EFFICIENT SIMULATION OF 3D ELASTIC SOLID AND ITS APPLICATIONS

Yin Yang, Assistant Professor
Electrical & Comp. Engineering Dept.
The University of New Mexico
April 2019

THE UNIVERSITY of NEW MEXICO

About Me

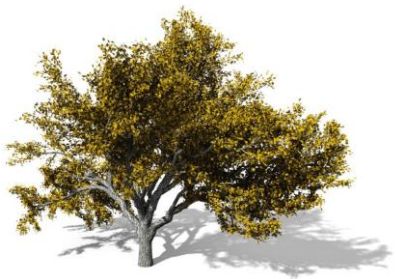
- Ph.D. 2013 from the University of Texas at Dallas
  - Awardee of David Daniel Fellowship Prize
  - Intern Researcher at Microsoft Research Asia, 2012
- Research interests
  - Interactive simulation and 3D modeling
  - Computer graphics
  - Machine learning for graphics and vision
  - Computational design & digital fabrication
  - Visualization
- Our research is generously supported by:

NSF · AIR FORCE RESEARCH LABORATORY · NVIDIA · … · THE UNIVERSITY of NEW MEXICO

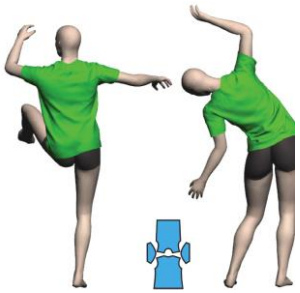
Deformable Models

- Accurate deformable simulation is important for realism
- FEM is the commonly-adopted approach
- It is known to be computational expensive
  - Partial/ordinary differential equations with a large number of DOFs
  - DOFs are two-way coupled
- Nonlinear models are more challenging as the system matrix varies

[Zheng & James, 2012]



[Zhao & Barbič, 2013]



[Xu et al, 2014]

[Teng et al, 2015]

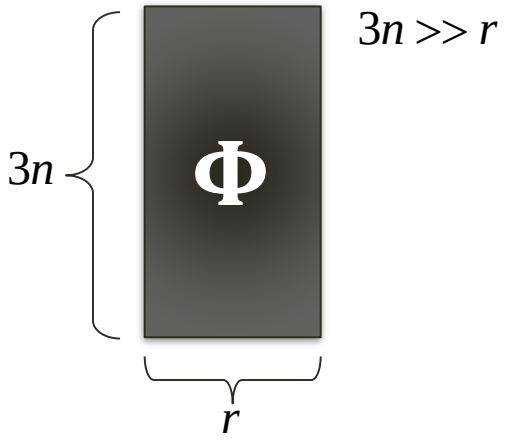
.....

Model Reduction

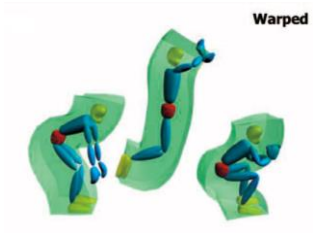
- A standard technique is called *model reduction*

$$\mathbf{M}\ddot{\mathbf{u}} = \mathbf{f}(\mathbf{u}, \dot{\mathbf{u}}, t) \qquad \mathbf{u} = \Phi\mathbf{q}$$

$$\mathbf{M}_q\ddot{\mathbf{q}} = \Phi^{\cdot}\,\mathbf{f}(\Phi\mathbf{q}, \Phi\dot{\mathbf{q}}, t),\ \mathbf{M}_q = \Phi^{\cdot}\,\mathbf{M}\Phi$$

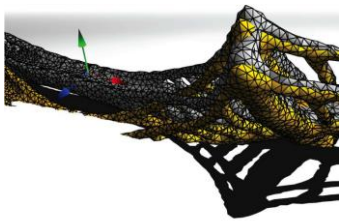


- Used in most existing interactive simulation techniques

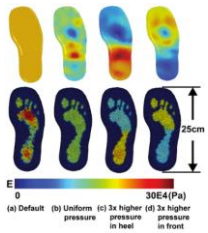


[Choi & Ko, 2005]


[Kim & James, 2011]


[Barbič et al, 2012]


[Harmon & Zorin, 2013]

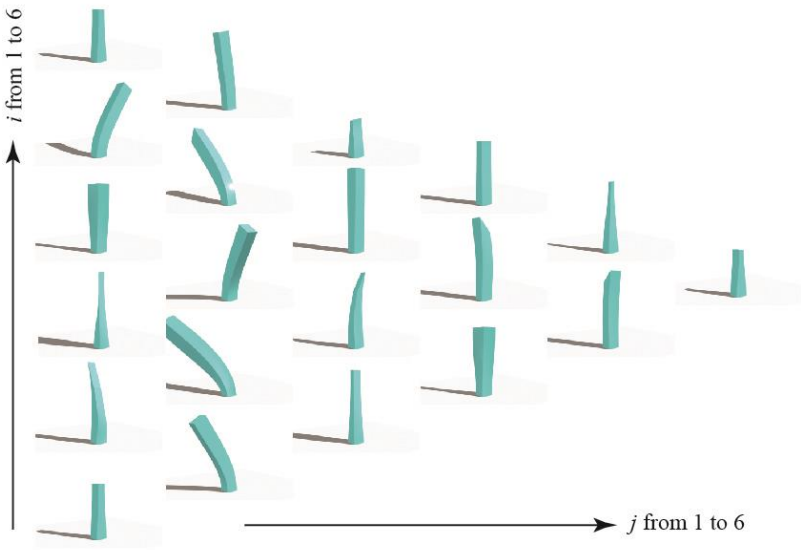


[Xu et al, 2015]

.....

Model Reduction (cont.)

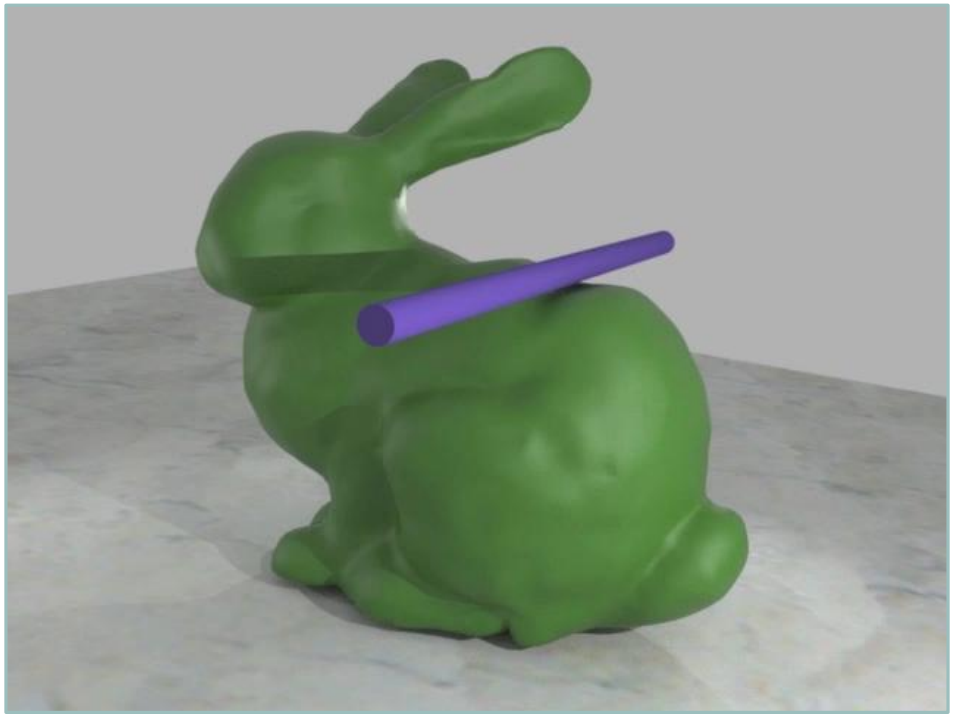
- How to compute modal matrix
  - Solve a large scale $3n \times 3n$ eigen problem
  - *Modal analysis*, similar to performing PCA over a dynamic system
  - We call each column in Φ a *mode*
- Physical property
  - High-frequency modes (bigger eigen values) are less likely to occur
  - Low-frequency modes represent dominant deformation



Eigen modes of a beam model

Model Reduction w/ Local Effects

- What if local deformation is favored
  - Adding more modes into the system (r becomes bigger)



[Hormon & Zorin, 2013]

