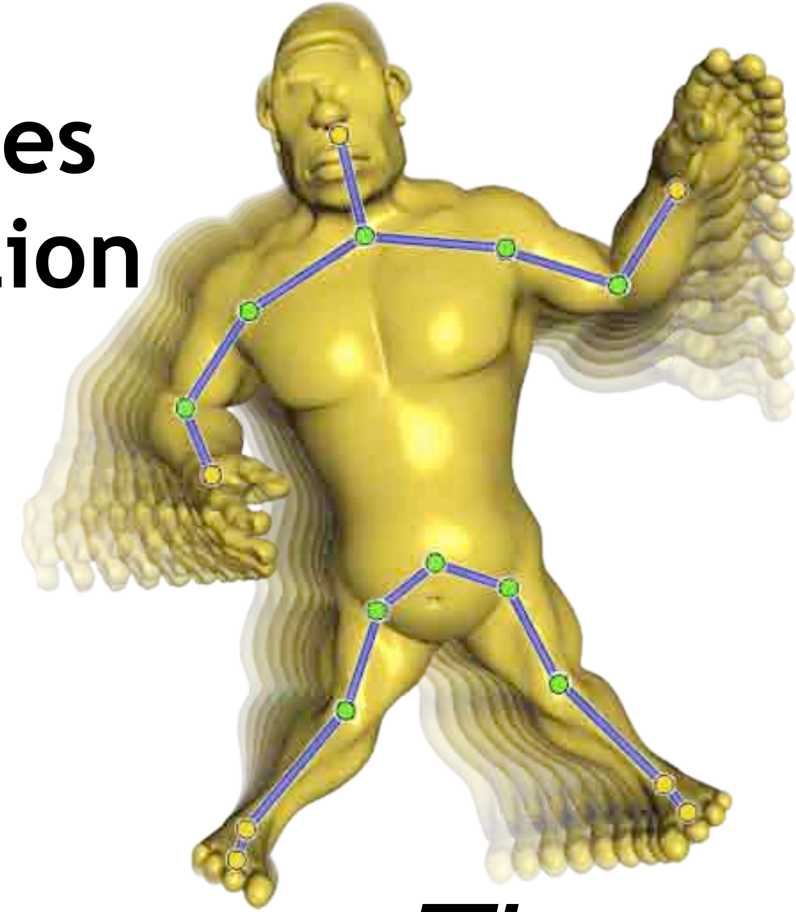
Algorithms and Interfaces for Real-Time Deformation of 2D and 3D Shapes



Alec Jacobson

ETH Zurich

INTERACTIVE GEOMETRY LAB

June 9, 2013

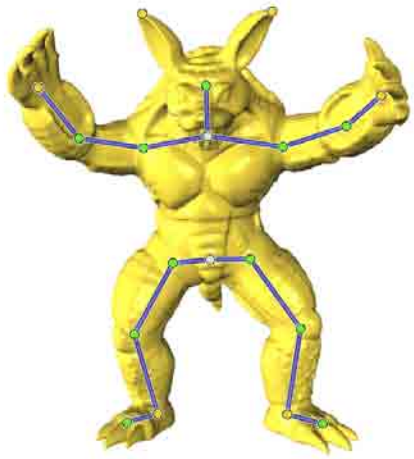
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Deformation is an important phase in the life of a shape

