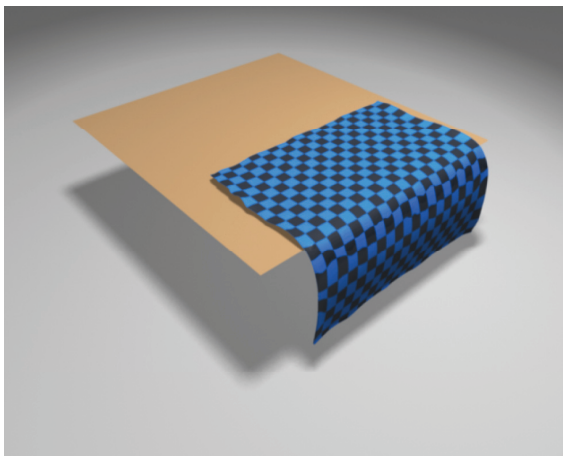
This time the plot shows aggregate torque vs. effective shear angle

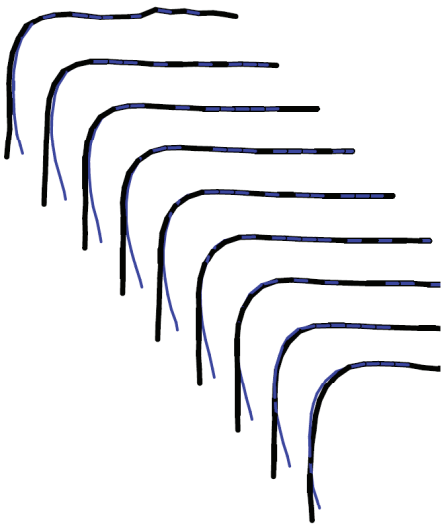
Validation: Shear

	Soft Constraints	St.VK	Spring
Cotton satin			
Rayon/spandex knit			
Cotton denim			
Wool/cotton blend			

Validation: Shear

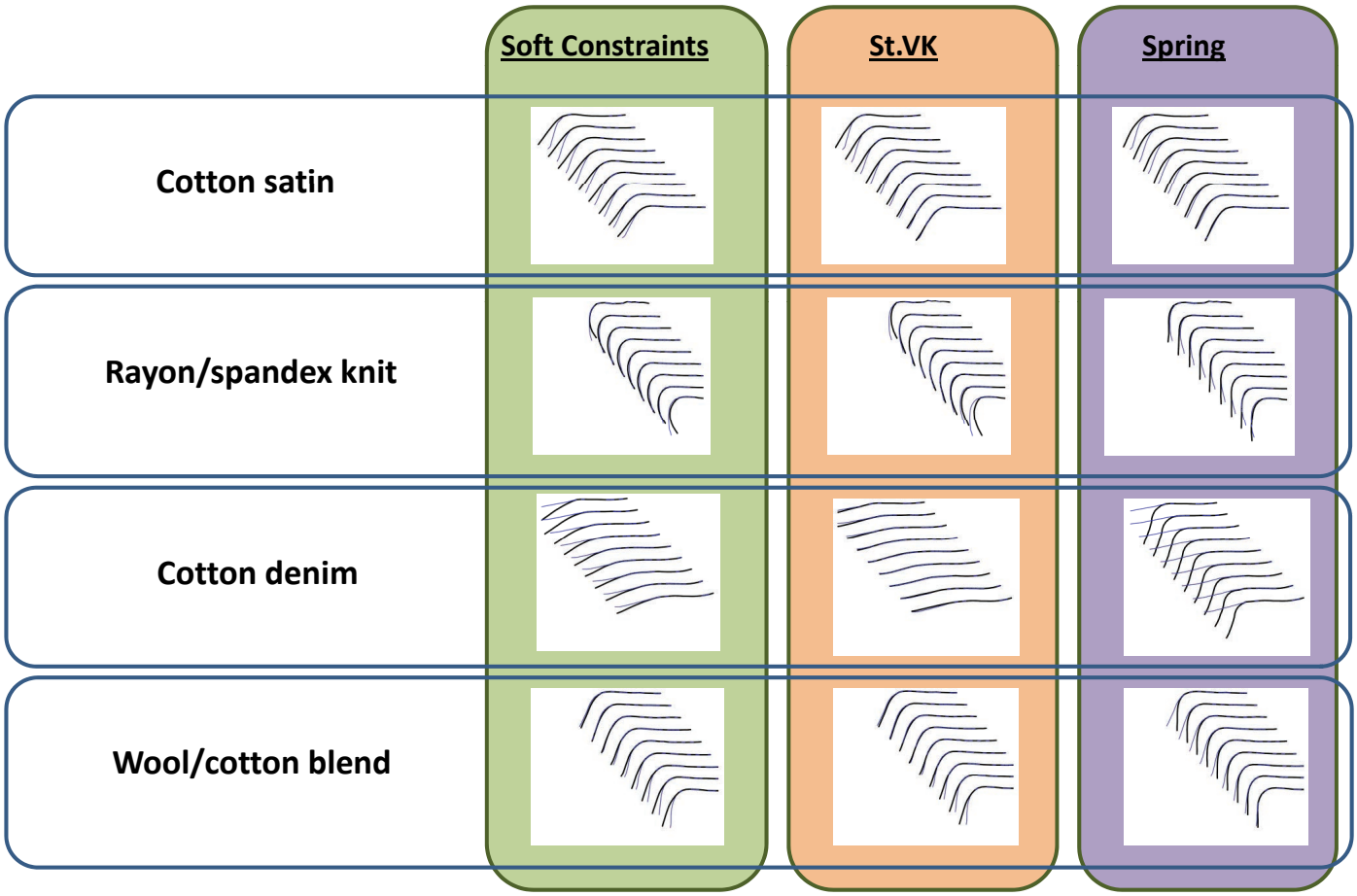
WARP-BENDING



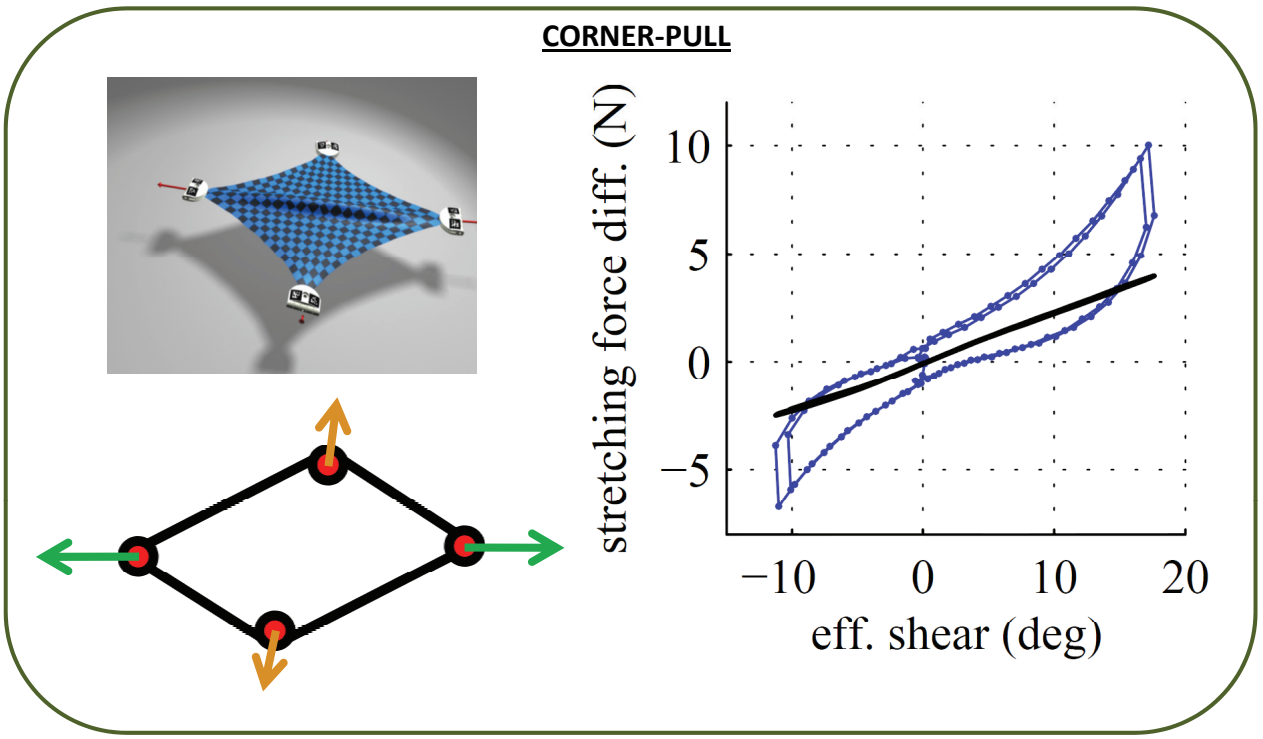


The plot compares measured and simulated cloth profiles

Validation: Bend

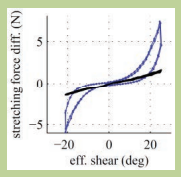
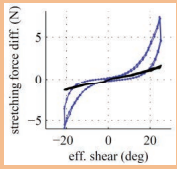
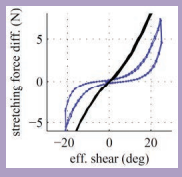
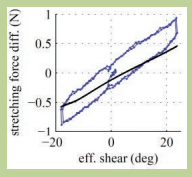
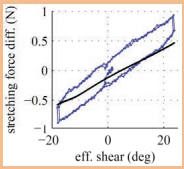
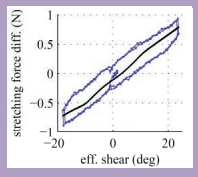
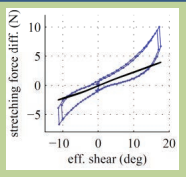
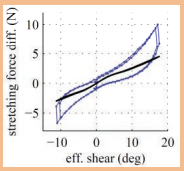
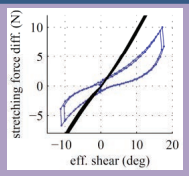
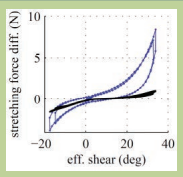
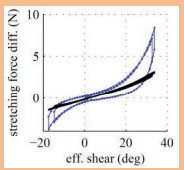
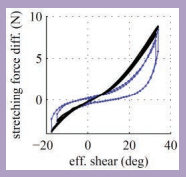


Validation: Corner-Pull



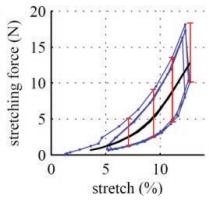
Validation under experiments not used for fitting

Validation: Corner-Pull

	Soft Constraints	St.VK	Spring
Cotton satin			
Rayon/spandex knit			
Cotton denim			
Wool/cotton blend			

Limitations

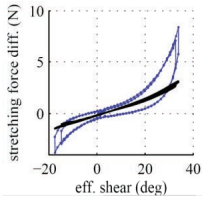
Hysteresis



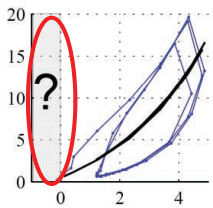
Poisson

$$W = \frac{1}{2}\begin{pmatrix}\varepsilon_0 & \varepsilon_1 & \varepsilon_2\end{pmatrix}\begin{pmatrix}k_{00} & k_{01} & 0\\ k_{01} & k_{11} & 0\\ 0 & 0 & k_{22}\end{pmatrix}\begin{pmatrix}\varepsilon_0\\ \varepsilon_1\\ \varepsilon_1\end{pmatrix}$$

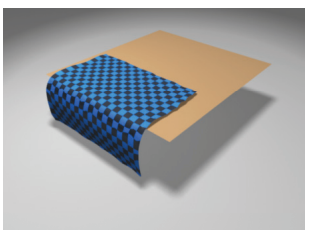
Cross-Modal Stiffening



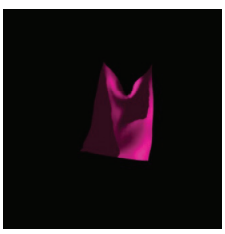
Compression



Bending



Dynamics

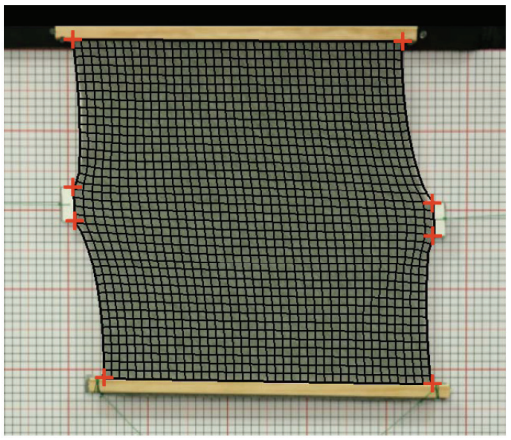


Approach 2 [Wang et al. 2011]

ELASTIC MODEL

$$\begin{bmatrix} \sigma_u \\ \sigma_v \\ \sigma_s \end{bmatrix} = \begin{bmatrix} C_{uu}(\varepsilon_{max},\varphi) & C_{uv}(\varepsilon_{max},\varphi) & \\ C_{uv}(\varepsilon_{max},\varphi) & C_{vv}(\varepsilon_{max},\varphi) & \\ & & C_{ss}(\varepsilon_{max},\varphi) \end{bmatrix} \begin{bmatrix} \varepsilon_u \\ \varepsilon_v \\ \varepsilon_s \end{bmatrix}$$

DATA CAPTURE



FITTING

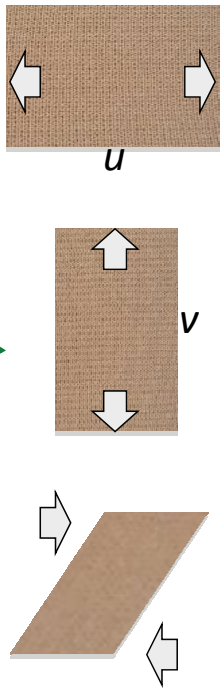
Elastic Model: Deformation Modes

- Infinitesimal strain theory

$$\begin{bmatrix} \sigma_u \\ \sigma_v \\ \sigma_s \end{bmatrix} = \mathbf{E} \left(\begin{bmatrix} \varepsilon_u \\ \varepsilon_v \\ \varepsilon_s \end{bmatrix} \right)$$