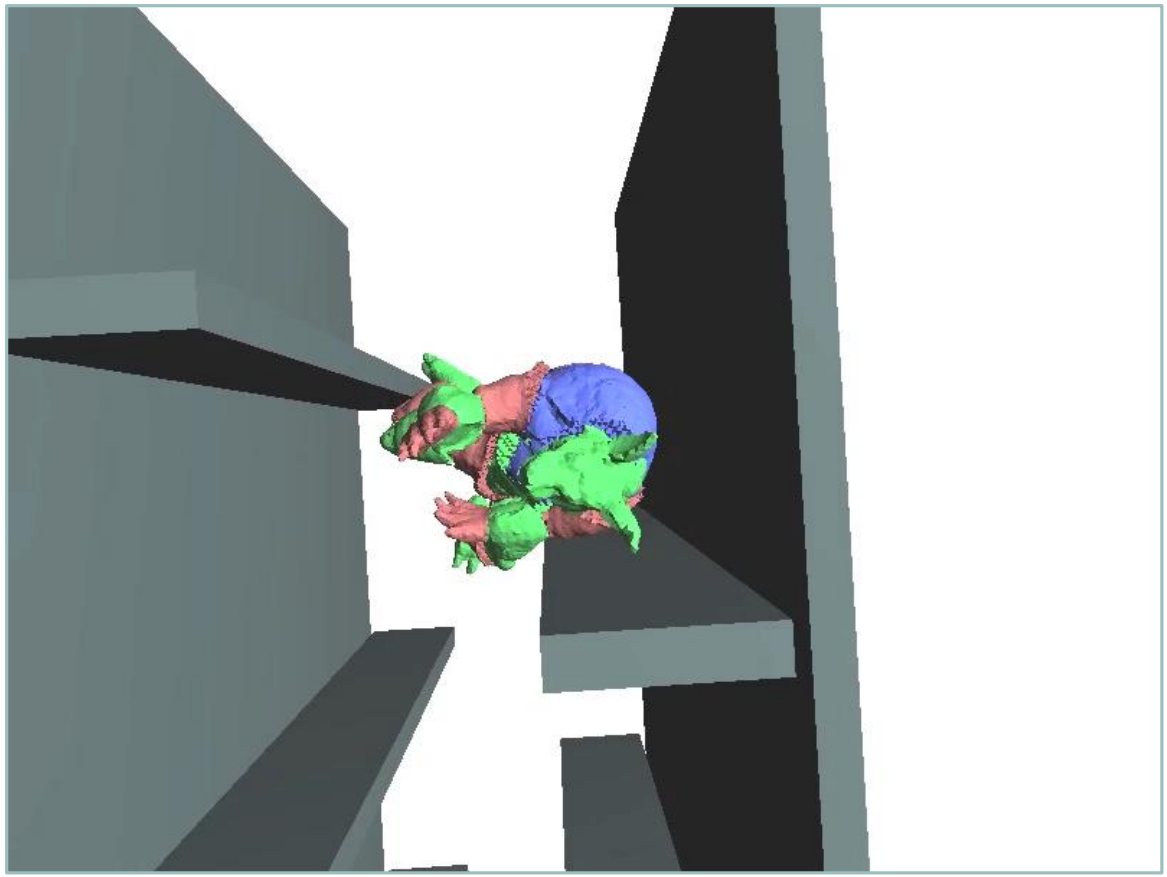
Domain Decomposition

- Subdivide the elastic volume into multiple *domains*
- Build mode locally at each domain

[Kim & James, 2011]

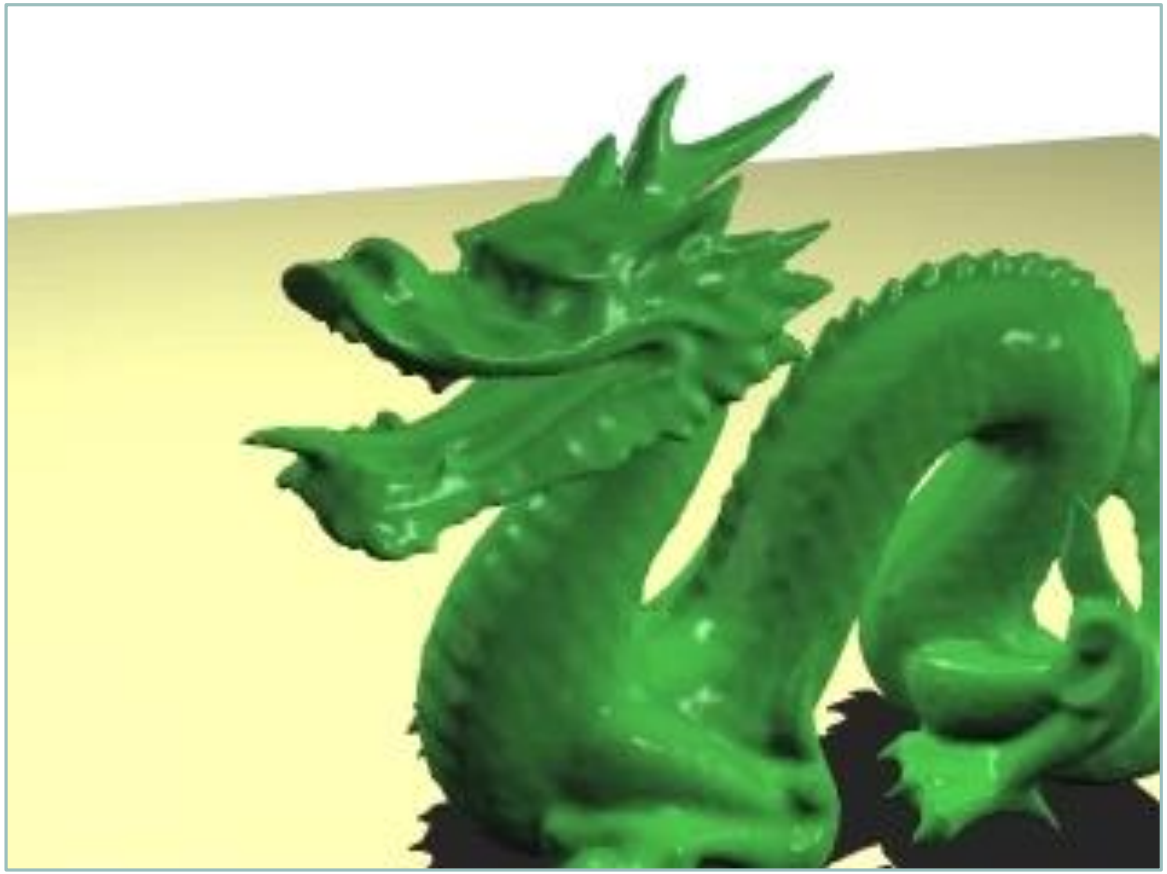
- We need to connect domains at their interfaces!

Domain Discontinuity (Example I)



[Kim & James, 2011]

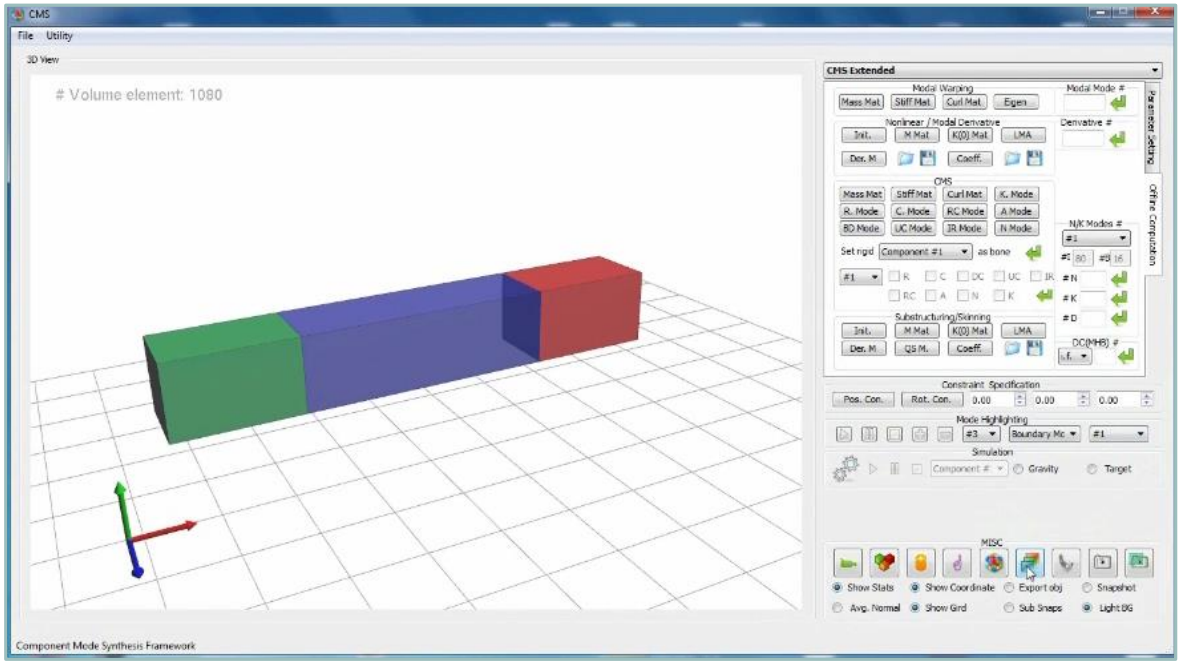
Domain Discontinuity (Example II)



[Barbič & Zhao, 2011]

Boundary-Driven Local Mode

- Computed as local equilibrium via boundary excitations
  - No locking artifacts and no cracks at interfaces



[Yang et al. 2013]

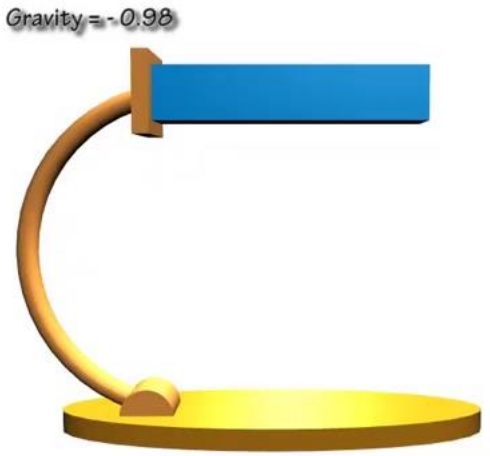
Boundary-Driven Local Mode

[Yang et al. 2013]

Incorporate Nonlinearity

- Local equilibrium is based on the linear stiffness matrix
  - Around the rest shape geometry of the model
  - Not able to handle rotational deformation
- Fundamental limitation for linear elasticity
  - Build subspace that captures nonlinear shapes