creation

modeling in Maya

analysis and manipulation



skinning deformation

consumption

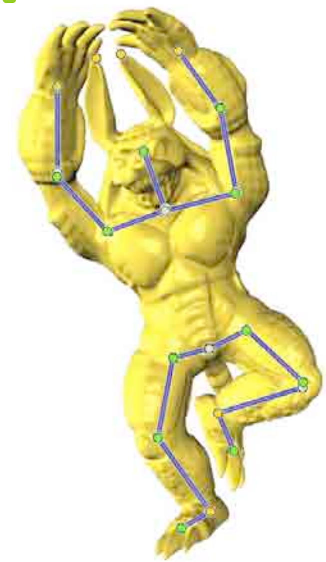
3d printing

Deformation is an important phase in the life of a shape

creation

modeling in Maya

analysis and manipulation



skinning deformation

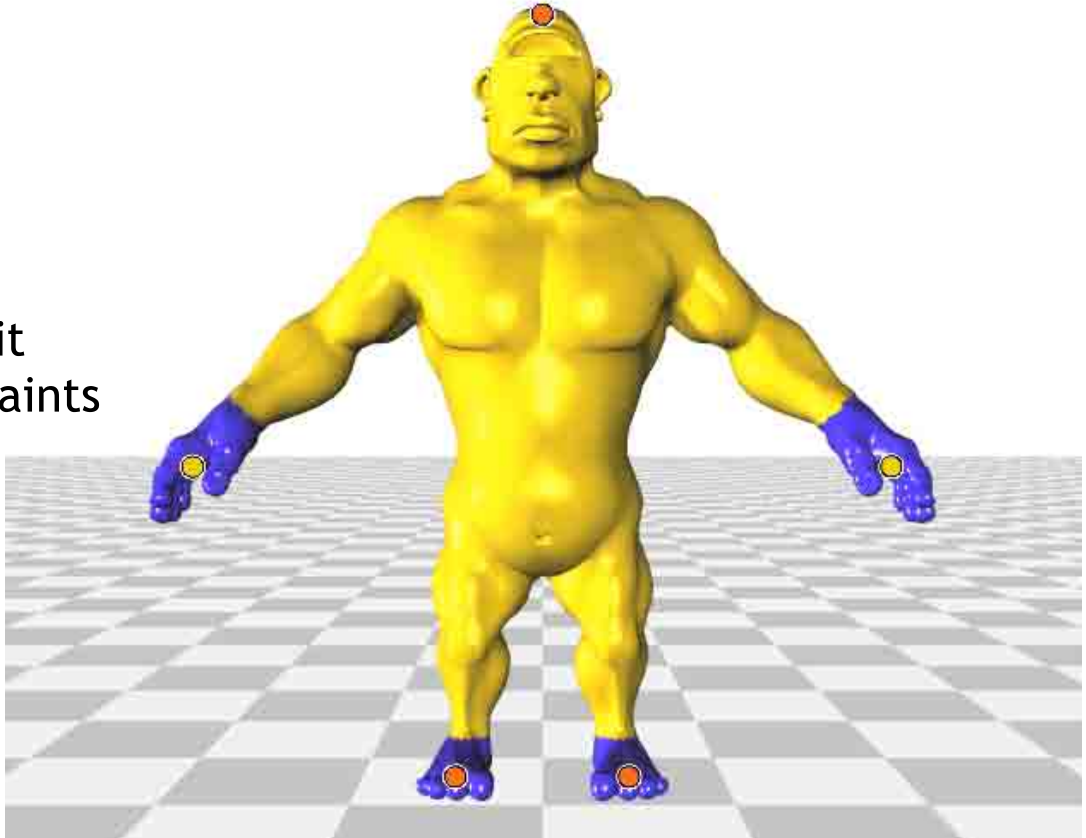
consumption

3d printing

User constraints drive deformation toward goal

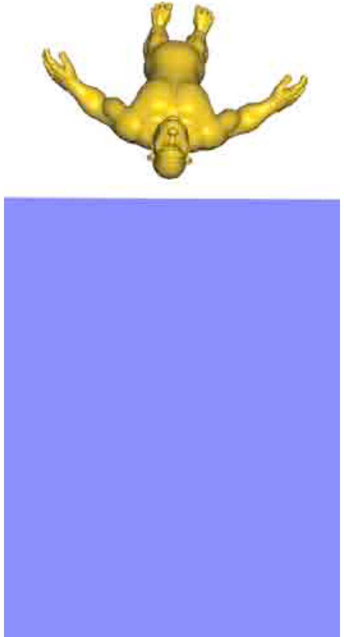
procedural
constraints

explicit
constraints

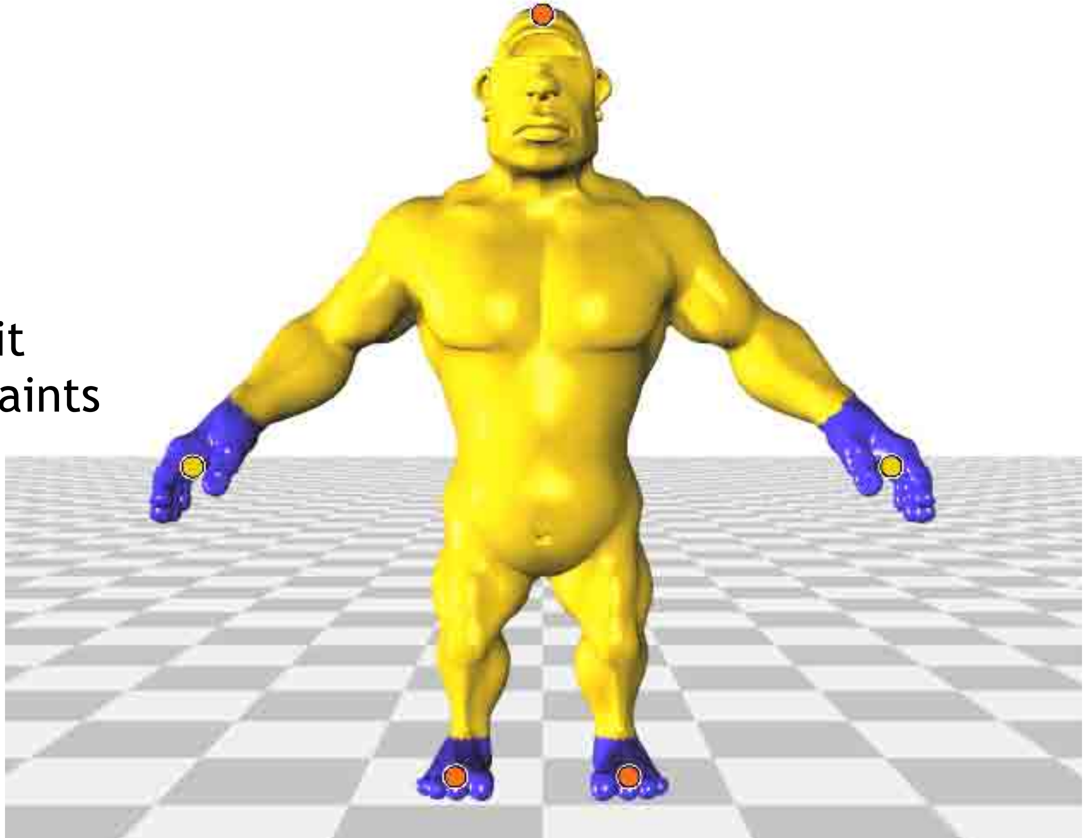