Stress Tensor

Strain Tensor



Elastic Model: Linear Model

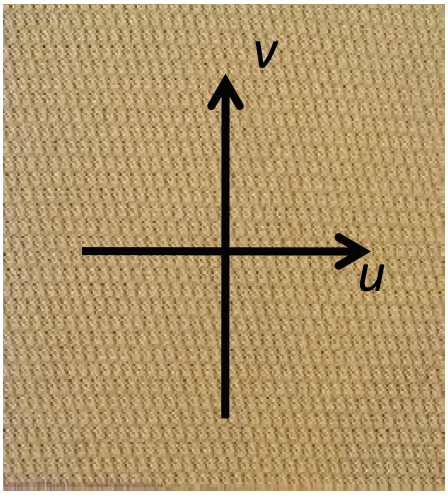
- A full linear model

$$\begin{bmatrix} \sigma_u \\ \sigma_v \\ \sigma_s \end{bmatrix} = \begin{bmatrix} c_{uu} & c_{uv} & c_{us} \\ c_{uv} & c_{vv} & c_{vs} \\ c_{us} & c_{vs} & c_{ss} \end{bmatrix} \begin{bmatrix} \varepsilon_u \\ \varepsilon_v \\ \varepsilon_s \end{bmatrix}$$

Elastic Model: Orthotropic Model

- Orthotropic model: linear, anisotropic, symmetric

$$\begin{bmatrix} \sigma_u \\ \sigma_v \\ \sigma_s \end{bmatrix} = \begin{bmatrix} c_{uu} & c_{uv} & \\ c_{uv} & c_{vv} & \\ & & c_{ss} \end{bmatrix} \begin{bmatrix} \varepsilon_u \\ \varepsilon_v \\ \varepsilon_s \end{bmatrix}$$



Elastic Model: Nonlinear Model

- Nonlinear orthotropic model

$$\begin{bmatrix} \sigma_u \\ \sigma_v \\ \sigma_s \end{bmatrix} = \begin{bmatrix} C_{uu}(\varepsilon_u,\varepsilon_v,\varepsilon_s) & C_{uv}(\varepsilon_u,\varepsilon_v,\varepsilon_s) & \\ C_{uv}(\varepsilon_u,\varepsilon_v,\varepsilon_s) & C_{vv}(\varepsilon_u,\varepsilon_v,\varepsilon_s) & \\ & & C_{ss}(\varepsilon_u,\varepsilon_v,\varepsilon_s) \end{bmatrix} \begin{bmatrix} \varepsilon_u \\ \varepsilon_v \\ \varepsilon_s \end{bmatrix}$$

Elastic Model: 2D Parameterization

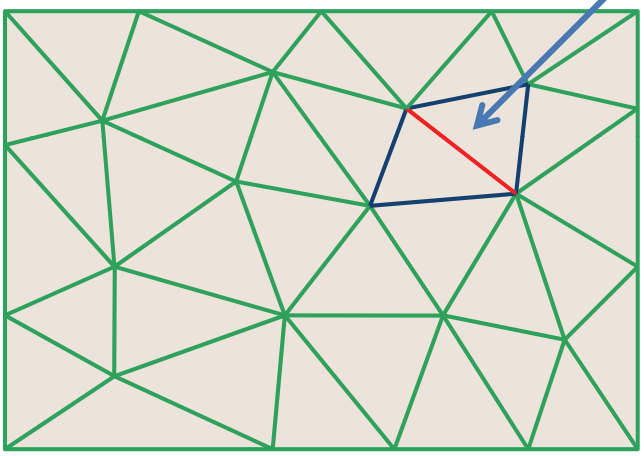
- Simplified model

$$\begin{bmatrix} \sigma_u \\ \sigma_v \\ \sigma_s \end{bmatrix} = \begin{bmatrix} C_{uu}(\varepsilon_{\max},\varphi) & C_{uv}(\varepsilon_{\max},\varphi) & \\ C_{uv}(\varepsilon_{\max},\varphi) & C_{vv}(\varepsilon_{\max},\varphi) & \\ & & C_{ss}(\varepsilon_{\max},\varphi) \end{bmatrix} \begin{bmatrix} \varepsilon_u \\ \varepsilon_v \\ \varepsilon_s \end{bmatrix}$$

in which

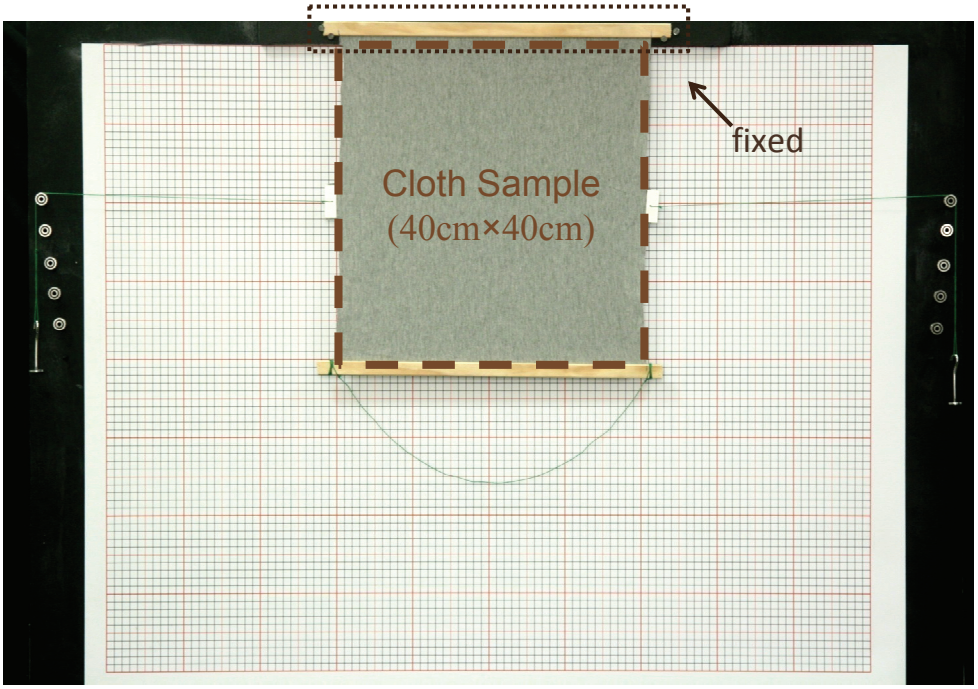
$$\begin{bmatrix} \varepsilon_u & \varepsilon_s \\ \varepsilon_s & \varepsilon_v \end{bmatrix} = \begin{bmatrix} \cos\varphi & \sin\varphi \\ -\sin\varphi & \cos\varphi \end{bmatrix} \begin{bmatrix} \varepsilon_{\max} & \\ & \varepsilon_{\min} \end{bmatrix} \begin{bmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{bmatrix}$$

Elastic Model: Bending

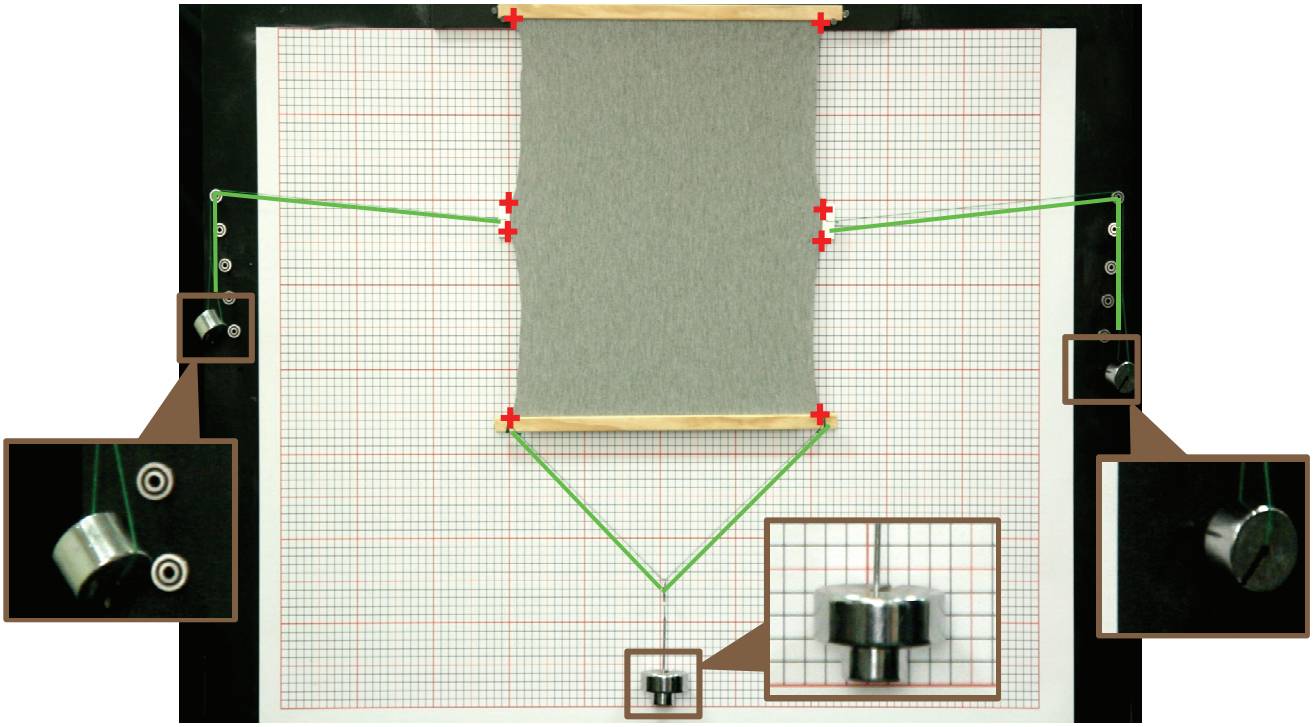


- Stiffness K
  - Edge orientation (anisotropic)
  - Bending angle (nonlinear)