[Yang et al. 2015]



Krylov Block Subspace

Add a perturbation to rest shape stiffness

$$\mathbf{K}(\delta\mathbf{u}) = \mathbf{K} + \delta\mathbf{K},\ \mathbf{K} = \mathbf{K}(\mathbf{0})$$

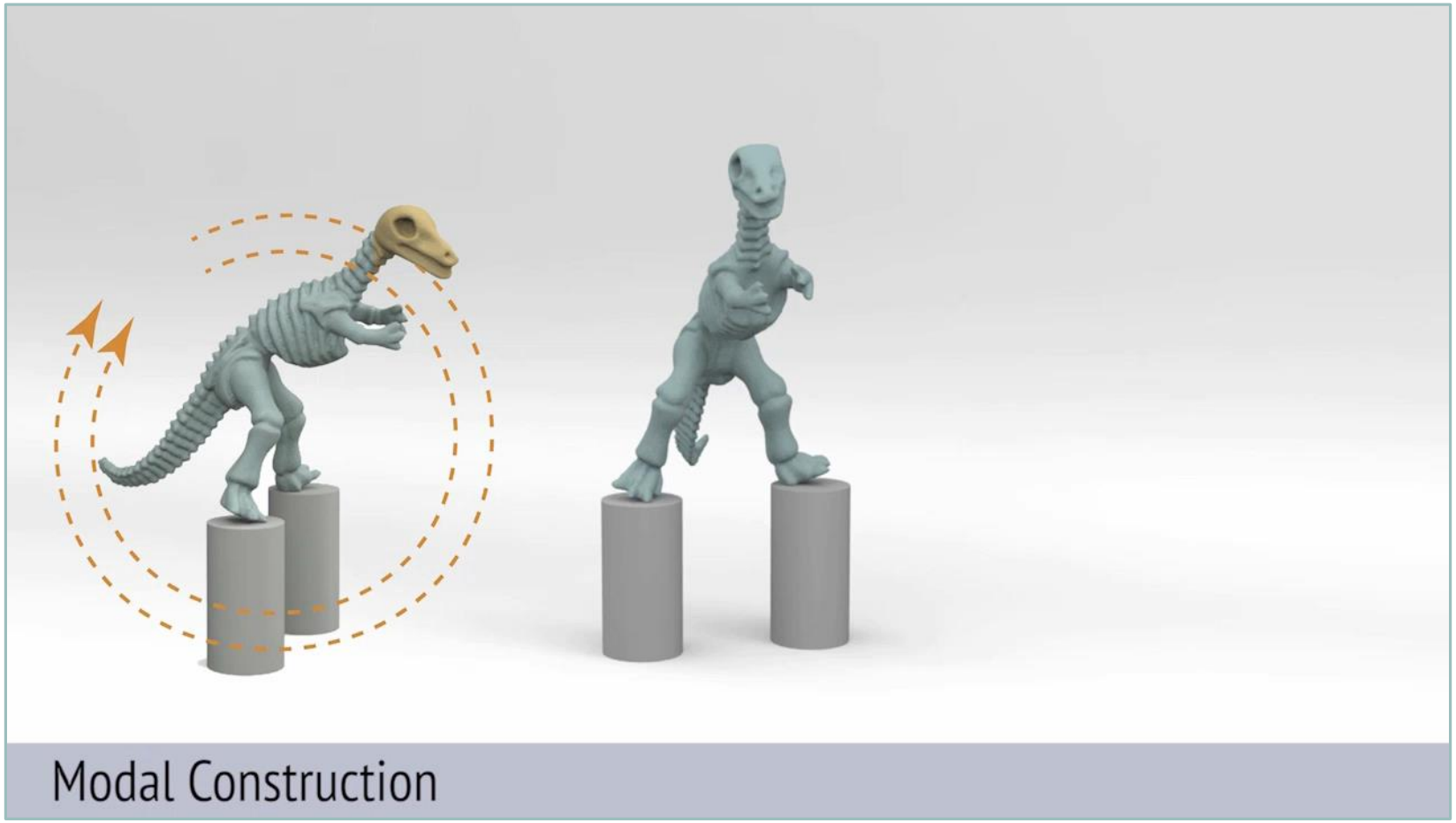
Expand using Taylor series (in tensor notation)

$$\mathbf{K}(\delta\mathbf{u})^{-1} = (\mathbf{K} + \delta\mathbf{K})^{-1} = \mathbf{K}^{-1} + \mathbf{K}^{-1}\delta\mathbf{K}\mathbf{K}^{-1} + \text{H.O.T.}$$

where $\delta\mathbf{K}$ is the stiffness differential

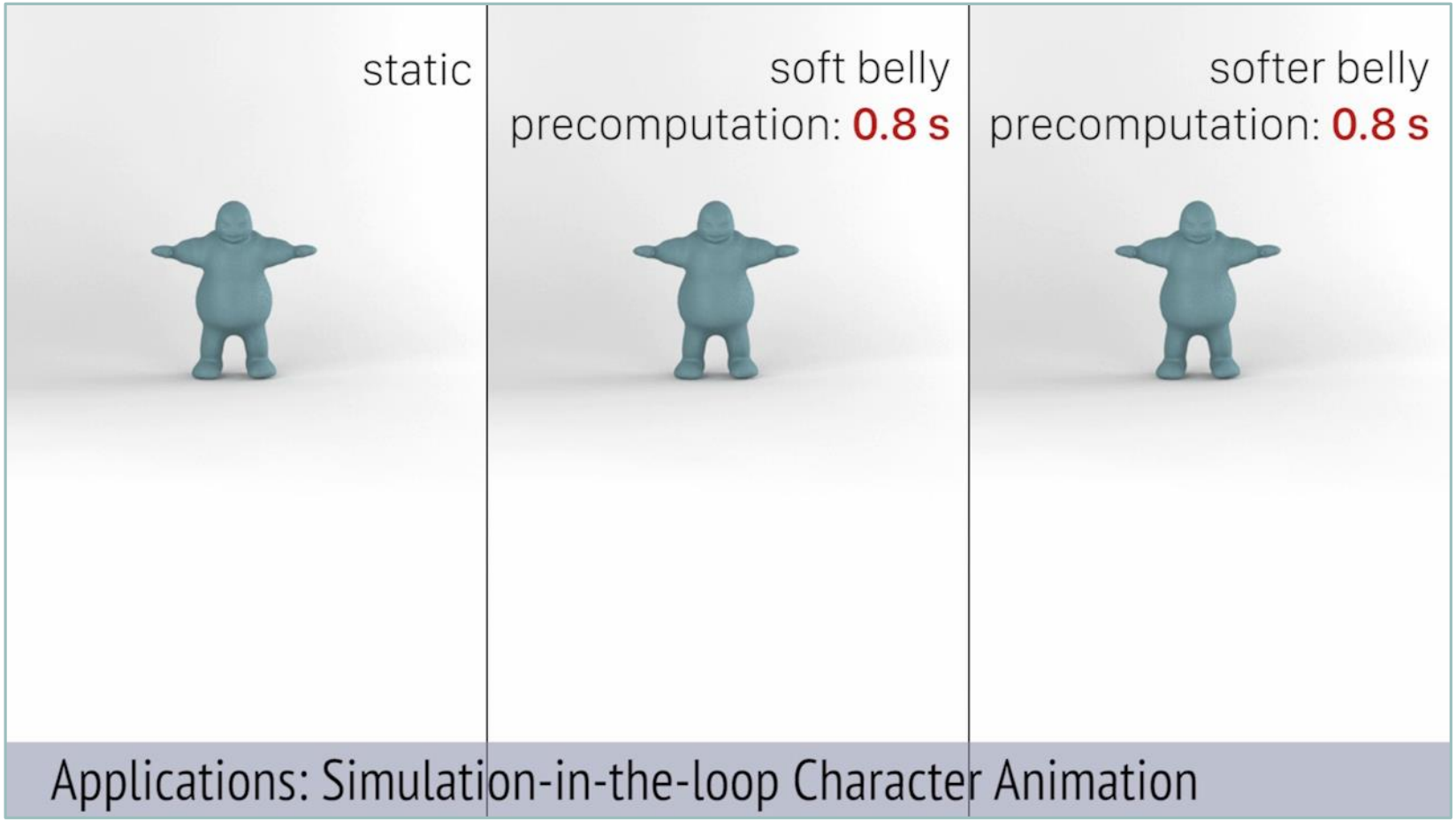
- A recurrence approximation of arbitrary high-order!
  - State-of-the-art is one order approximation (known as the modal derivative)
- Essentially a block-wise Krylov iteration