User constraints drive deformation toward goal

procedural constraints



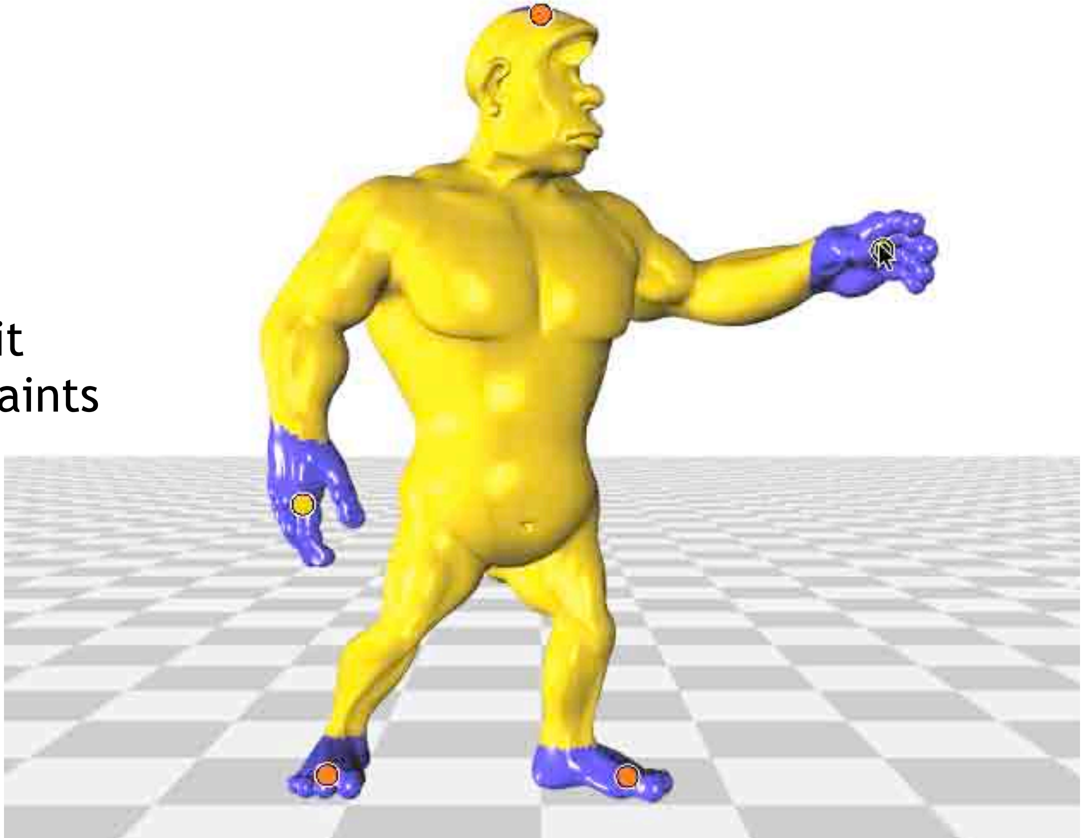
explicit constraints



User constraints drive deformation toward goal

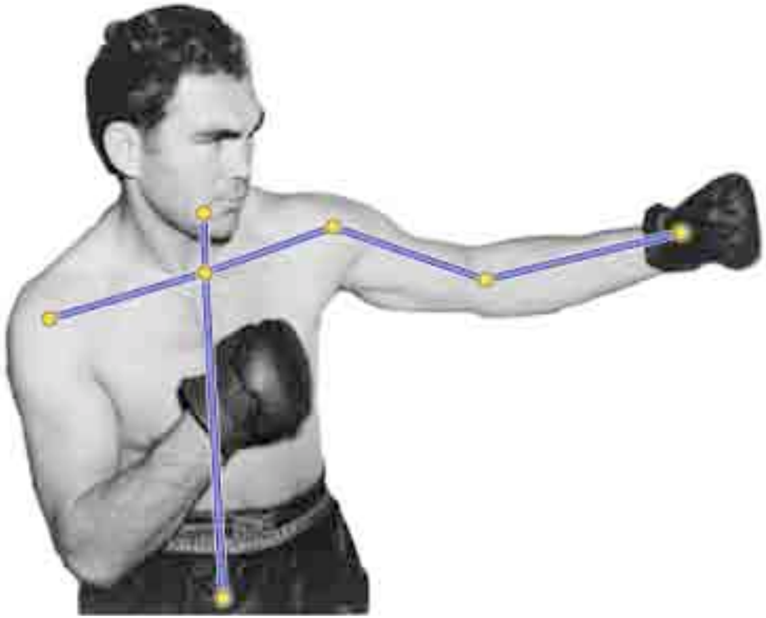
procedural
constraints

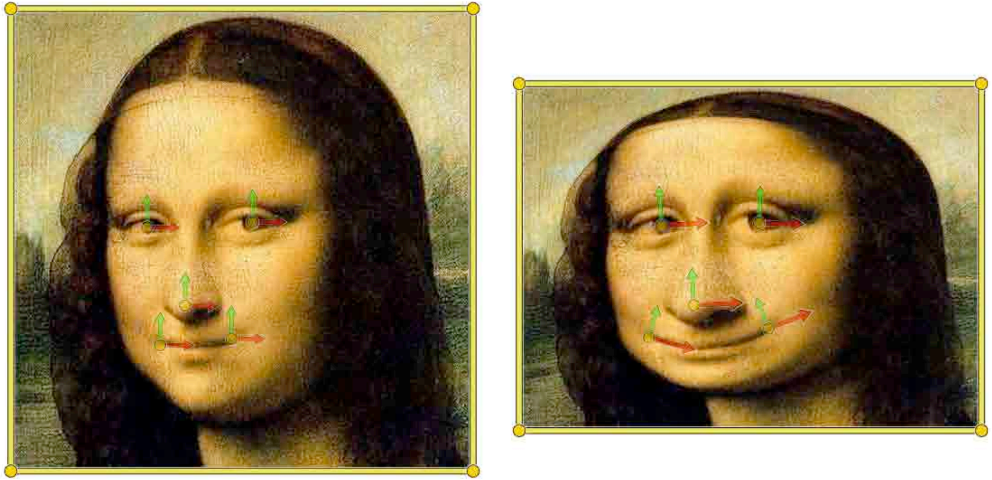