Data Capture: Rest Conditions

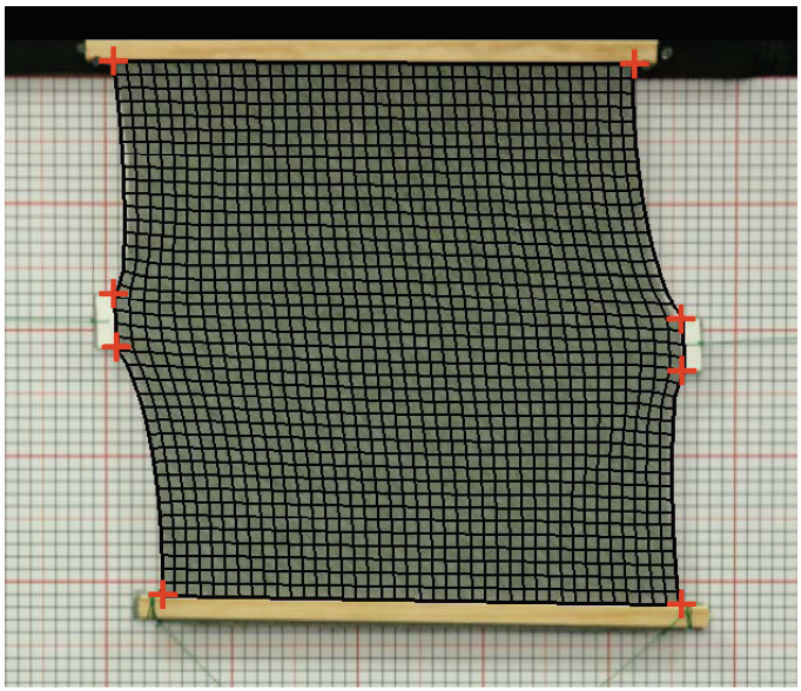


Data Capture: Deformation

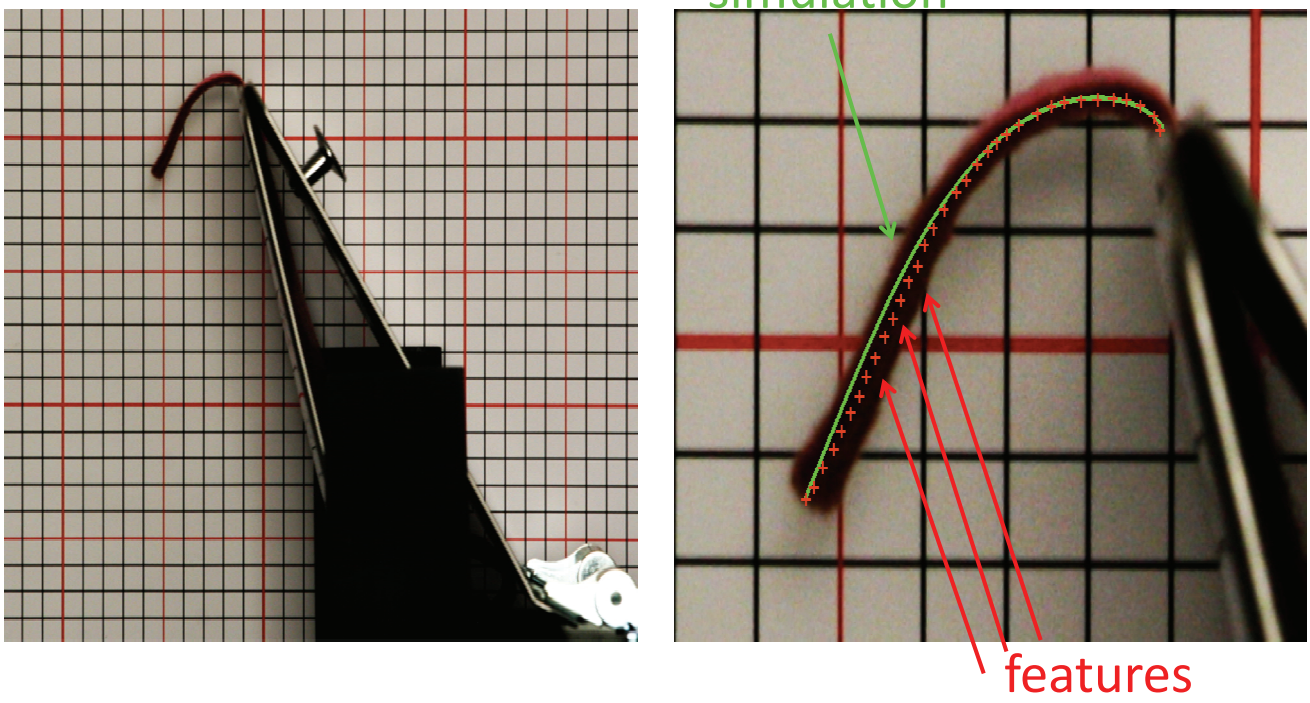


Data Capture: Accuracy

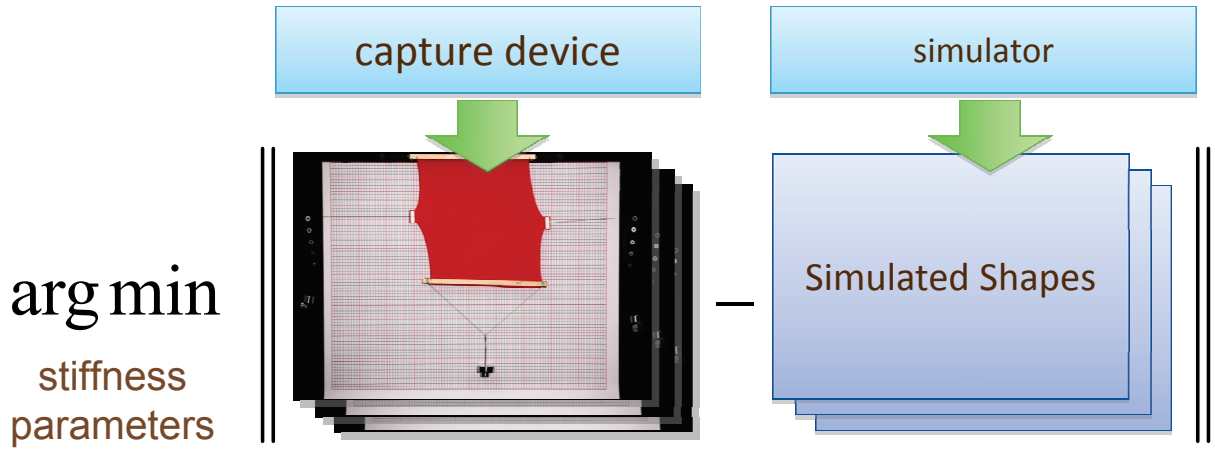
Average error:
2.58mm



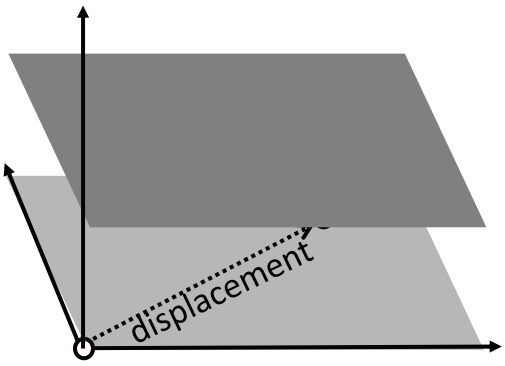
Data Capture: Bending



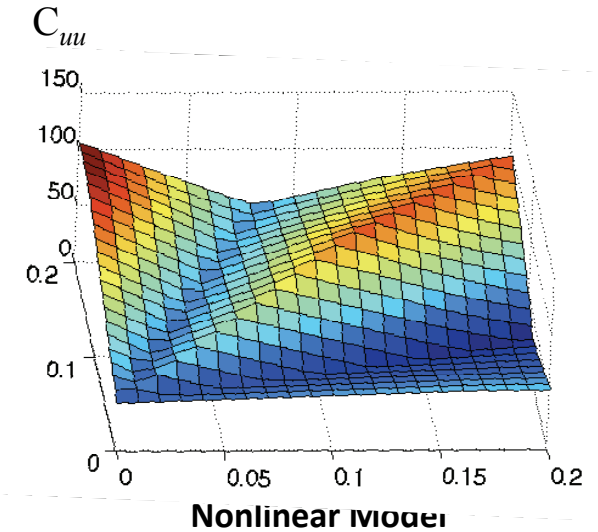
Fitting & Results: Optimization



Fitting & Results: Nonlinear Model



Linear Model



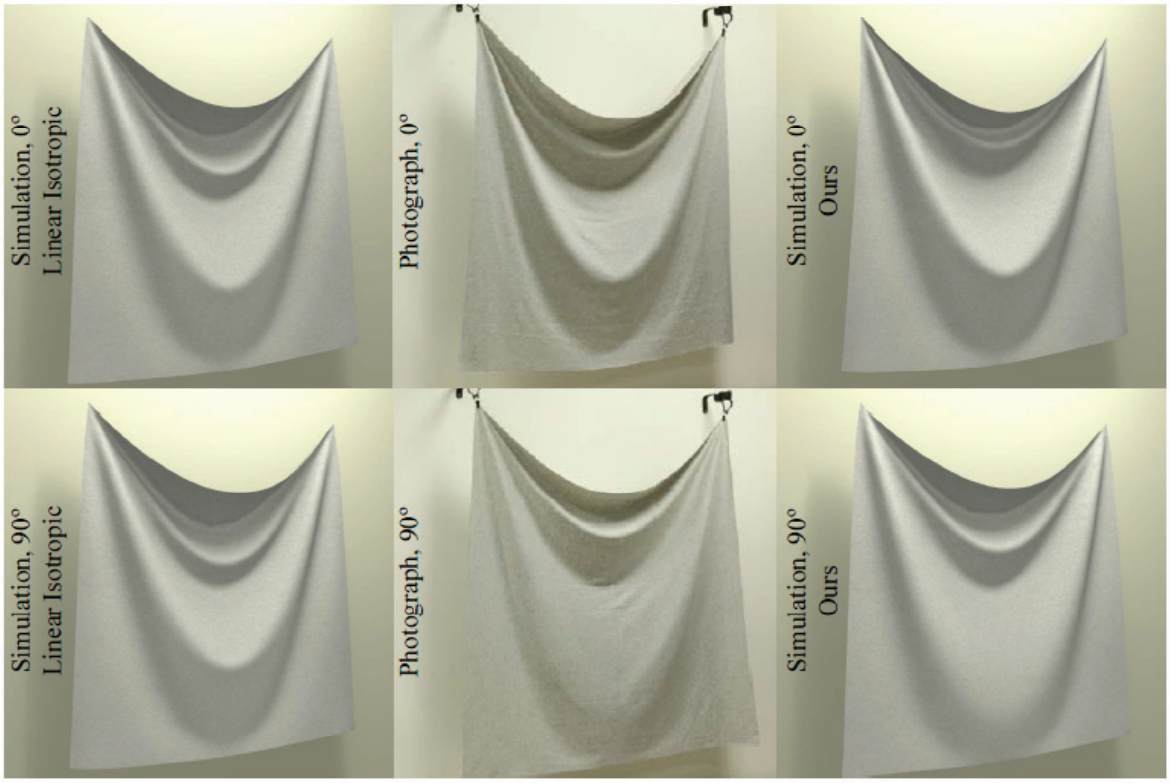
Nonlinear Model

Fitting & Results: Dataset

Color & Name	Ivory Rib Knit	Pink Ribbon Brown	White Dots on Black	Navy Sparkle Sweat	Camel Ponte Roma
Composition	95% Cotton 5% Spandex	100% Polyester	100% Polyester	96% Polyester 4% Spandex	60% Polyester 40% Rayon
Common Usage	Underwear	Blanket	Tablecloth	Sweater	Jacket
Density (kg/m^2)	0.276	0.228	0.128	0.224	0.284
Error (mm)	1.92	1.66	2.42	3.24	1.67
Color & Name	Gray Interlock	11oz Black Denim	White Swim Solid	Tango Red Jet Set	Royal target
Composition	60% Cotton 40% Polyester	99% Cotton 1% Spandex	87% Nylon 13% Spandex	100% Polyester	65% Cotton 35% Polyester
Common Usage	T-shirt	Jeans	Swimsuit	Fashion dress	Pants
Density (kg/m^2)	0.187	0.324	0.204	0.113	0.220
Error (mm)	2.58	2.30	1.57	2.06	0.89

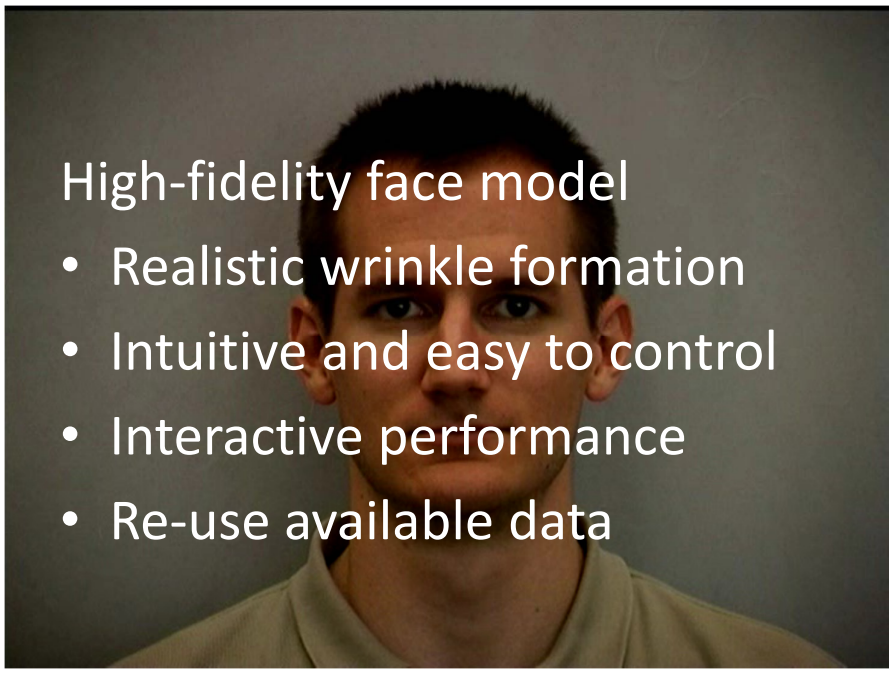
http://graphics.berkeley.edu/papers/Wang-DDE-2011-08/index.html

Fitting & Results: Comparison



6 Animation of Faces with Data-Driven Wrinkles

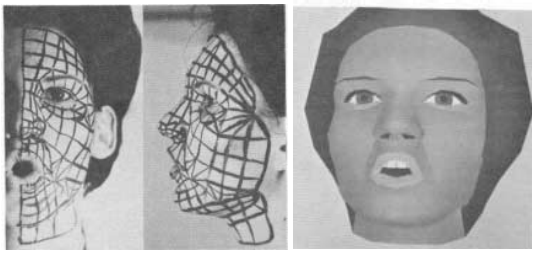
Goals



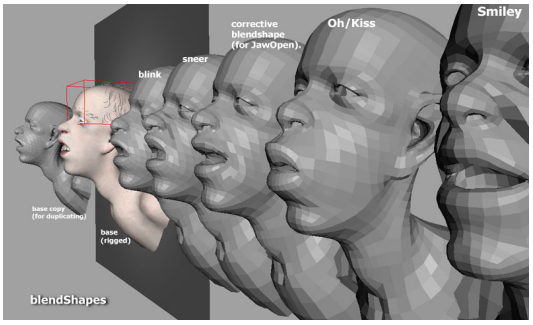
Hybrid Face Model

Related Work

- Blend shapes
  - [Parke 72, 74]
  - Automatic blending [Pighin 98]
  - Automatic segmentation [Joshi 03]
  - Morphable models [Blanz and Vetter 99]

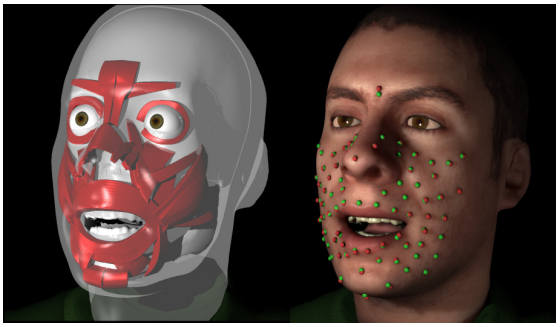


[Parke 72]



Related Work

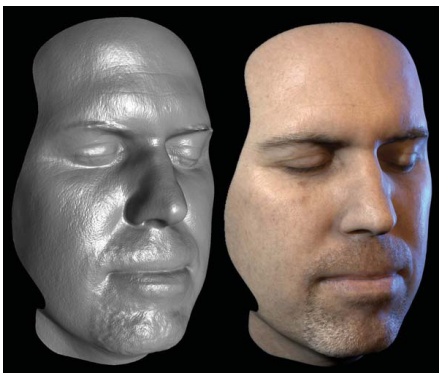
- Blend shapes
- Anatomical models
  - [Koch et al. 96]
  - [Essa et al. 96]
  - [Sifakis et al. 05]



[Sifakis et al. 05]

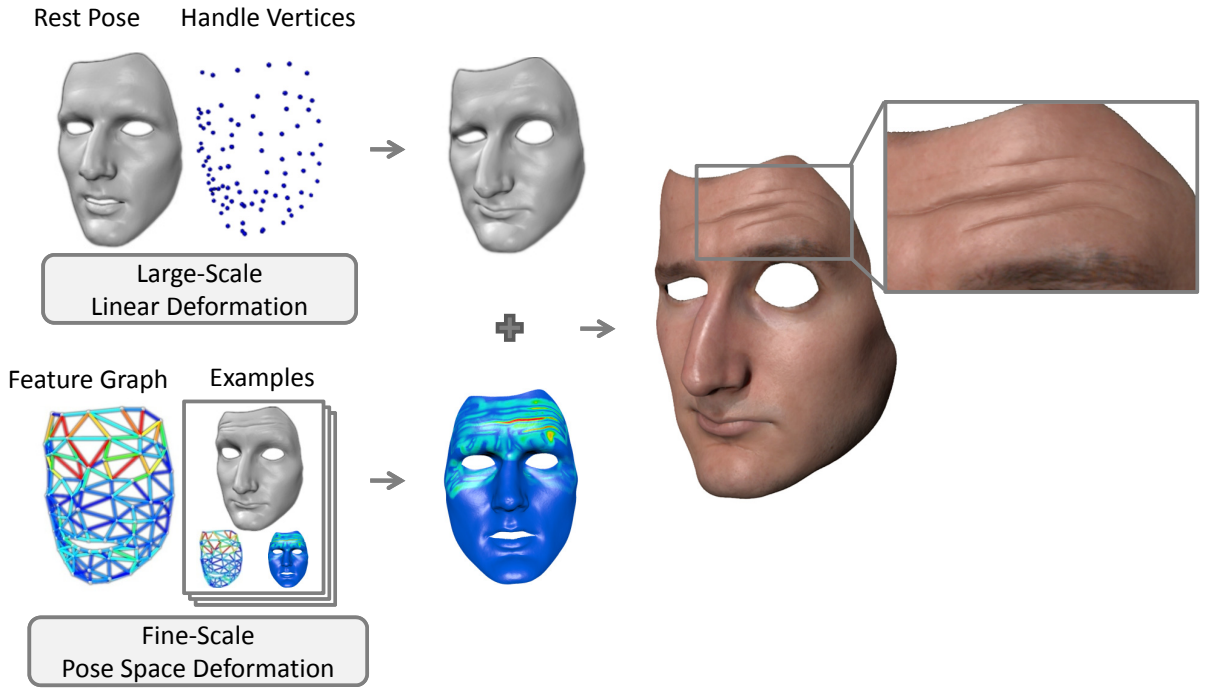
Related Work

- Blend shapes
- Anatomical models
- Multi-Scale Capture of Facial Geometry and Motion
  - [Bickel et al. 07]
  - [Ma et al. 07]
  - [Beeler et al. 11]