Inertia Modes



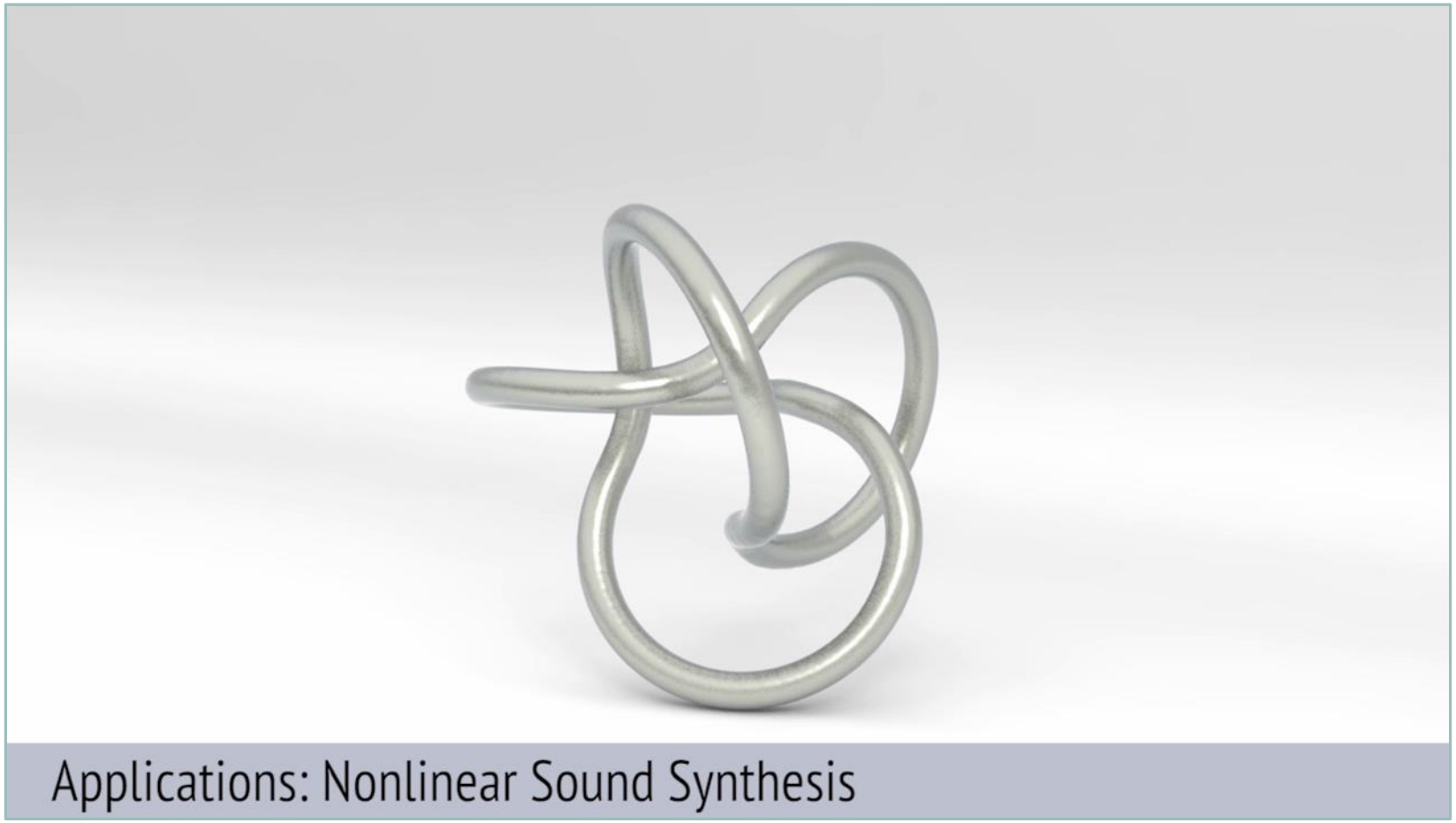
[Yang et al. 2015]

Example I



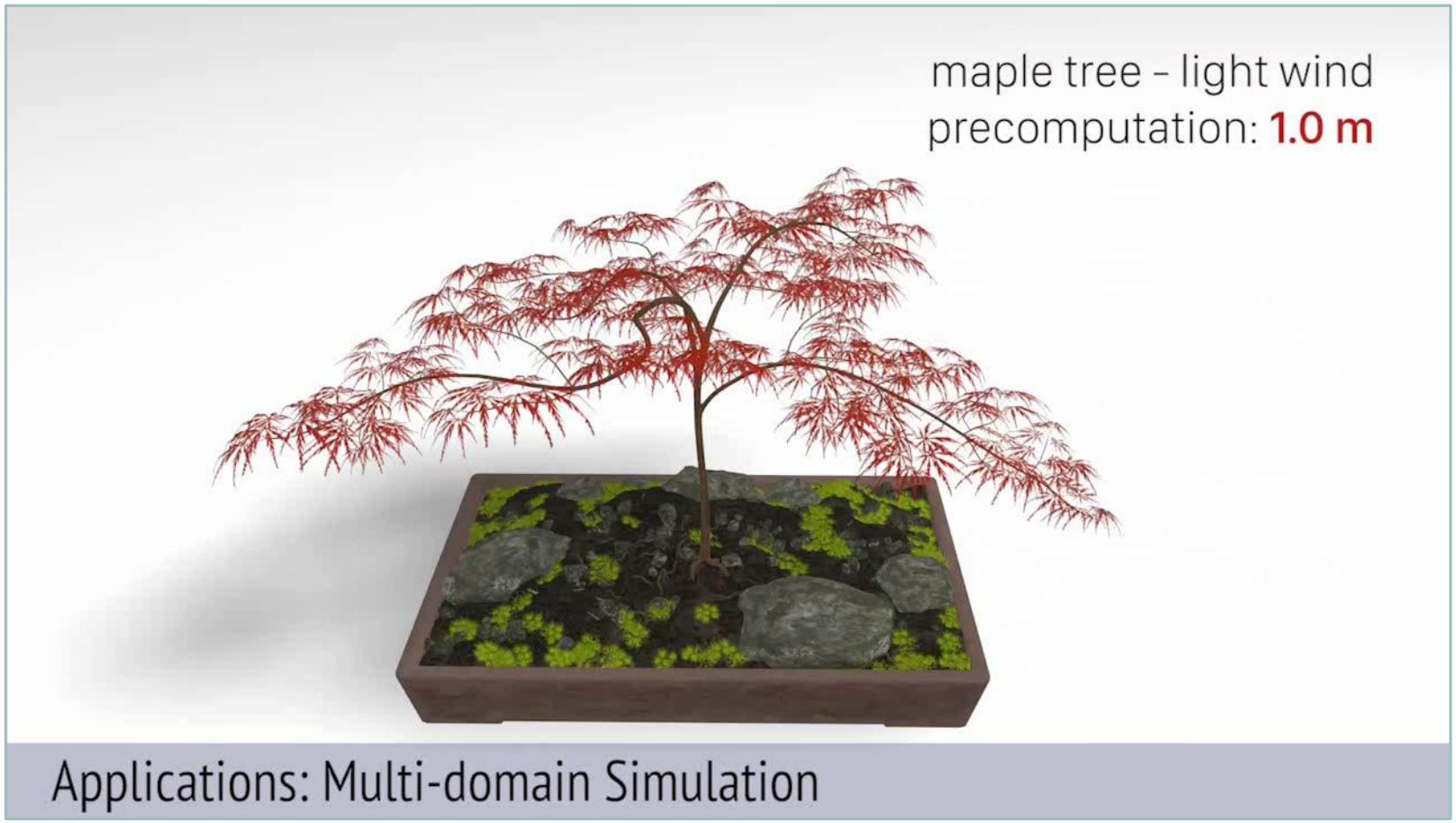
[Yang et al. 2015]

Example II



[Yang et al. 2015]

Example III



[Yang et al. 2015]

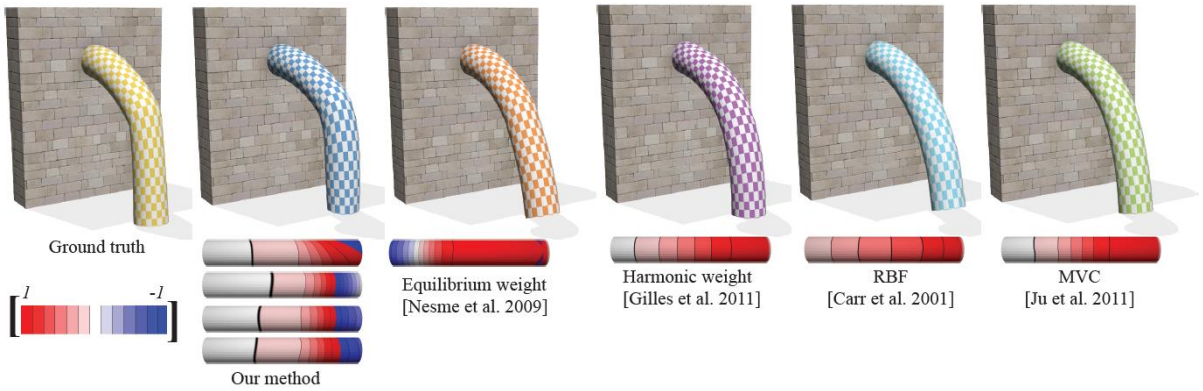
Towards Extreme Deformation

- Explicit interface constraints downgrade numerical stability
  - Highly deformed interface leads very big constraint force
  - Orders-of-magnitude larger than elastic internal force

[Luo et al. 2017]

Towards Extreme Deformation

- How to make the simulation efficient, versatile and *stable*
  - Get rid of interface constraint
  - How to glue domains and prevent cracks at the interface
- Make them overlapping (similar to skinning)
  - A material point in the overlapping region receives contribution from multiple adjacent domains
  - No duplicated DOFs



[Luo et al. 2017]