explicit
constraints



Deformation applies to images as planar shapes

non-convex "cut-out" cartoons

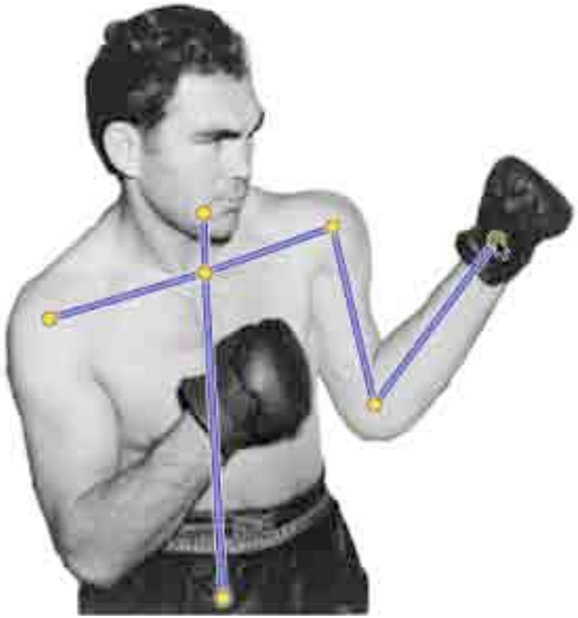


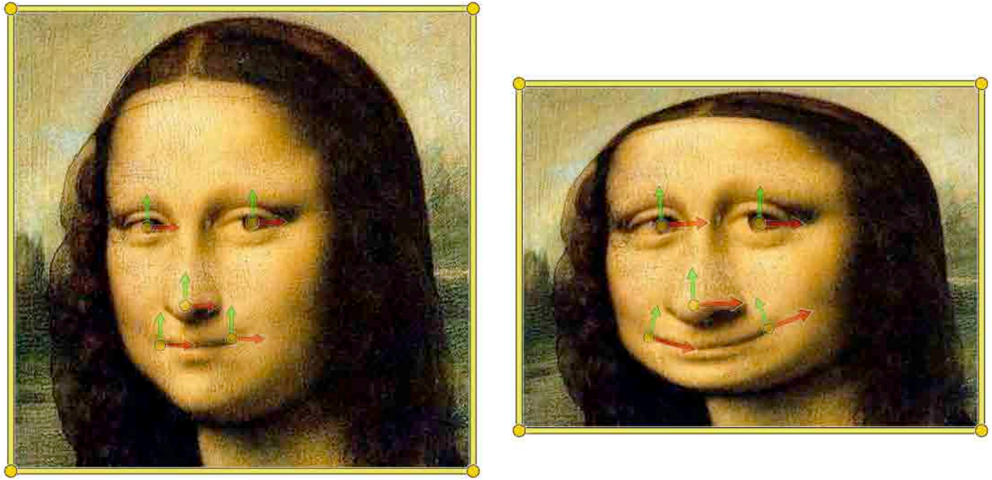


entire image rectangle

Deformation applies to images as planar shapes