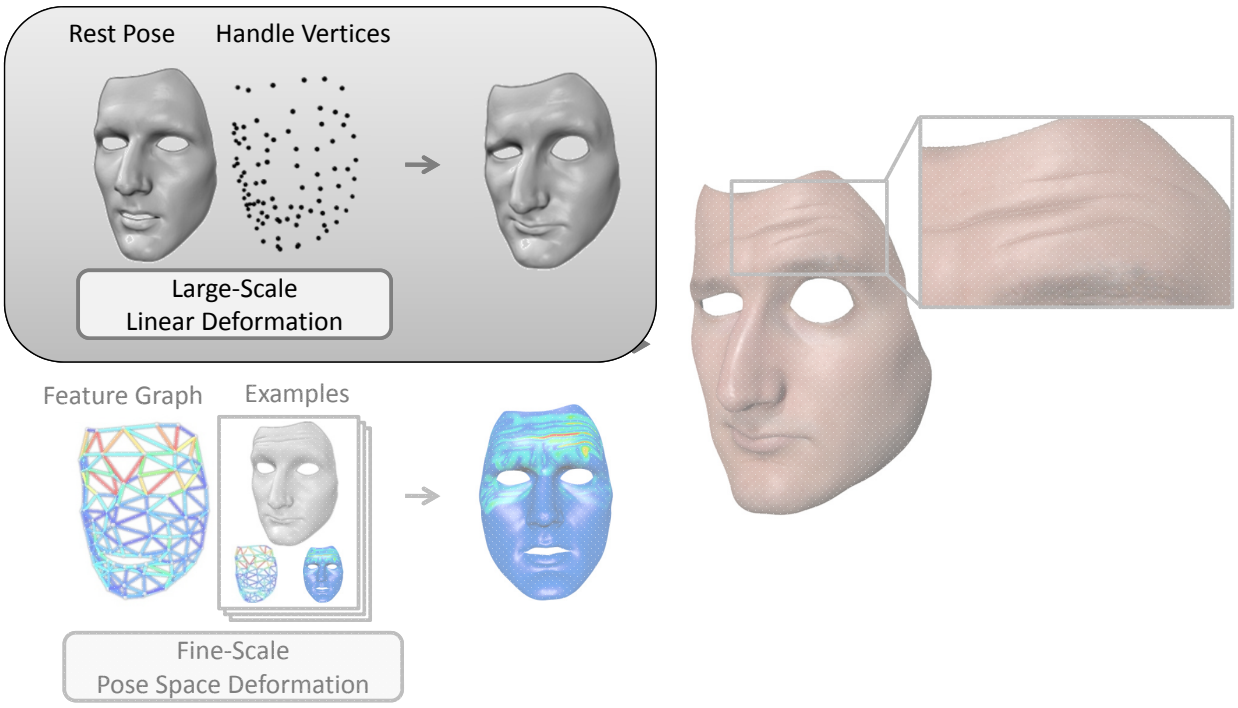


[Ma et al. 07]

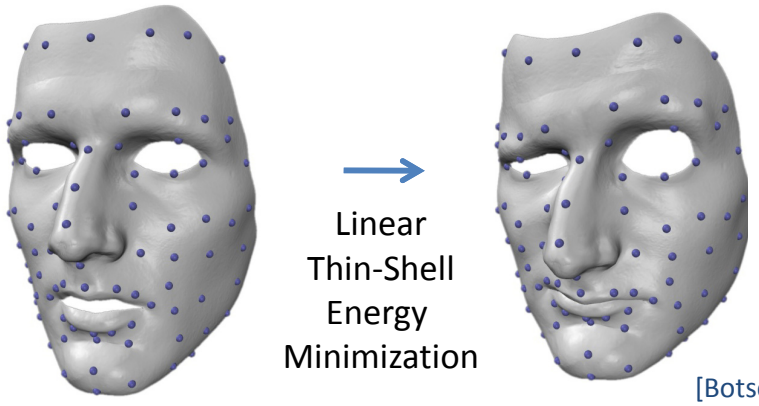
Hybrid Model



Overview



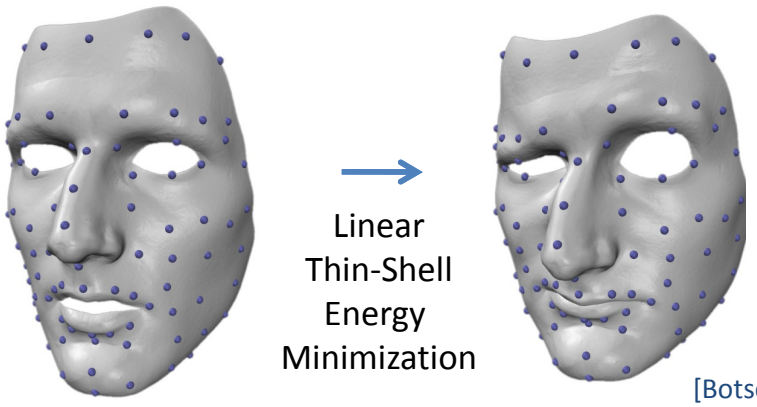
Large-Scale Deformation



[Botsch et al. 04;
Bickel et al. 07]

$$\int_F \underbrace{k_s \left(\left\| \frac{\partial \mathbf{d}}{\partial u} \right\|^2 + \left\| \frac{\partial \mathbf{d}}{\partial v} \right\|^2 \right)}_{\text{stretching}} + \underbrace{k_b \left(\left\| \frac{\partial^2 \mathbf{d}}{\partial u^2} \right\|^2 + 2 \left\| \frac{\partial^2 \mathbf{d}}{\partial u \partial v} \right\|^2 + \left\| \frac{\partial^2 \mathbf{d}}{\partial v^2} \right\|^2 \right)}_{\text{bending}} du\, dv$$

Large-Scale Deformation



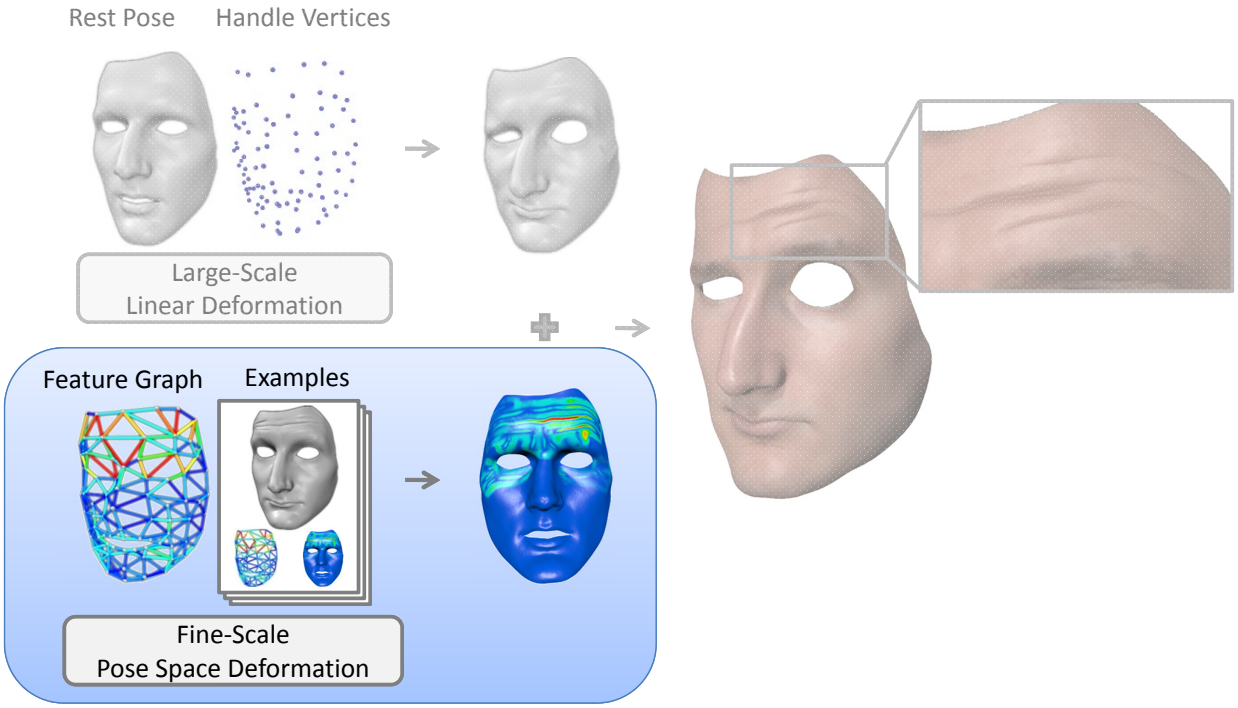
[Botsch et al. 04; Bickel et al. 07]

$$\begin{pmatrix} \mathbf{A} & \mathbf{A}_H \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \mathbf{u}_H \end{pmatrix} = \mathbf{0}$$

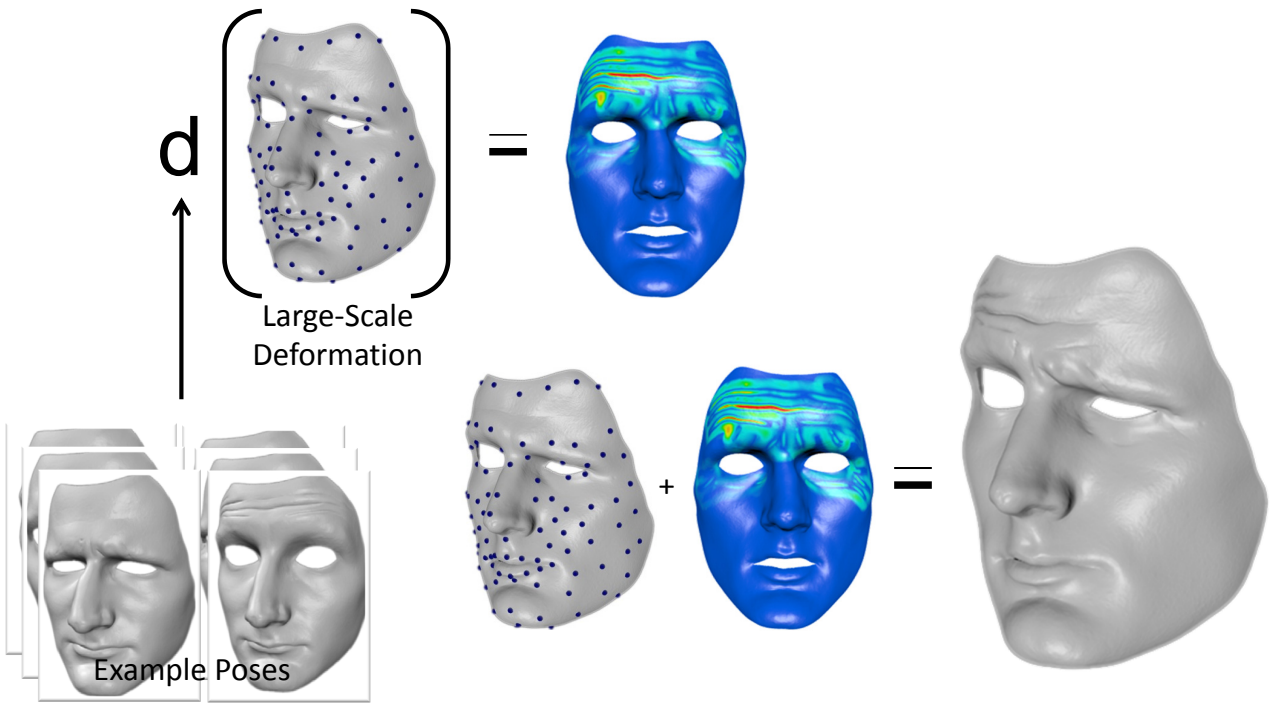
$$\mathbf{u} = \underbrace{-\mathbf{A}^{-1}\mathbf{A}_H}_{\mathbf{B}} \cdot \mathbf{u}_H$$

Large-Scale Deformation

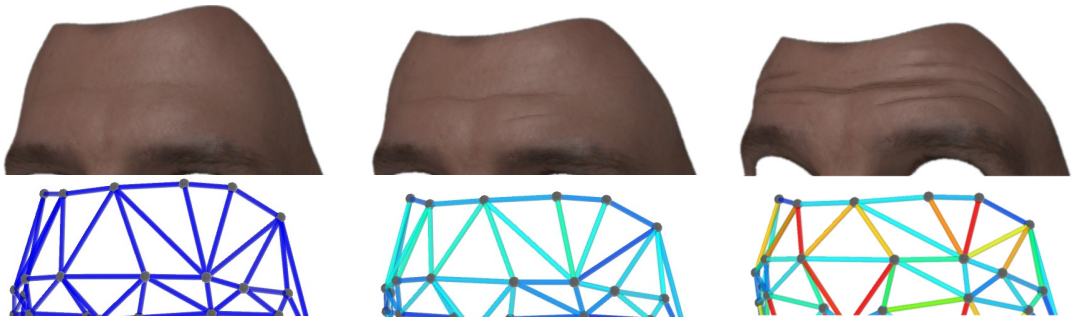
Overview



Fine-Scale Deformation



Fine-Scale Deformation



- Correlation wrinkling and lateral compression [Wu et al. 96]
- Feature graph defines F-dimensional feature vector

$$f = [f_1, \ldots, f_F]^T$$

$$f_i = \left(\|\mathbf{p}_{i,1} - \mathbf{p}_{i,2}\| - l_i\right) / l_i$$

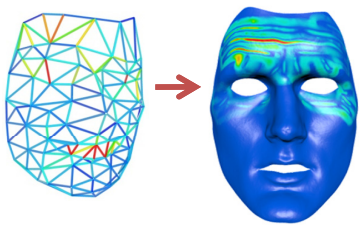
Fine-Scale Deformation

- Pose-Space Deformation [Lewis et al. 00]
- Mapping facial pose to 3D fine-scale displacements

$$\mathbf{d} : \mathbb{R}^F \rightarrow \mathbb{R}^{3V}$$

$$\mathbf{d}(\mathbf{f}) = \sum_{j=1}^{P} \mathbf{w}_j \cdot \varphi\left(\|\mathbf{f} - \mathbf{f}_j\|\right)$$