Weight Function

- What is the best weight function
  - Many geometry-based interpolation functions are possible
- Formulated as a QCQP, solve the weight that is consistent with the most visible deformation of a domain
  - Different external forces yield different deformation patterns
  - The force comes from a direction that maximize the visual effect

$$\arg\max_{\mathbf{p}} \|\mathbf{u}\|$$

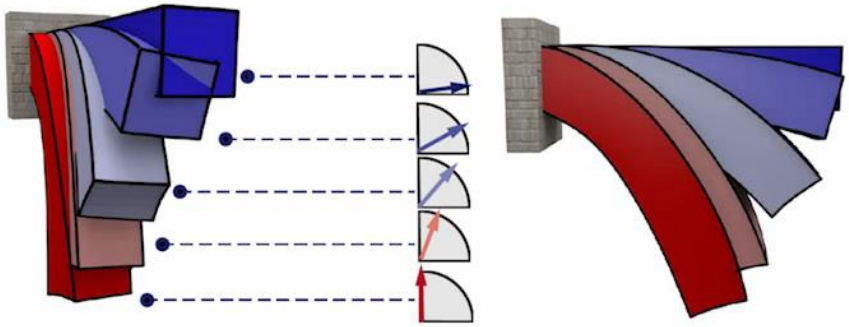
$$\text{subject to} \begin{bmatrix} \mathbf{K}_{ss} & \mathbf{K}_{sn} & \mathbf{0} \\ \mathbf{K}_{sn}^{\cdot} & \mathbf{K}_{nn} & \mathbf{C}^{\cdot} \\ \mathbf{0} & \mathbf{C} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{u}_s \\ \mathbf{u}_n \\ \lambda \end{bmatrix} = \begin{bmatrix} \mathbf{p} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix},$$

$$\text{and} \quad \|\mathbf{p}\| = 1$$

Principal Direction

Experiment 2

Principal direction



(Fig. 7 in the paper)

[Luo et al. 2017]

Some Examples

Experiment 4

Elastic multi-weight enveloping

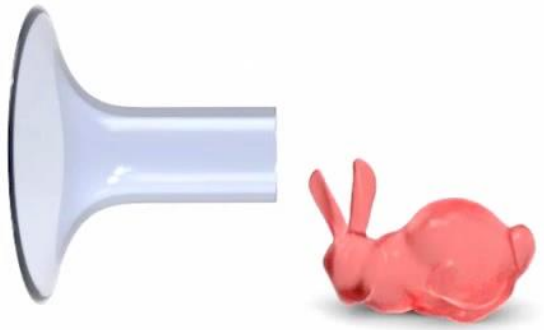
(Fig. 8 in the paper)

[Luo et al. 2017]

Highly Robust

Funnelling the bunny

Num. of elements: 64K
Num. of domains: 5 (150 DOFs)
Pre-computation: 2 min
Simulation FPS: 26
Time step size: 1/100 s



[Luo et al. 2017]

GPU Deformable Model

- Model reduction loses accuracy ($\mathbf{u} = \Phi\mathbf{q}$)
- Can we achieve the real-time performance without reduction
  - DOFs are two-way coupled and solving a linear system is sequential
- Need dedicated algorithm for parallel hardware (i.e. GPU)

$$\mathbf{u}_{t+1} = \mathbf{u}_t + h\mathbf{v}_{t+1}$$
$$\mathbf{v}_{t+1} = \mathbf{v}_t + h\mathbf{M}^{-1}\mathbf{f}(\mathbf{u}_{t+1}) \quad \Rightarrow \quad \mathbf{M}(\mathbf{u}_{t+1} - \mathbf{u}_t - h\mathbf{v}_t) = h^2\mathbf{f}(\mathbf{u}_{t+1})$$

$$\mathbf{u}_{t+1} = \arg\min \varepsilon(\mathbf{u})$$

where $\varepsilon(\mathbf{u}) = \frac{1}{2h^2}(\mathbf{u} - \mathbf{u}_t - h\mathbf{v}_t)^{\cdot}\ \mathbf{M}(\mathbf{u} - \mathbf{u}_t - h\mathbf{v}_t) + E(\mathbf{u})$

Why Optimization

- A wide range of numerical methods are available
- Start from an initial guess $\mathbf{u}^{(0)}$ approaches to the target $\mathbf{u}^{*}$ following a descent direction $\Delta\mathbf{u}$

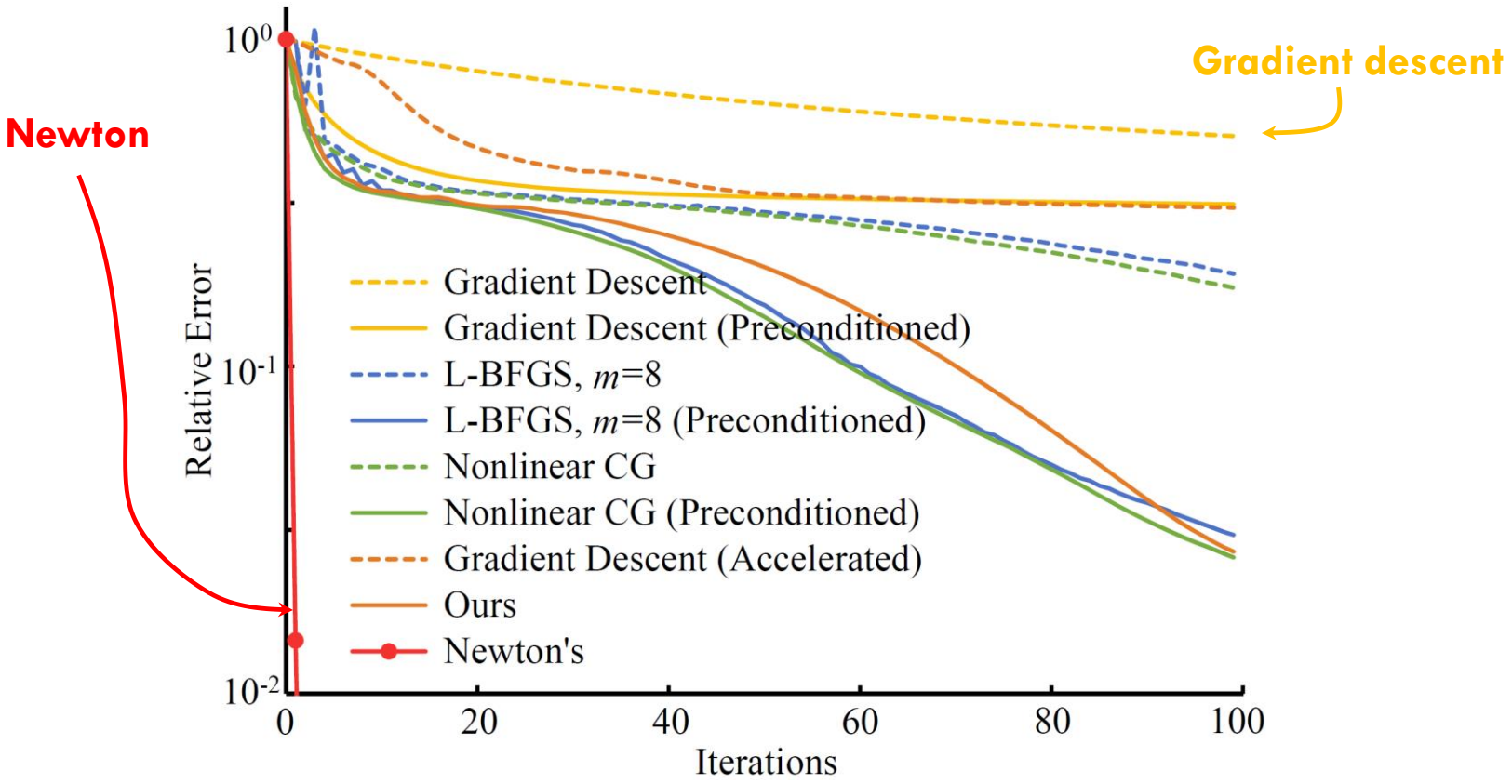
$$\mathbf{u}^{(k+1)} \leftarrow \mathbf{u}^{(k)} + \alpha^{(k)}\Delta\mathbf{u}^{(k)}$$

- The key question is how to compute a **good** $\Delta\mathbf{u}$ **efficiently**
  - Step length is also tweak-able (i.e. enforcing the Wolfe condition)
- Typically, computing a better descent direction is more expensive
- **Our method:** a good descent direction that is inexpensive (on GPU)

Review: Our Candidates

- Newton's method
  - Approximates $\mathcal{E}(\mathbf{u})$ by a quadratic function
  - Compute the descent direction as $\Delta\mathbf{u}^{(k)} = -\mathbf{H}^{(k)\bullet}\,\mathbf{g}^{(k)}$
  - Need to 1) evaluate the **full Hessian** and, 2) **solve** it
- Quasi-Newton method (L-/BFGS)
  - Newton's method with an approximate Hessian inverse (dense)
  - Need to 1) smartly guess the **Hessian inverse** and, 2) enforce the **secant condition**
- Nonlinear CG
  - Roughly speaking, similar to LBFGS but only memorize the previous secant condition
  - Need to compute vector **inner products**
- Gradient descent
  - Simply follow the negative gradient direction

Review: Our Candidates



Review: Our Candidates

- In terms of **energy reduction / iteration**

  Newton $>$ Quasi-Newton $\approx$ NLCG $\gg$ Gradient descent

- In terms of **computational cost / iteration**

  Newton $<$ Quasi-Newton $\approx$ NLCG $\ll$ Gradient descent

- Goal: fast **energy reduction / time**

- We choose the gradient descent (GD) method
  - **Pros:** simple computation, very fast each iteration (<1 ms vs hundreds ms)
  - **Cons:** converge slowly without special treatment