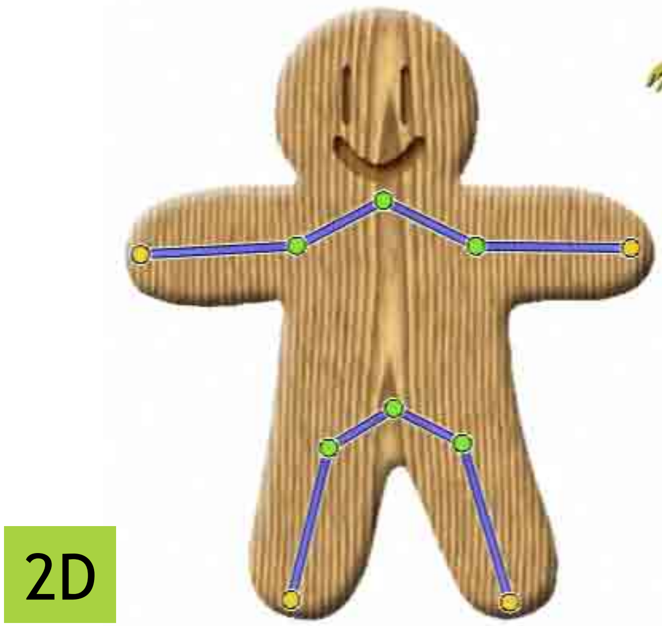
non-convex “cut-out” cartoons

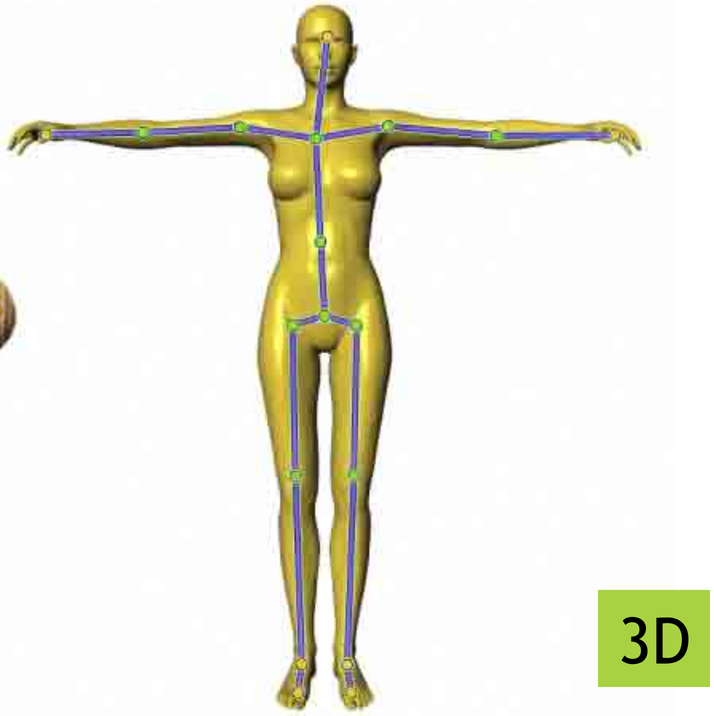




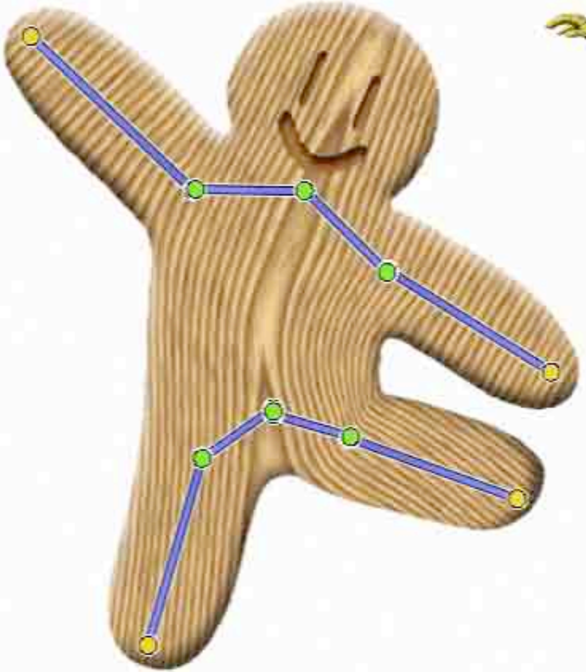
entire image rectangle

Real-time performance critical for interactive design and animation