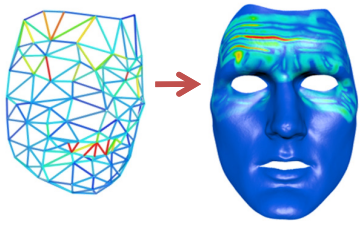
biharmonic RBF kernel $\varphi(r) = r$

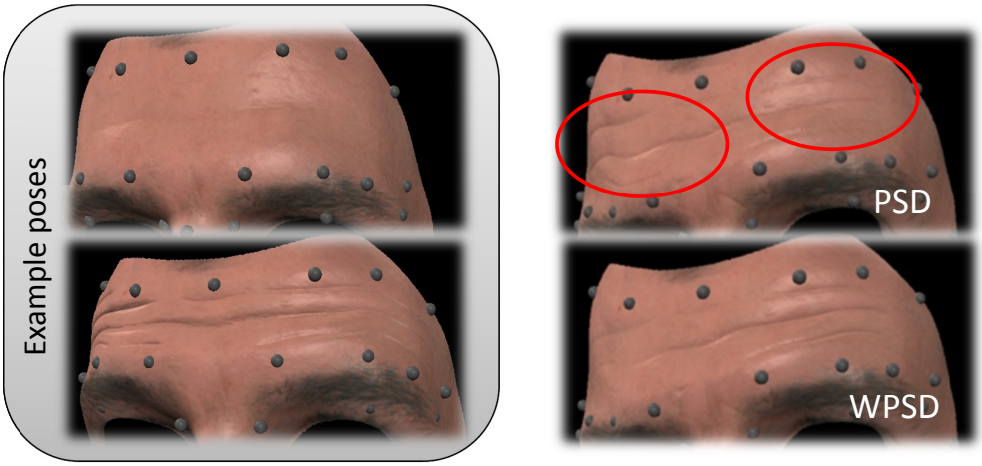


Fine-Scale Deformation

Problem: Exponential growth of example poses

$$\mathbf{d}(\mathbf{f}) = \sum_{j=1}^{P} \mathbf{w}_j \cdot \varphi(\|\mathbf{f} - \mathbf{f}_j\|)$$



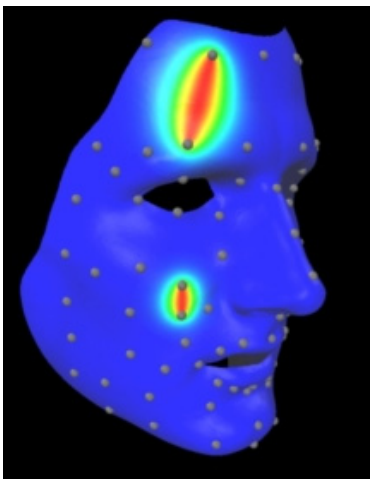


Fine-Scale Deformation

- Weighted Pose-Space Deformation

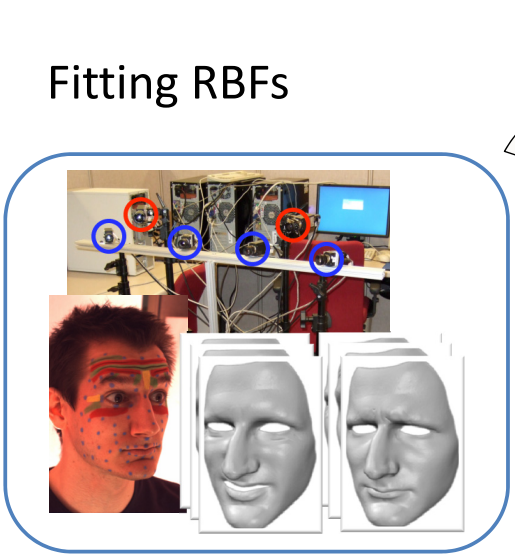
$$\|\mathbf{f} - \mathbf{f}_j\|_v = \left(\sum_{i=1}^{F} \alpha_{v,i} (f_i - f_{j,i})^2 \right)^{1/2}$$

$$\mathbf{d}_v(\mathbf{f}_i) = \sum_{j=1}^{P} \mathbf{w}_{v,j} \cdot \varphi\left(\|\mathbf{f} - \mathbf{f}_j\|_v\right)$$

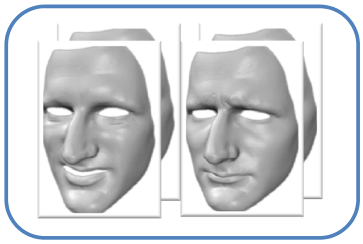


Fine-Scale Deformation

Fitting RBFs



Large training set T

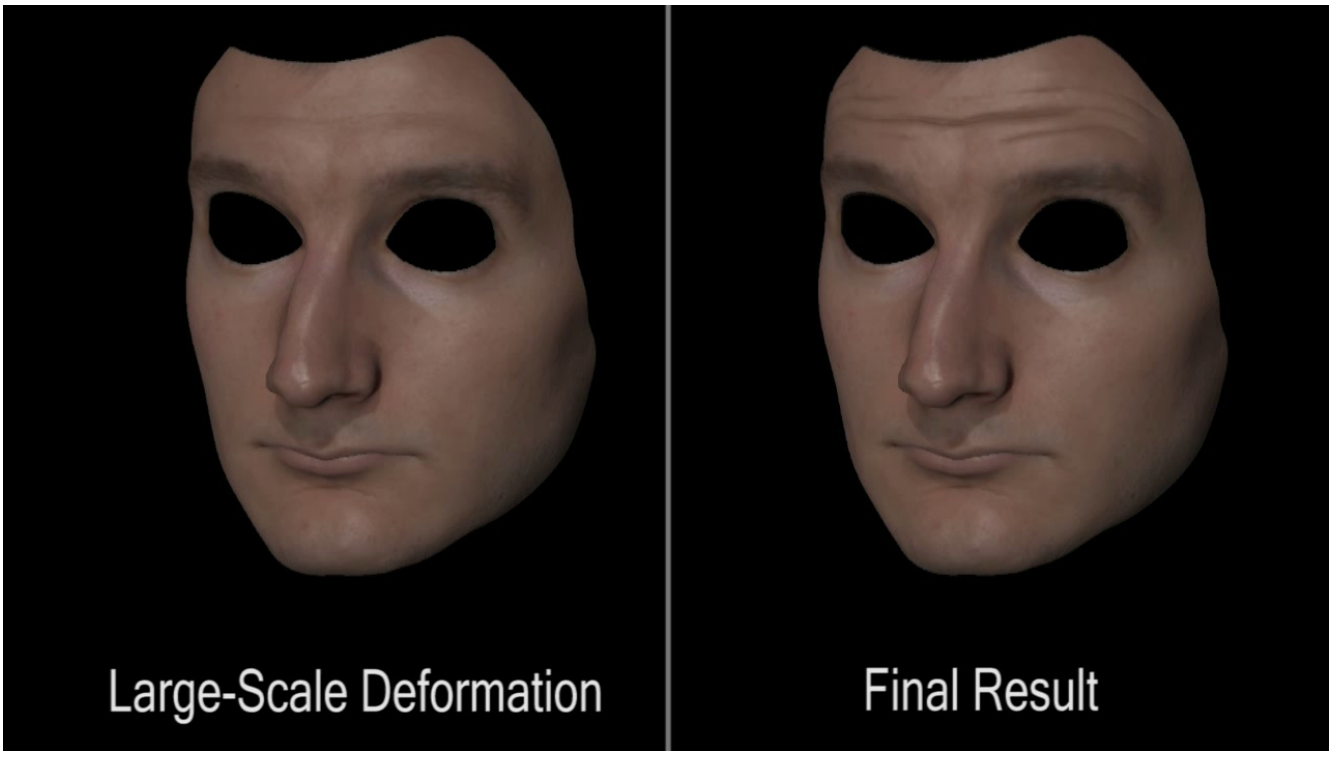


Compact basis P

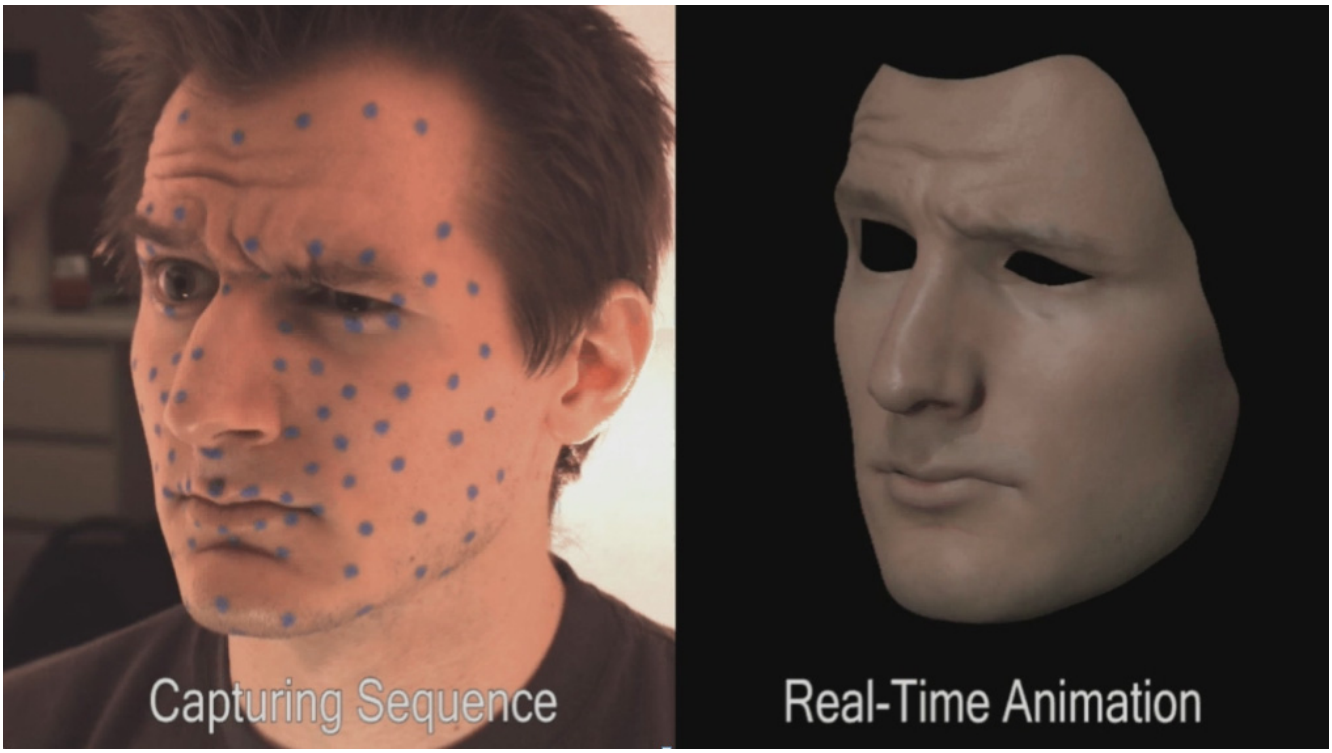
$$\begin{pmatrix} \varphi_{1,1} & \varphi_{1,2} & \cdots & \varphi_{1,P} \\ \varphi_{2,1} & \varphi_{2,2} & \cdots & \varphi_{2,P} \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_{T,1} & \varphi_{T,2} & \cdots & \varphi_{T,P} \end{pmatrix} \cdot \begin{pmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \\ \vdots \\ \mathbf{w}_P \end{pmatrix} = \begin{pmatrix} \mathbf{d}(\mathbf{f}_1) \\ \mathbf{d}(\mathbf{f}_2) \\ \vdots \\ \mathbf{d}(\mathbf{f}_T) \end{pmatrix}$$

Least-squares fit
(solve P×P system per vertex)

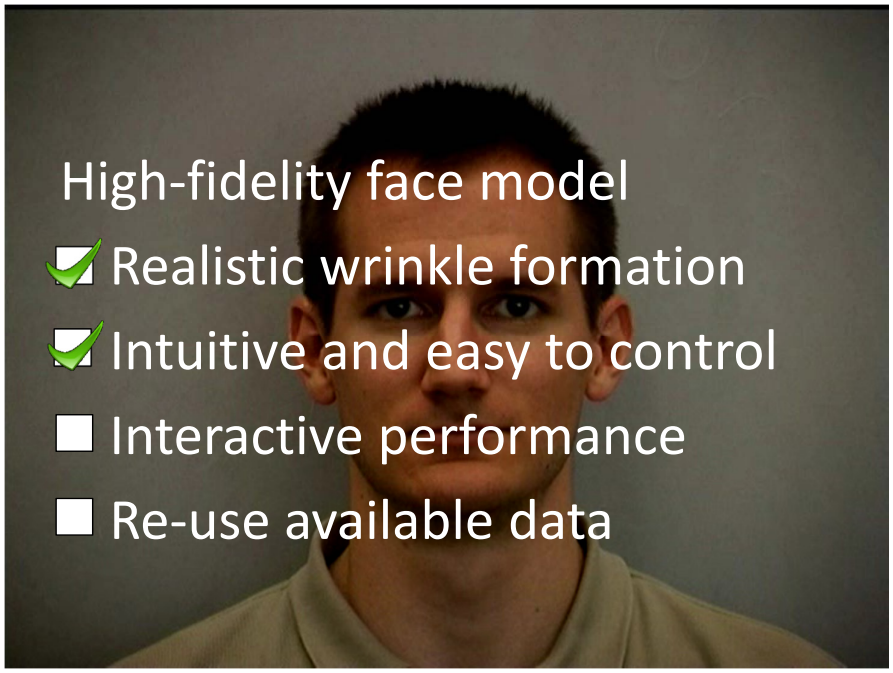
Fine-Scale Deformation



Results



Goals



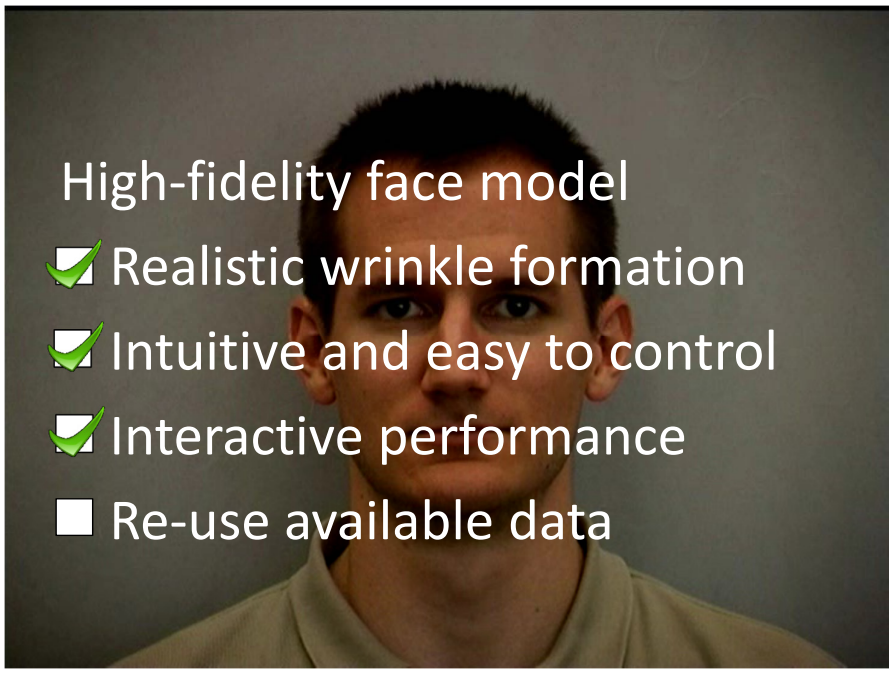
Performance

Pipeline-Step	Animation
Large-Scale	9 ms
Small-Scale	18 ms
Standard Renderer	4 ms
Skin Renderer [D'Eon et al. 07]	36 ms
Total Skin Renderer	**15 fps**
Total Standard Renderer	**30 fps**