Make GD Great Again

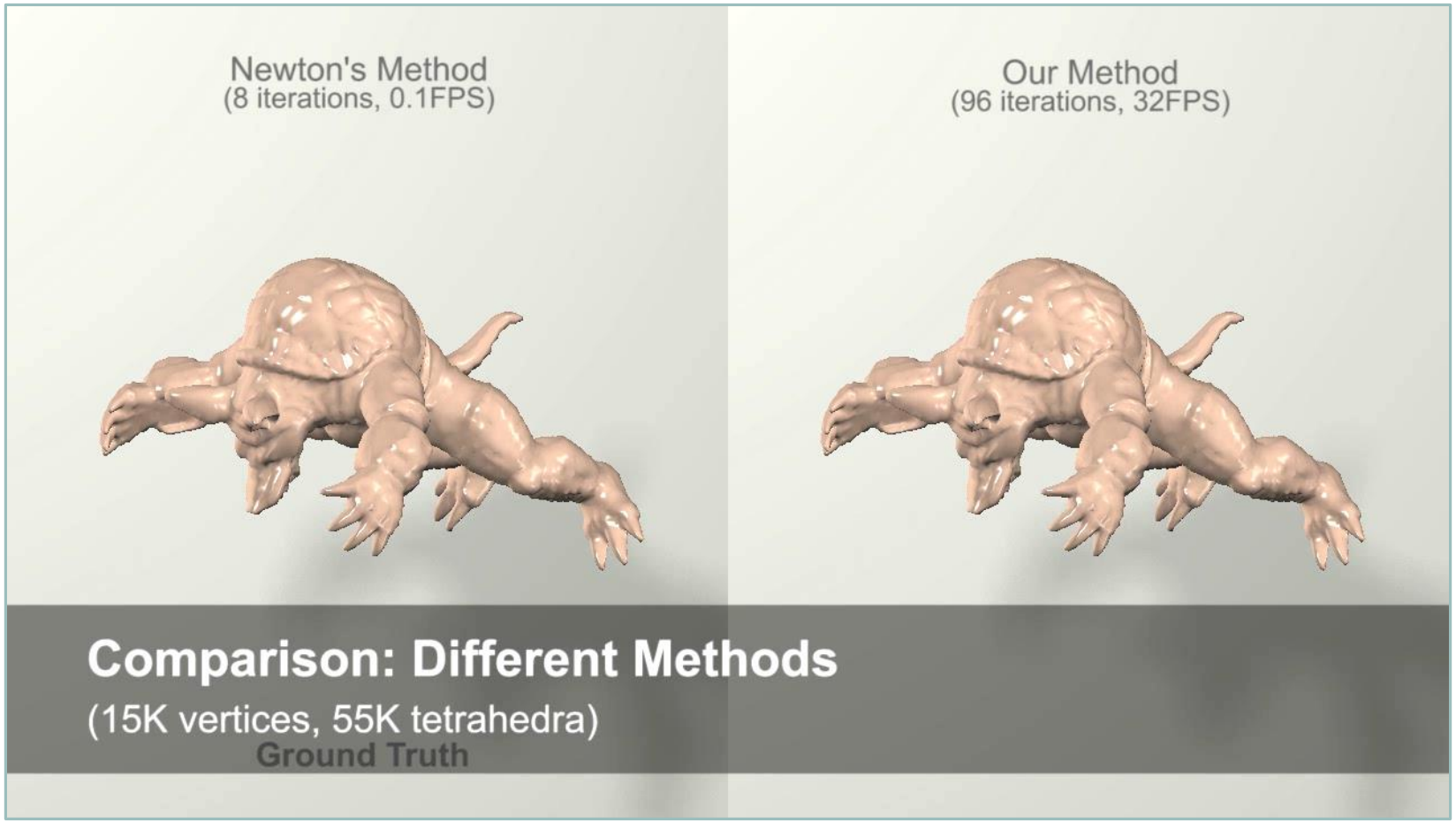
- Good preconditioning
- Momentum-based optimization (Chebyshev acceleration)
- Better initial guess (warm start)
- Adjust step length (simplified Wolfe condition)
- All of above does not destroy the advantages of GD
  - Simple computation
  - Parallelization friendly

Performance

- Our method is not particularly attractive on CPU
- However, it can be significantly accelerated by GPU (each nonlinear iteration takes less than 1 ms)
- Note that this is **fullspace simulation**

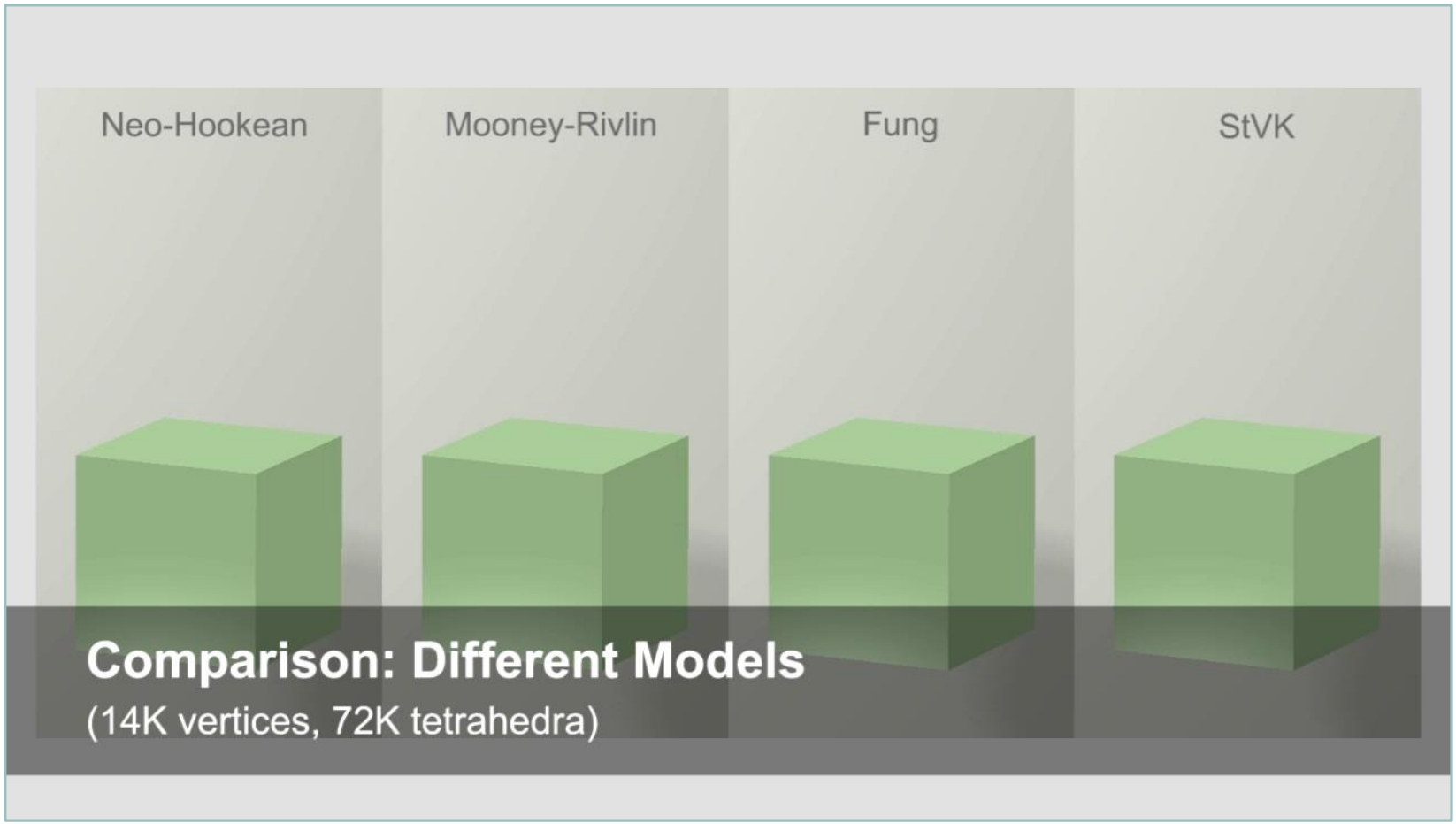
Name	#vert	#ele	CPU Cost	GPU Cost	GPU FPS
Dragon (Fig. 1)	16K	58K	1.35s	32.8ms	30.5
Armadillo (Fig. 2)	15K	55K	1.28s	31.4ms	31.8
Box (Fig. 7)	14K	72K	1.47s	37.6ms	26.6
Dress (Fig. 4)	15K	44K	0.29s	26.6ms	37.6
Double helix (Fig. 9)	13K	41K	0.98s	27.5ms	36.4
Double helix (Fig. 9)	24K	82K	1.91s	38.5ms	26.0
Double helix (Fig. 9)	48K	158K	3.86s	65.4ms	15.3
Double helix (Fig. 9)	96K	316K	7.78s	122ms	8.2

Result (Real-time Screen Record)