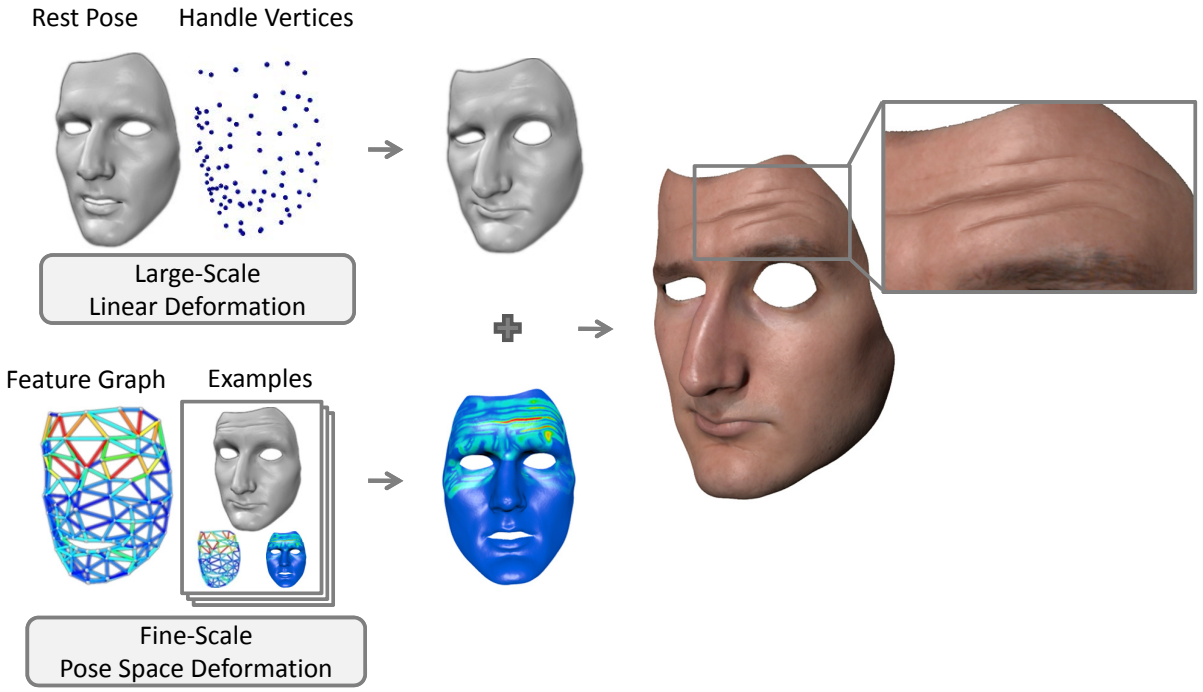
- NVidia 8800 GTX, CUDA
- 530k vertices
- 6 example poses
- 89 handle vertices, 243 feature edges

Results

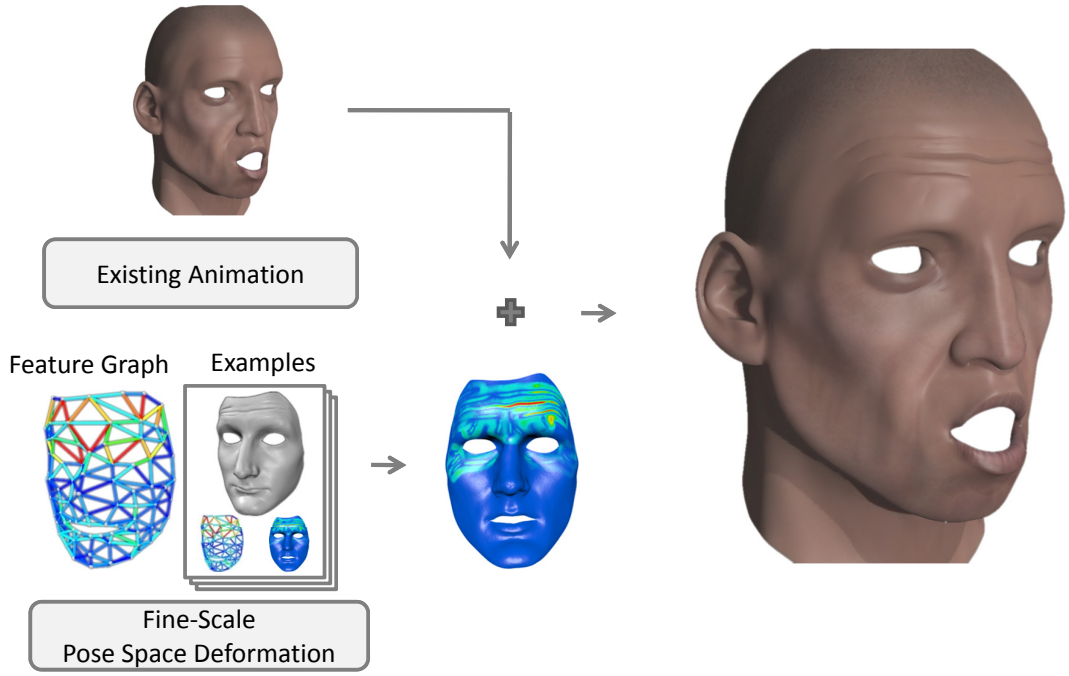
Goals



Overview



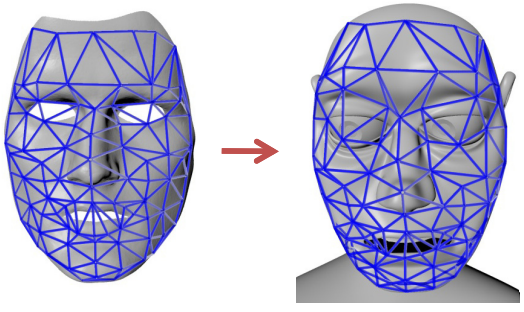
Overview



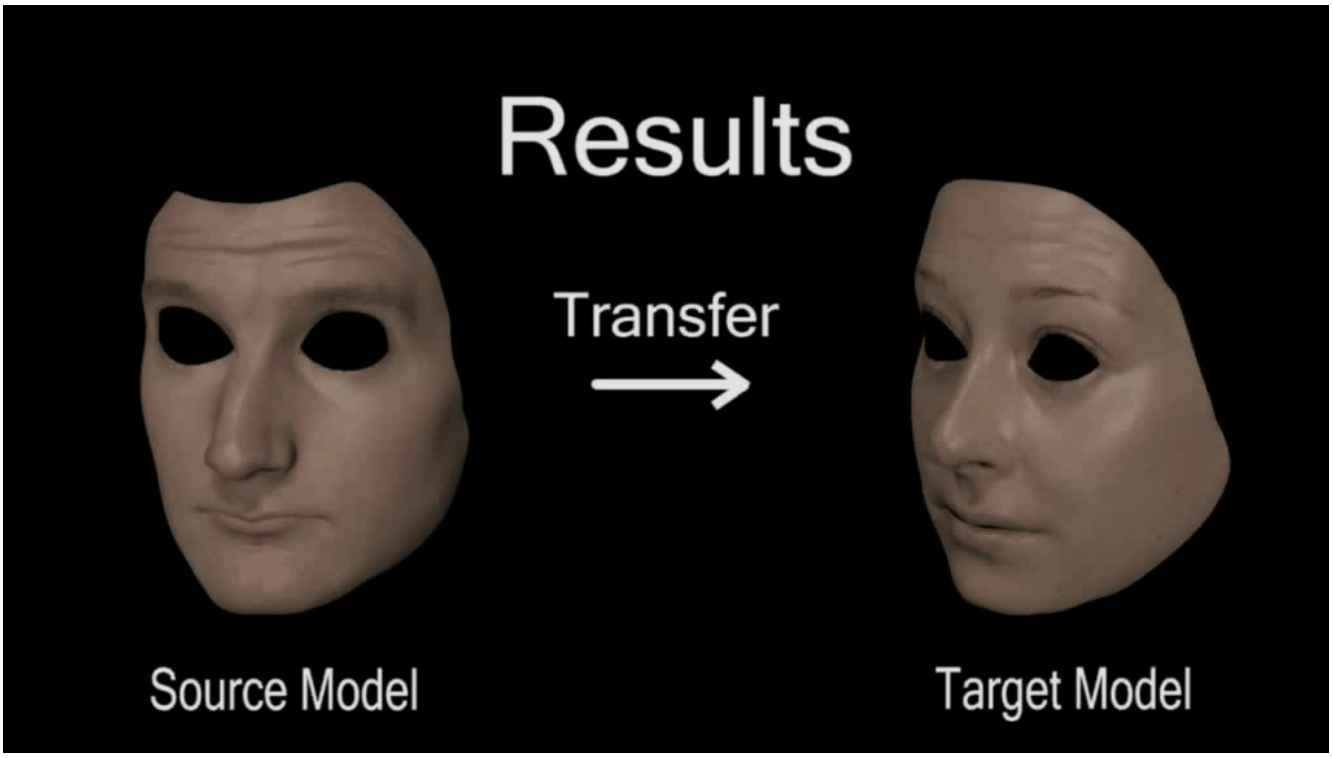
Transfer

- Visually enhance face animations
- Re-use available fine-scale data

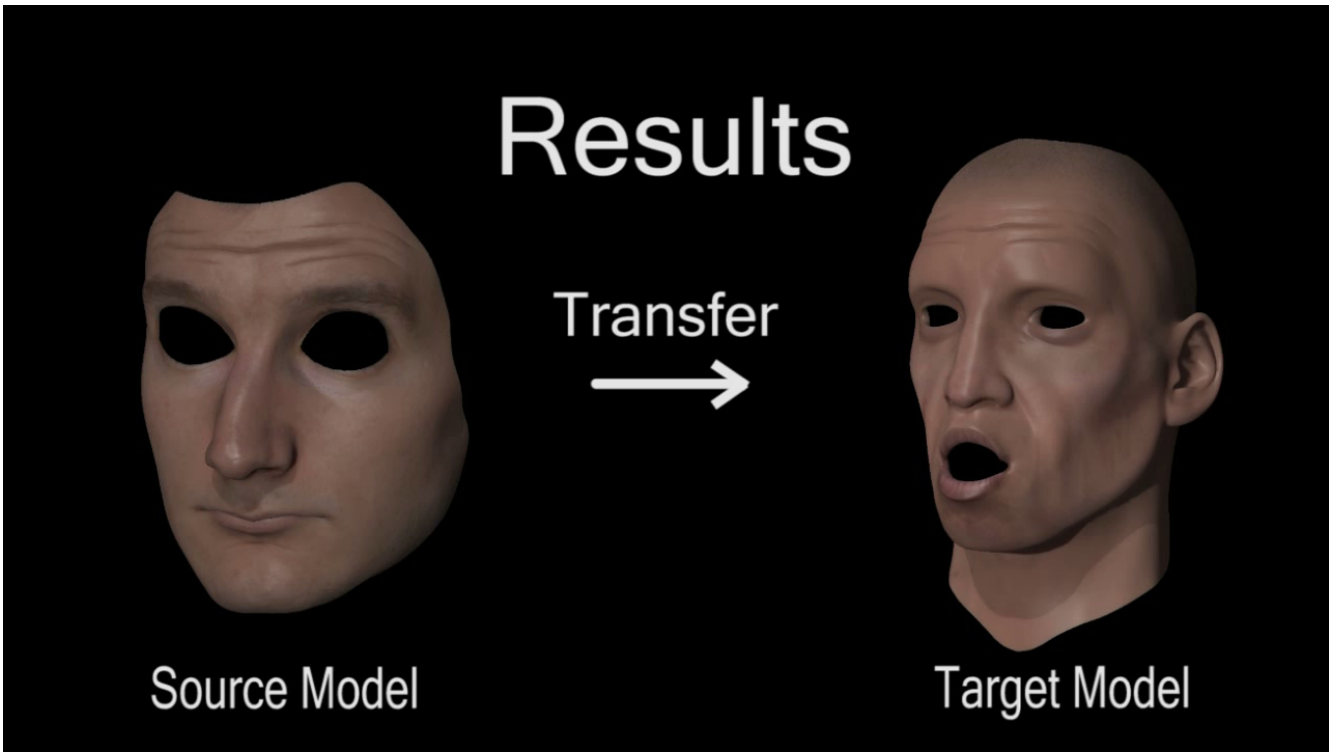
1. Establish correspondence
2. Transfer feature graph and fine-scale data
3. Evaluate
   - Strain
   - Fine-scale displacements



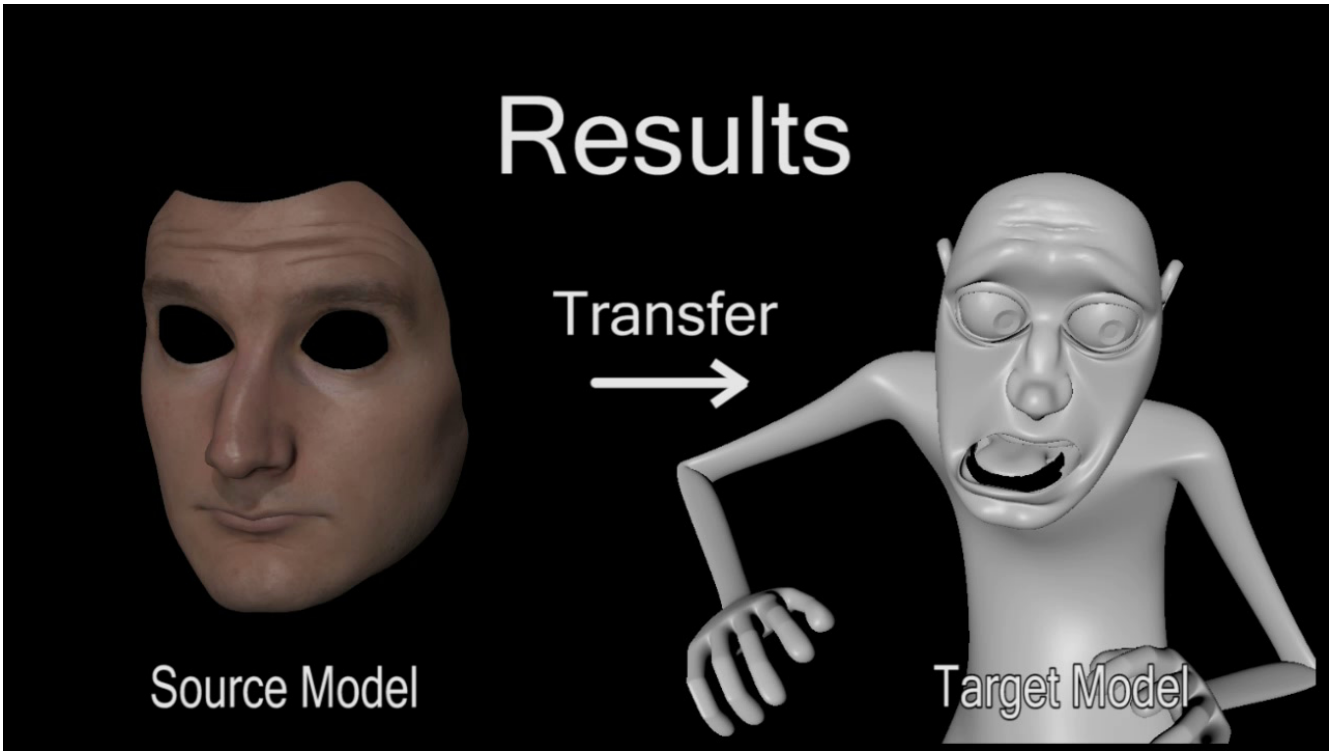
Results



Results



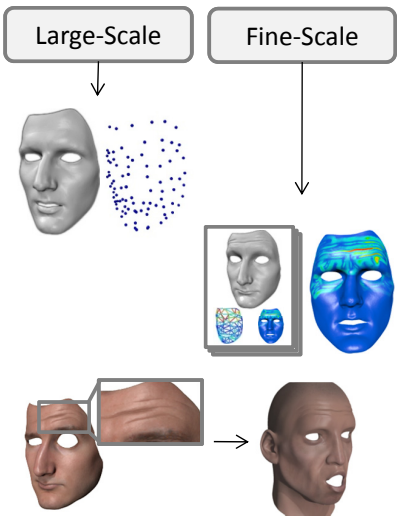
Results



Summary

Hybrid face animation method

- Constrained deformation approach
  - Driven by handles
- Example-based deformation approach
  - Strain-based
- Benefits
  - Complex wrinkling effects
  - Intuitive and easy control
  - Interactive performance
  - Transfer of facial details

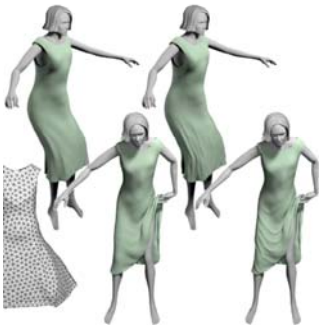


7 Clothing Animation with Wrinkle Synthesis from Examples

Related Work: Cloth Wrinkles

Cutler 2005

Kim 2008



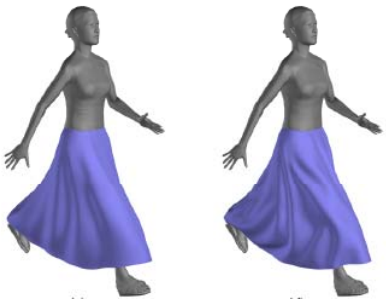
Rohmer 2010

Müller 2010

Wang 2010

Feng 2010

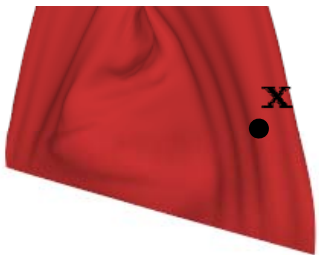
Kavan 2011



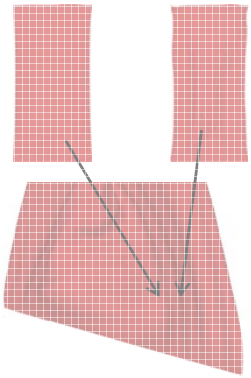
Non-Skinned Cloth

- Loose cloth, not a function of a character's pose
  - Skirts
  - Dresses
  - Coats
  - Flags
  - Sails
  - Curtains
  - …

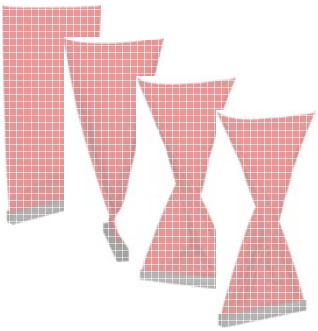
Outline



Cloth representation

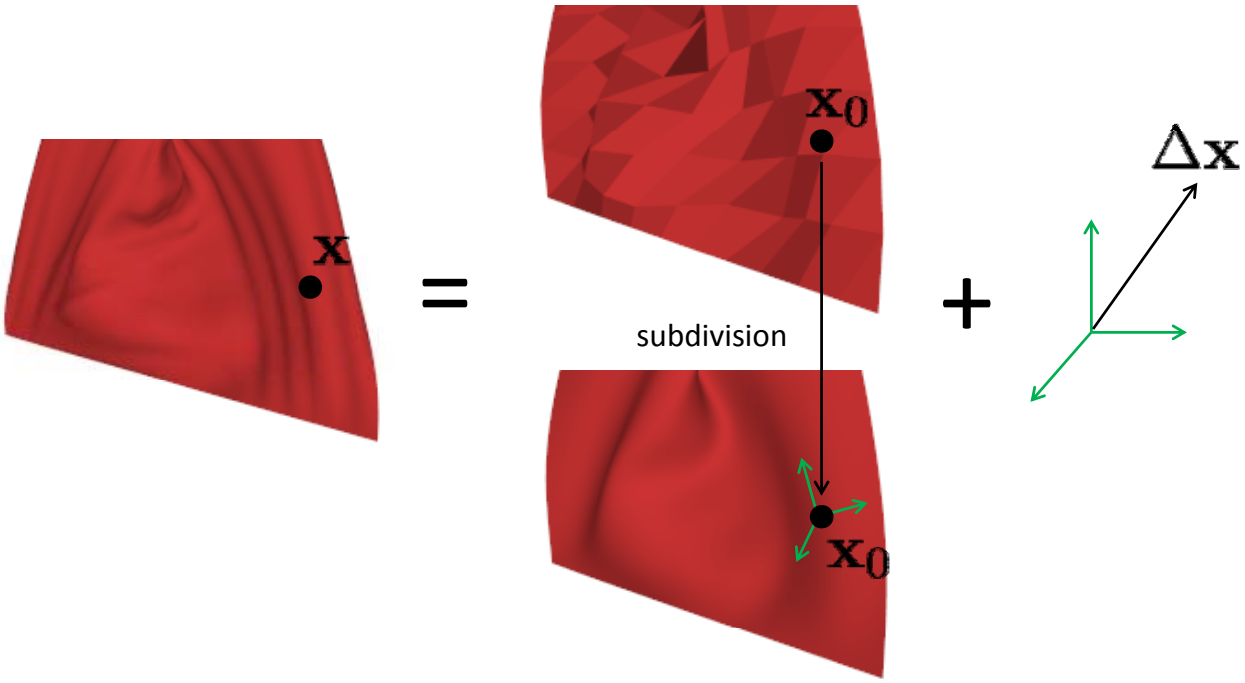


Example-based wrinkles



Example generation

Cloth Representation

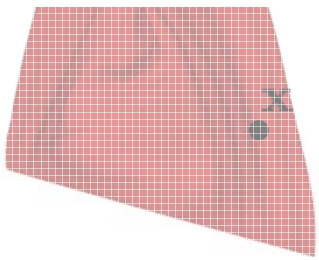


Coarse (400) Subdivided 3x (25600) Detailed (25600)

Low-Res Cloth

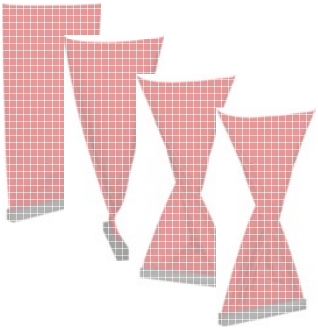
- Dynamics
  - Unlike skinned cloth.
- Collision handling
  - With a tolerance for wrinkles

Outline



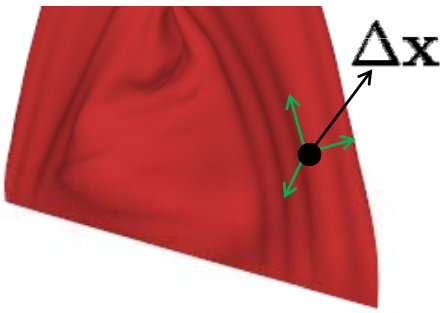
Cloth representation

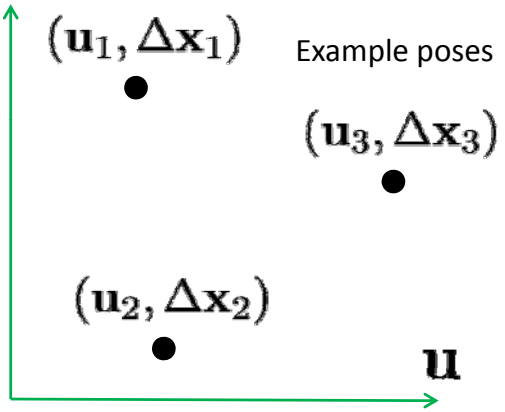
Example-based wrinkles



Example generation

Wrinkle Interpolation Model



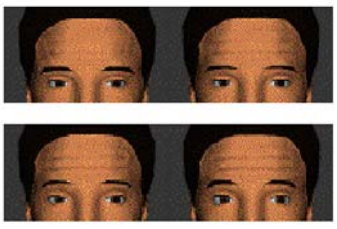


$$\Delta \mathbf{x} = \sum_i \alpha_i(\mathbf{u})\, \Delta \mathbf{x}_i$$

Pose weights

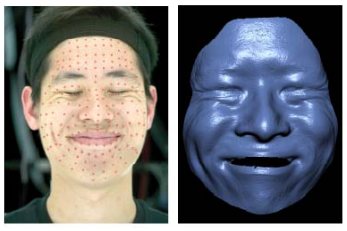
Interpolation Domain: Strain

- Define wrinkle detail as a function of low-res cloth deformation
- Done before for skin



Wu 1996

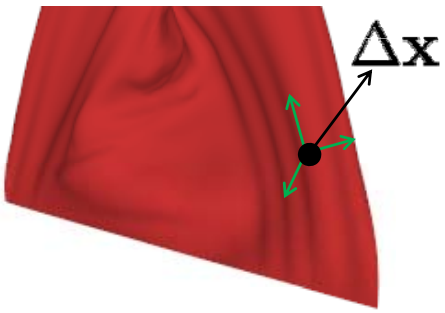
Bickel 2008

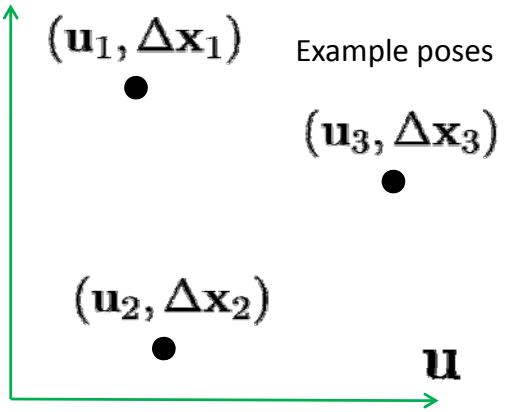


Ma 2008

- Rotation invariant strain metric
- Specifically, deformation of the edges of the low-res cloth [Bickel 2008]

Pose-Space Deformation [Lewis 2000]

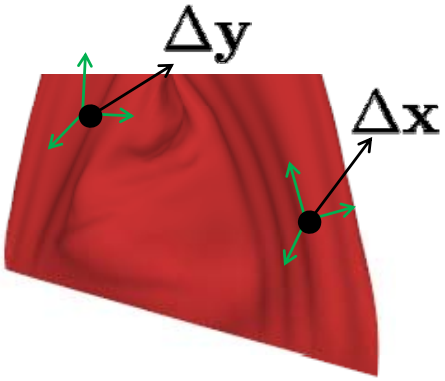




$$\Delta \mathbf{x} = \sum_i \alpha_i(\mathbf{u})\, \Delta \mathbf{x}_i$$

$$\alpha_i = \sum_j \alpha_{ij}\, \phi(\|\mathbf{u} - \mathbf{u}_j\|)$$ RBFs

Weighted PSD [Kurihara 2004]



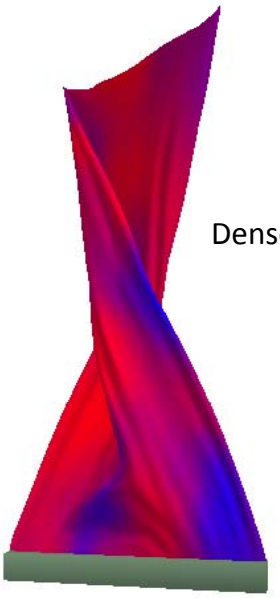
$$\Delta \mathbf{x} = \sum_i \alpha_i(\mathbf{u_x})\, \Delta \mathbf{x}_i$$

$$\Delta \mathbf{y} = \sum_i \beta_i(\mathbf{u_y})\, \Delta \mathbf{x}_i$$

Local strain

Sparse WPSD

- Detail is high-res, low-res strain is not



Dense weights

Sparse weights (1/64)

CPU-GPU Implementation

- CPU
  - Low-res cloth dynamics
  - Evaluation of low-res pose weights
- GPU
  - Interpolation of pose weights
  - Computation and addition of details

Related Work

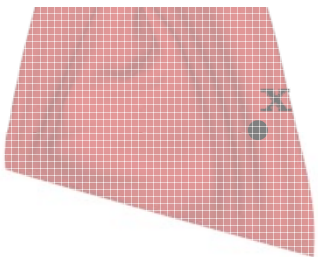
- [Kavan 2011]

  - Example-based linear model that extends subdivision

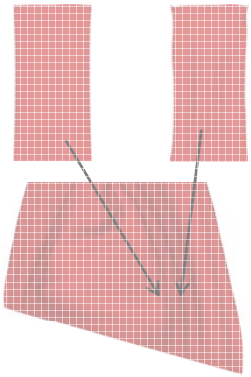
- [Seiler 2012]