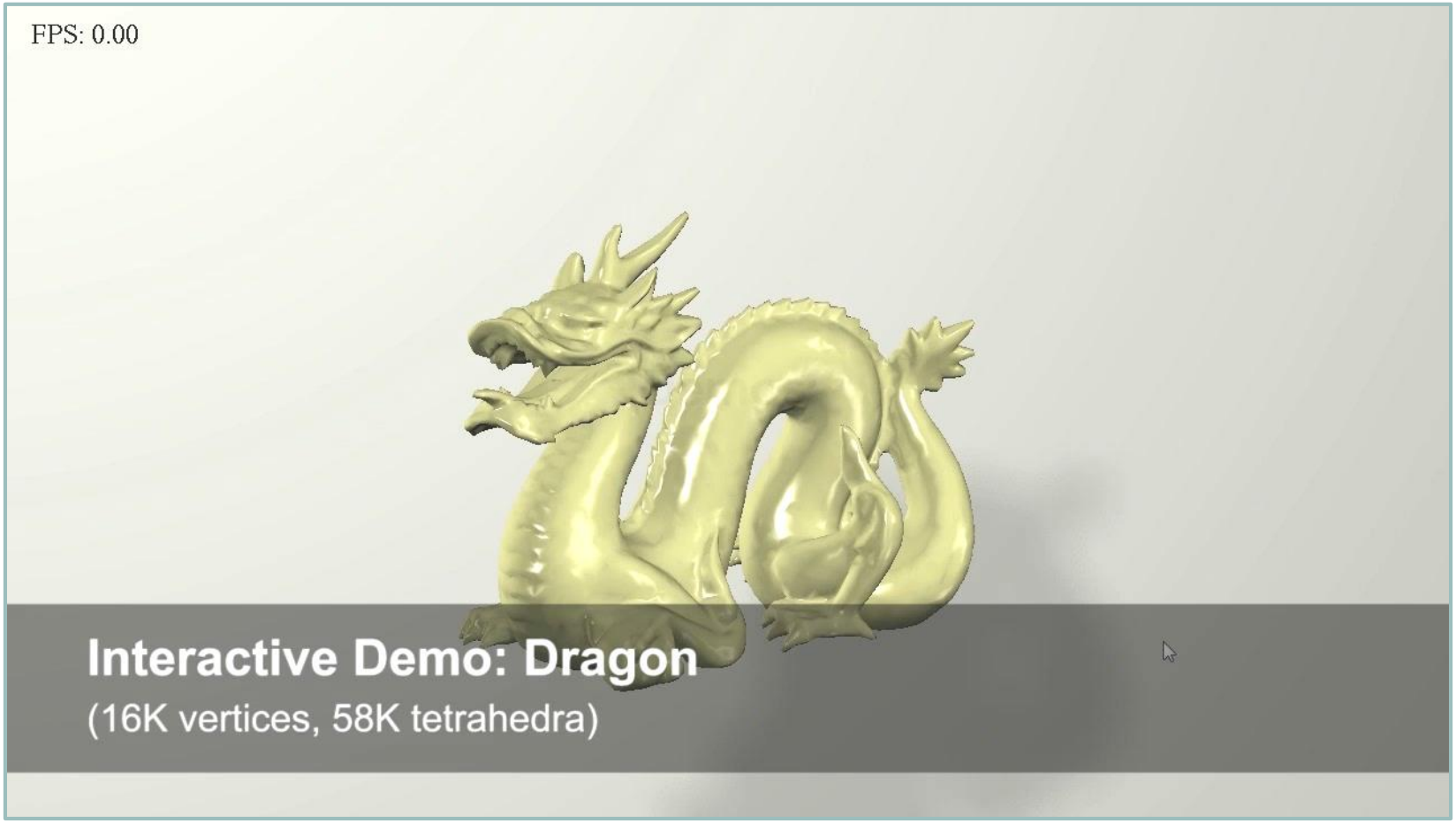
[Wang & Yang 2016]

RESULT (VARIOUS MATERIALS)



[Wang & Yang 2016]

Robust Under Interaction



[Wang & Yang 2016]

Data-driven Simulation With NN

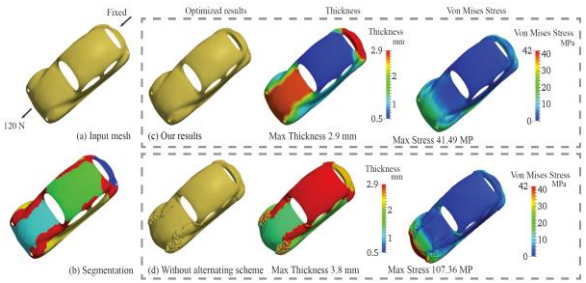
- Simulation provides us unlimited **clean** data
- Can we skip expensive computation in the simulation by leveraging existing seen simulation
- In theory yes; in practice difficult
  - Simulation is highly sophisticated
  - Very high-dimension input (tessellation, material parameters, simulation parameters, numerical parameters etc)
  - Huge network and tons of training data
- We are inspired by ResNet
  - Use NN as a correction mechanism not a simulation mechanism
  - Consider a baseline simulator, which is simplified and inexact
  - Use NN to align it to be the correct one

Maple Bonsai in Fullspace

[Luo at al 2018]

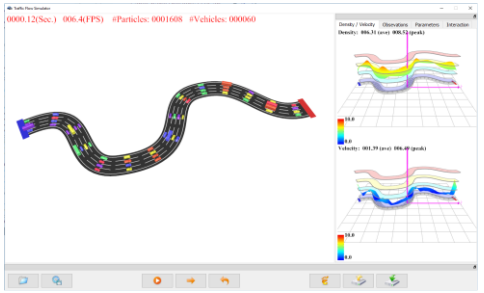
Simulation Beyond Animation

- A powerful tool in many applications beyond graphics
  - Computer vision e.g. real-time SLAM
  - Medical simulation e.g. tongue simulation for communication disorder
  - Intelligent CPS e.g. pavement distress detection
  - Digital fabrication e.g. simulation-in-the loop CAD
  - Computational transportation e.g. physics-based traffic dynamics



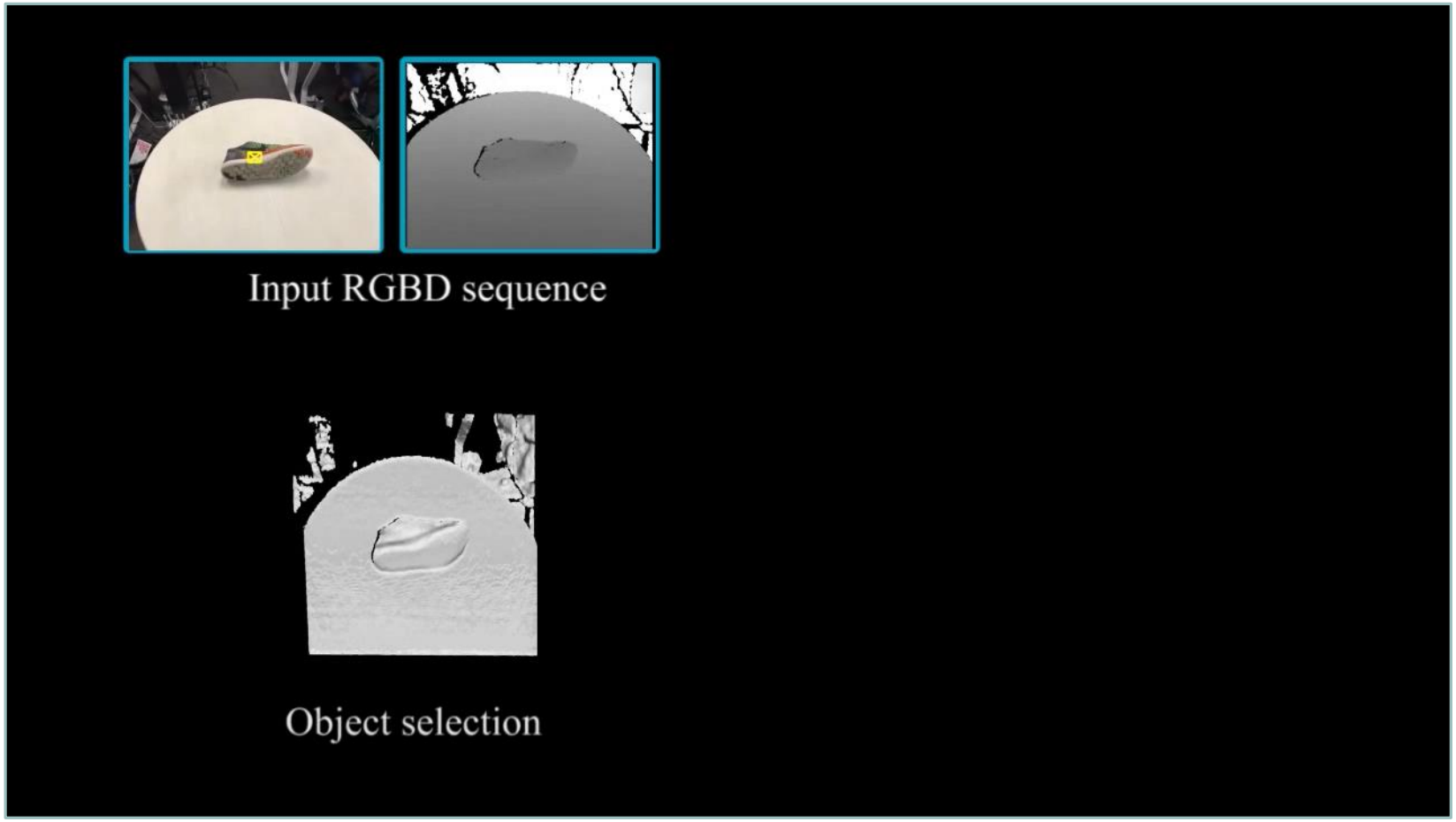
[Zhao et al. 2017]

[Xu et al. 2018]