  - Non-linear regression based on Gaussian kernels, in contact space

Outline



Cloth representation



Example-based wrinkles

Example generation

Synchronized Simulations

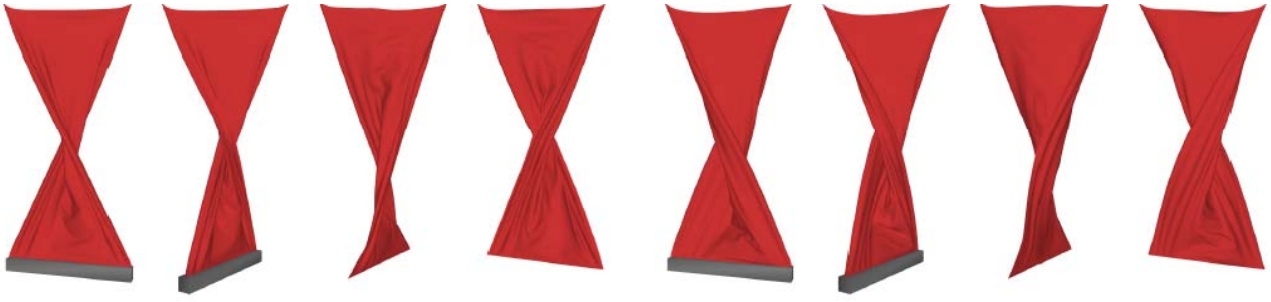
- TRACKS [Bergou 2007]

  - Constraints from low-res to high-res simulation
  - Simple thanks to subdivision
  - Collisions on both low-res and high-res simulations

  - Same as [Kavan 2011]

Coarse (167fps)

Data-driven (125ps)



Full sim (2.53fps)

Tracked (2.85fps)

Coarse

Data-driven

Conclusions

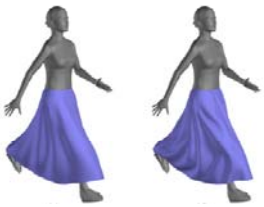
√ Fast low-res dynamics and contact
+ fast quasi-static wrinkles

√ Preserve wrinkle content

x Lack of high-res dynamics

Dynamics

- Travelling waves [Kavan 2011]



- Learning linear dynamics model [de Aguiar 2010]

8 Outlook

This course focuses on three applications of data-driven methods in computer graphics: cloth, tissue and face animation. Data-driven methods have also been successfully applied to character motion, and we expect that data-driven approaches will become popular in other areas of simulation in computer graphics too. To maximize their applicability, it will be crucial to understand which processes and properties can be modeled accurately with data-driven approaches. Effort should be devoted to developing general algorithmic and methodological procedures, together with a clear understanding of their limitations.

One of the major difficulties in data-driven methods is to find suitable descriptions of the simulated processes. Those descriptions have an impact on the smoothness and fairness of the output functions, which in turn affect the robustness and accuracy of interpolation and optimization methods. Multi-scale process decompositions appear particularly interesting in situations where fine-scale effects dominate the computational cost and the modeling complexity.

To conclude, the application of data-driven simulation methods in the computer graphics industry will depend largely on the access to accurate data. To this end, progress must be made along two paths. One is the creation of data libraries, both geometric and mechanical, that can be used by a large set of developers. The other is a combined progress of algorithms and capture systems, to enable fast and cheap synthesis of data-driven models.

References

ALEXA, M., GROSS, M., PAULY, M., PFISTER, H., STAMMINGER, M., AND ZWICKER, M. 2004. Point-based computer graphics. *ACM SIGGRAPH 2004 Course Notes*.

ATCHESON, B., IHRKE, I., HEIDRICH, W., TEVS, A., BRADLEY, D., MAGNOR, M., AND SEIDEL, H.-P. 2008. Time-resolved 3d capture of non-stationary gas flows. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia) 27*, 5, 132.

BAKER, S., SCHARSTEIN, D., LEWIS, J., ROTH, S., BLACK, M., AND SZELISKI, R. 2011. A database and evaluation methodology for optical flow. *International Journal of Computer Vision 92*, 1, 1–31.

BECKER, M., AND TESCHNER, M. 2007. Robust and efficient estimation of elasticity parameters using the linear finite element method. In *SimVis*, 15–28.

BEELER, T., BICKEL, B., SUMNER, R., BEARDSLEY, P., AND GROSS, M. 2010. High-quality single-shot capture of facial geometry. *ACM Trans. Graph. (Proc. SIGGRAPH)*.

BEELER, T., HAHN, F., BRADLEY, D., BICKEL, B., BEARDSLEY, P., GOTSMAN, C., SUMNER, R. W., AND GROSS, M. 2011. High-quality passive facial performance capture using anchor frames. *ACM Trans. Graph. 30*, 75:1–75:10.

BHAT, K. S., TWIGG, C. D., HODGINS, J. K., KHOSLA, P. K., POPOVIĆ, Z., AND SEITZ, S. M. 2003. Estimating cloth simulation parameters from video. In *Proc. ACM SIGGRAPH/Eurographics SCA*, 37–51.

BICKEL, B., LANG, M., BOTSCH, M., OTADUY, M. A., AND GROSS, M. 2008. Pose-space animation and transfer of facial details. In *Proc. of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, 57–66.

BICKEL, B., BÄCHER, M., OTADUY, M. A., MATUSIK, W., PFISTER, H., AND GROSS, M. 2009. Capture and modeling of non-linear heterogeneous soft tissue. *ACM Trans. Graph. 28*, 3 (July), 89:1–89:9.

BICKEL, B., BÄCHER, M., OTADUY, M. A., LEE, H. R., PFISTER, H., GROSS, M., AND MATUSIK, W. 2010. Design and fabrication of materials with desired deformation behavior. *ACM Transactions on Graphics 29*, 4 (July), 63:1–63:10.

BOUBEKEUR, T., REUTER, P., AND SCHLICK, C. 2005. Visualization of point-based surfaces with locally reconstructed subdivision surfaces. In *Shape Modeling International*.

BOUBEKEUR, T., HEIDRICH, W., GRANIER, X., AND SCHLICK, C. 2006. Volume-surface trees. *Computer Graphics Forum (Proceedings of EUROGRAPHICS 2006) 25*, 3, 399–406.

BRADLEY, D., AND HEIDRICH, W. 2010. Binocular camera calibration using rectification error. *IEEE Conference on Computer and Robot Vision (CRV)*.

BRADLEY, D., BOUBEKEUR, T., AND HEIDRICH, W. 2008. Accurate multi-view reconstruction using robust binocular stereo and surface meshing. In *Proc. CVPR*.

BRADLEY, D., POPA, T., SHEFFER, A., HEIDRICH, W., AND BOUBEKEUR, T. 2008. Markerless garment capture. *ACM Trans. Graph. (Proc. of SIGGRAPH) 27*, 3, 99:1–99:9.

BRADLEY, D., ATCHESON, B., IHRKE, I., AND HEIDRICH, W. 2009. Synchronization and rolling shutter compensation for consumer video camera arrays. In *International Workshop on Projector-Camera Systems (PROCAMS 2009)*.

BRADLEY, D., HEIDRICH, W., POPA, T., AND SHEFFER, A. 2010. High resolution passive facial performance capture. *ACM Trans. Graph. (Proc. SIGGRAPH) 29*, 4, 41:1–41:10.

BREEN, D., HOUSE, D., AND WOZNY, M. 1994. Predicting the drape of woven cloth using interacting particles. In *Proc. of ACM SIGGRAPH*, 365–372.

DE AGUIAR, E., STOLL, C., THEOBALT, C., AHMED, N., SEIDEL, H.-P., AND THRUN, S. 2008. Performance capture from sparse multi-view video. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 98.

DE AGUIAR, E., SIGAL, L., TREUILLE, A., AND HODGINS, J. K. 2010. Stable spaces for real-time clothing. *ACM Transactions on Graphics 29*, 4 (July), 106:1–106:9.

EBERHARDT, B., WEBER, A., AND STRASSER, W. 1996. A fast, flexible, particle-system model for cloth draping. *IEEE Computer Graphics and Applications 16*, 5, 52–59.

EKMAN, P., AND FRIESEN, W. 1978. The facial action coding system: A technique for the measurement of facial movement. In *Consulting Psychologists*.

FENG, W.-W., YU, Y., AND KIM, B.-U. 2010. A deformation transformer for real-time cloth animation. *ACM Transactions on Graphics 29*, 4 (July), 108:1–108:9.

FURUKAWA, Y., AND PONCE, J. 2008. Dense 3d motion capture from synchronized video streams. In *Proc. CVPR*.

FURUKAWA, Y., AND PONCE, J. 2009. Dense 3d motion capture for human faces. In *CVPR*.

FURUKAWA, Y., AND PONCE, J. 2010. Accurate, dense, and robust multi-view stereopsis. *IEEE Trans. on Pattern Analysis and Machine Intelligence 32*, 8, 1362–1376.

FUSIELLO, A., TRUCCO, E., AND VERRI, A. 2000. A compact algorithm for rectification of stereo pairs. *Mach. Vision Appl. 12*, 1, 16–22.

GOPI, M., KRISHNAN, S., AND SILVA, C. 2000. Surface reconstruction based on lower dimensional localized delaunay triangulation. In *Eurographics*.

GROSS, M., AND PFISTER, H., Eds. 2007. *Point-Based Graphics*. Morgan Kaufmann Publishers.

GUENTER, B., GRIMM, C., WOOD, D., MALVAR, H., AND PIGHIN, F. 1998. Making faces. In *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, 55–66.

KAJBERG, J., AND LINDKVIST, G. 2004. Characterisation of materials subjected to large strains by inverse modelling based on in-plane displacement fields. *International Journal of Solids and Structures 41*, 13, 3439–3459.

KAUER, M., VUSKOVIC, V., DUAL, J., SZEKELY, G., AND BAJKA, M. 2002. Inverse finite element characterization of soft tissues. *Medical Image Analysis 6*, 3, 257–287.

KAVAN, L., GERSZEWSKI, D., BARGTEIL, A. W., AND SLOAN, P.-P. 2011. Physics-inspired upsampling for cloth simulation in games. *Proc. of ACM SIGGRAPH*.

KAZHDAN, M., BOLITHO, M., , AND HOPPE, H. 2006. Poisson surface reconstruction. In *Symposium on Geometry Processing*.

KUNITOMO, S., NAKAMURA, S., AND MORISHIMA, S. 2010. Optimization of cloth simulation parameters by considering static and dynamic features. In *ACM SIGGRAPH Posters*, 15:1.

LANG, J., PAI, D. K., AND WOODHAM, R. J. 2002. Acquisition of elastic models for interactive simulation. *International Journal of Robotics Research 21*, 8, 713–733.

LIN, I.-C., AND OUHYOUNG, M. 2005. Mirror mocap: Automatic and efficient capture of dense 3d facial motion parameters from video. *The Visual Computer 21*, 6, 355–372.

MA, W.-C., JONES, A., CHIANG, J.-Y., HAWKINS, T., FREDERIKSEN, S., PEERS, P., VUKOVIC, M., OUHYOUNG, M., AND DEBEVEC, P. 2008. Facial performance synthesis using deformation-driven polynomial displacement maps. *ACM Trans. Graphics (Proc. SIGGRAPH Asia) 27*, 5, 121.

MA, W.-C., JONES, A., CHIANG, J.-Y., HAWKINS, T., FREDERIKSEN, S., PEERS, P., VUKOVIC, M., OUHYOUNG, M., AND DEBEVEC, P. 2008. Facial performance synthesis using deformation-driven polynomial displacement maps. *ACM Trans. Graph. (Proc. of ACM SIGGRAPH Asia) 27*, 5.

MIGUEL, E., BRADLEY, D., THOMASZEWSKI, B., BICKEL, B., MATUSIK, W., OTADUY, M. A., AND MARSCHNER, S. 2012. Data-driven estimation of cloth simulation models. *Computer Graphics Forum (Proc. of Eurographics) 31*, 2.

MVIEW. http://vision.middlebury.edu/mview/.

Open source computer vision library.

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

SCHNUR, D. S., AND ZABARAS, N. 1992. An inverse method for determining elastic material properties and a material interface. *International Journal for Numerical Methods in Engineering 33*, 10, 2039–2057.

SCHOLZ, V., STICH, T., KECKEISEN, M., WACKER, M., AND MAGNOR, M. 2005. Garment motion capture using color-coded patterns. In *Proc. Eurographics*, 439–448.

SCHONER, J. L., LANG, J., AND SEIDEL, H.-P. 2004. Measurement-based interactive simulation of viscoelastic solids. *Computer Graphics Forum (Proc. Eurographics) 23*, 3, 547–556.

SEITZ, S. M., CURLESS, B., DIEBEL, J., SCHARSTEIN, D., AND SZELISKI, R. 2006. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *CVPR*.

SORKINE, O., COHEN-OR, D., LIPMAN, Y., ALEXA, M., RÖSSL, C., AND SEIDEL, H.-P. 2004. Laplacian surface editing. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, 175–184.

VLASIC, D., BARAN, I., MATUSIK, W., AND POPOVIĆ, J. 2008. Articulated mesh animation from multi-view silhouettes. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 97.

VOLINO, P., MAGNENAT-THALMANN, N., AND FAURE, F. 2009. A simple approach to nonlinear tensile stiffness for accurate cloth simulation. *ACM Trans. Graph. 28*, 4.

WAND, M., MITRA, N., PAULY, M., CHANG, W., AND LI, H. 2012. Dynamic geometry processing. In *Eurographics Tutorials*.

WANG, H., HECHT, F., RAMAMOORTHI, R., AND O'BRIEN, J. 2010. Example-based wrinkle synthesis for clothing animation. *ACM Transactions on Graphics 29*, 4 (July), 107:1–107:8.

WANG, H., RAMAMOORTHI, R., AND O'BRIEN, J. 2011. Data-driven elastic models for cloth: Modeling and measurement. *ACM Trans. Graph. (Proc. SIGGRAPH) 30*, 4, 71.

WHITE, R., CRANE, K., AND FORSYTH, D. 2007. Capturing and animating occluded cloth. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 34.

WILBURN, B., JOSHI, N., VAISH, V., LEVOY, M., AND HOROWITZ, M. 2004. High-speed videography using a dense camera array. In *Proc. of CVPR*, vol. 2, 294–301.

WILLIAMS, L. 1990. Performance-driven facial animation. *SIGGRAPH Comput. Graph. 24*, 4, 235–242.

ZHANG, Z. 1999. Flexible camera calibration by viewing a plane from unknown orientations. In *ICCV*, 666–673.

ZURDO, J. S., BRITO, J. P., AND OTADUY, M. A. 2013. Animating wrinkles by example on non-skinned cloth. *IEEE Trans. on Visualization and Computer Graphics 19*, 1.