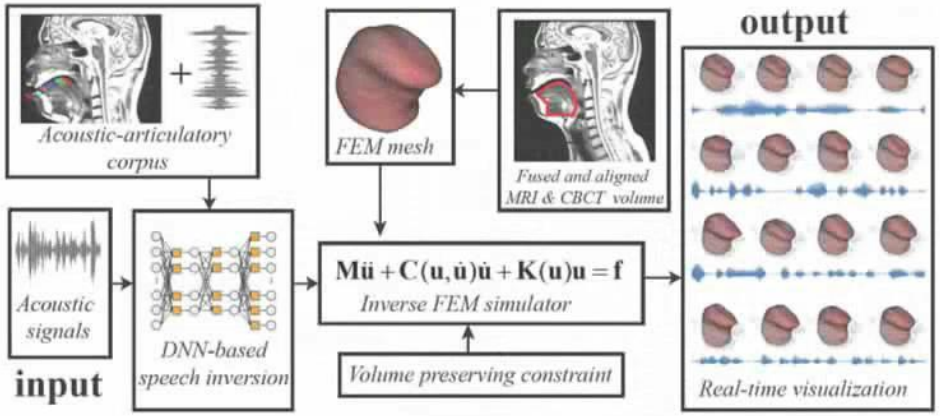
[Zhang et al. 2018]

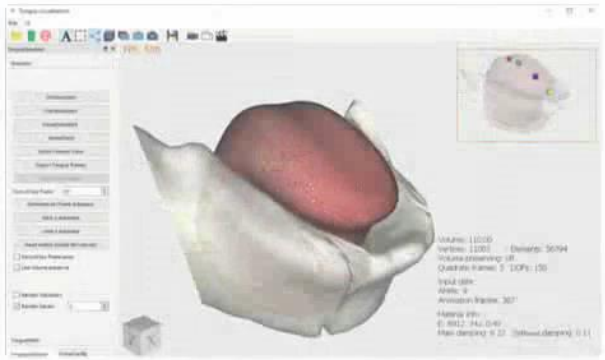
Application: Better SLAM



[Xu et al. 2018]

Application: Medical Simulation

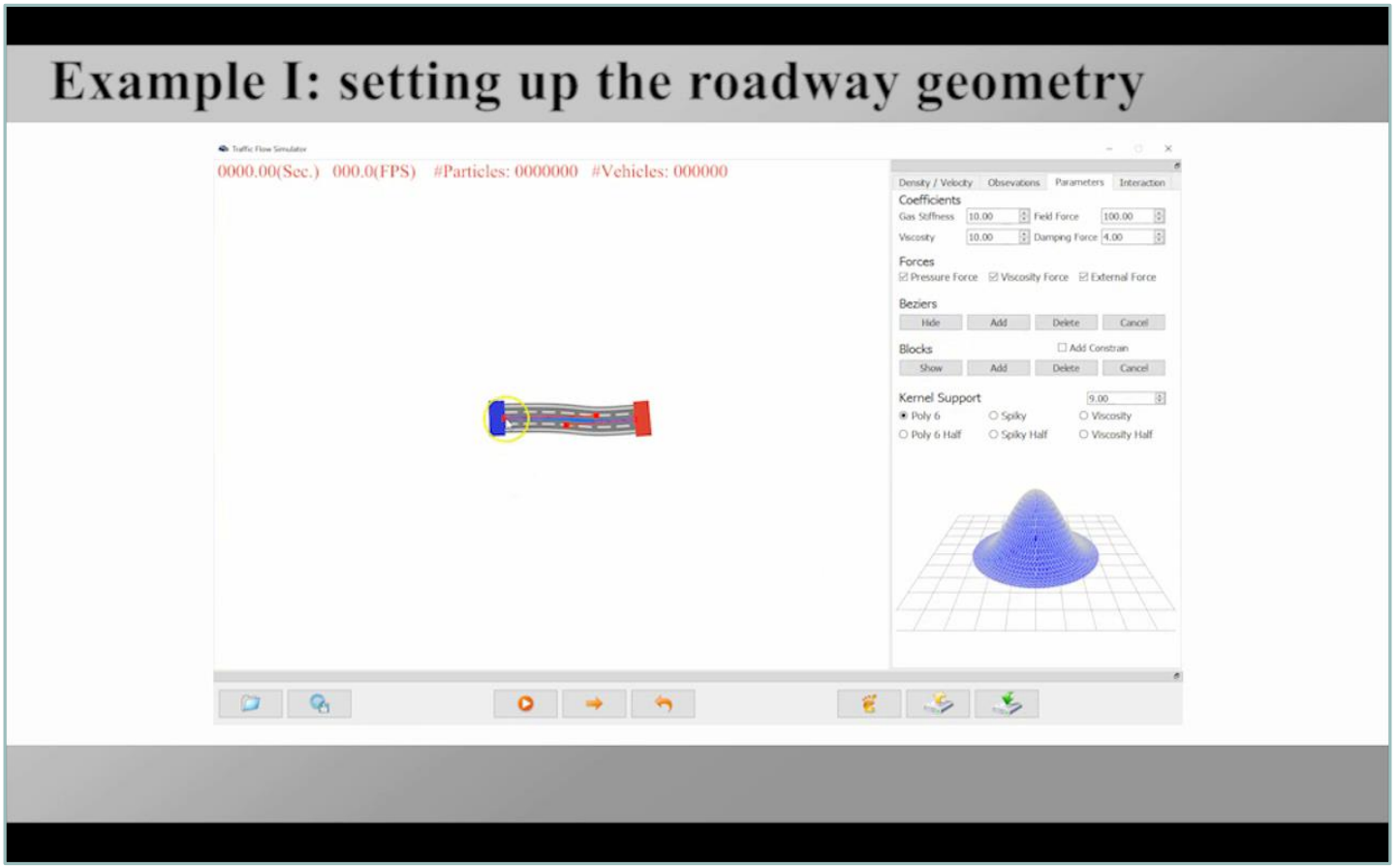




We propose an acoustic VR system that converts acoustic signals fo human language to realistic 3D tongue animation sequences in real time.

[Luo et al. 2016]

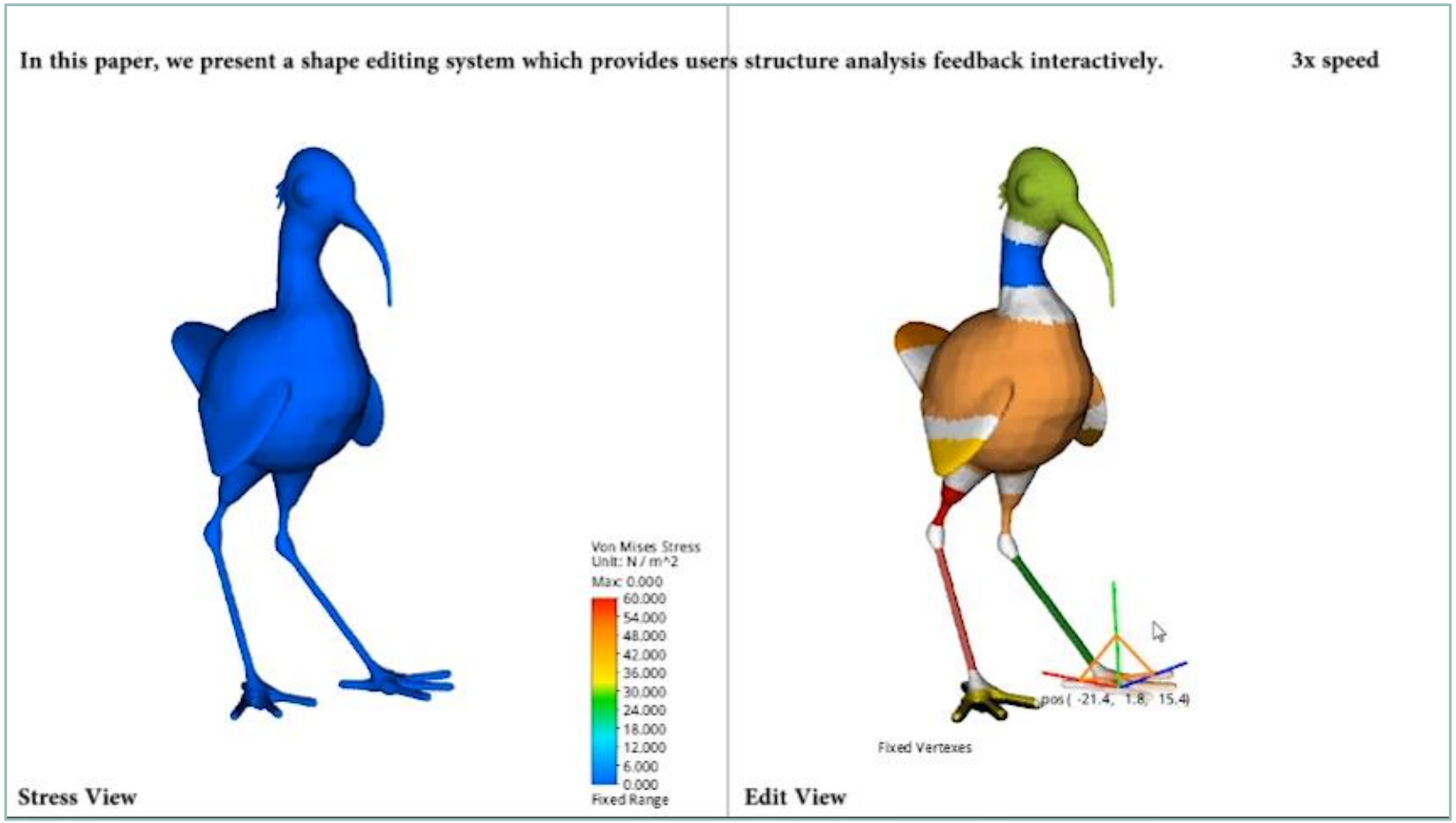
Application: Computational Transportation



[Zhang et al. 2018]

THE UNIVERSITY of NEW MEXICO

Application: Digital Fabrication



[Xu et al. 2017]

Thank You

THE UNIVERSITY of NEW MEXICO