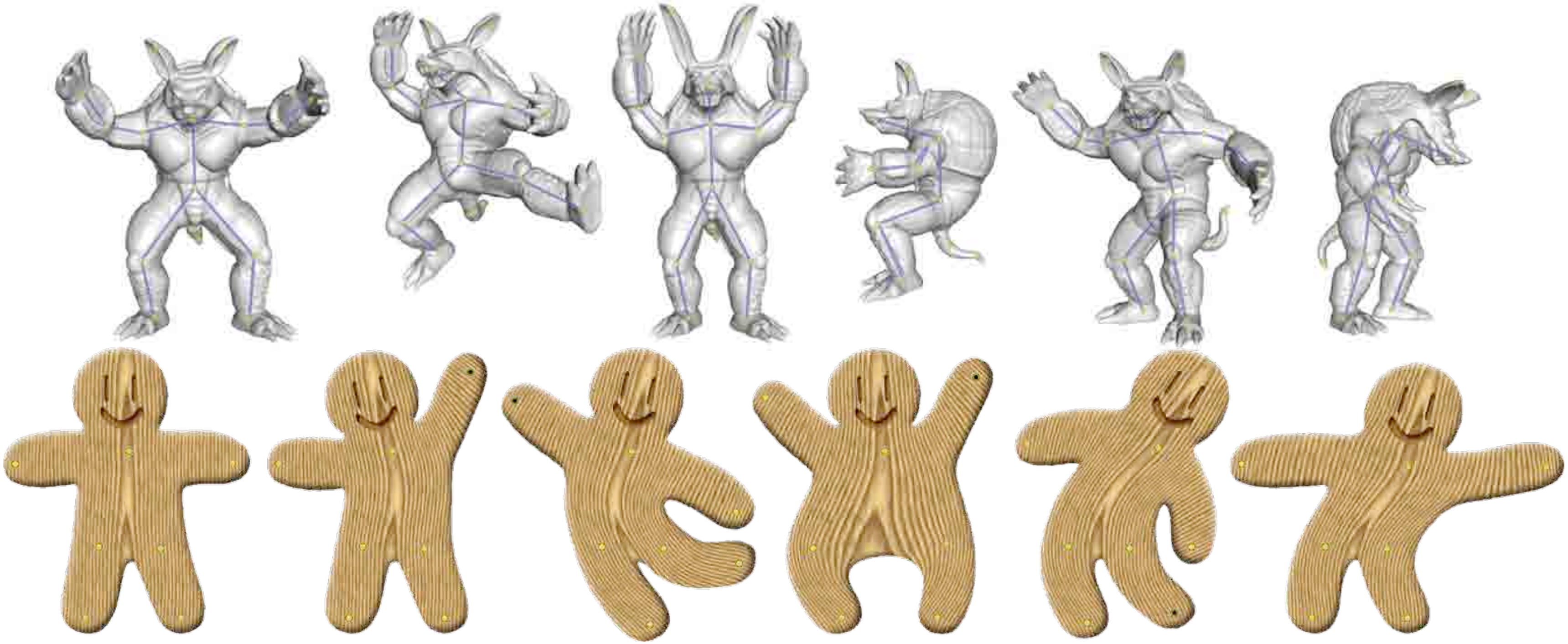
Real-time performance critical for interactive design and animation



2D

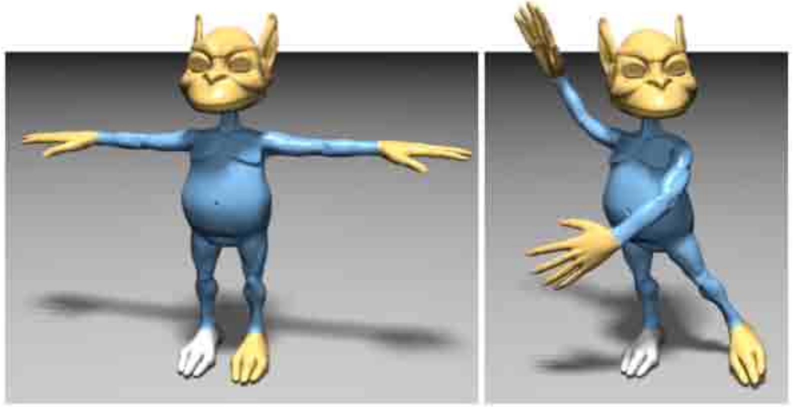
3D

Real-time performance critical for interactive design and animation



Many previous techniques provide quality, but not speed

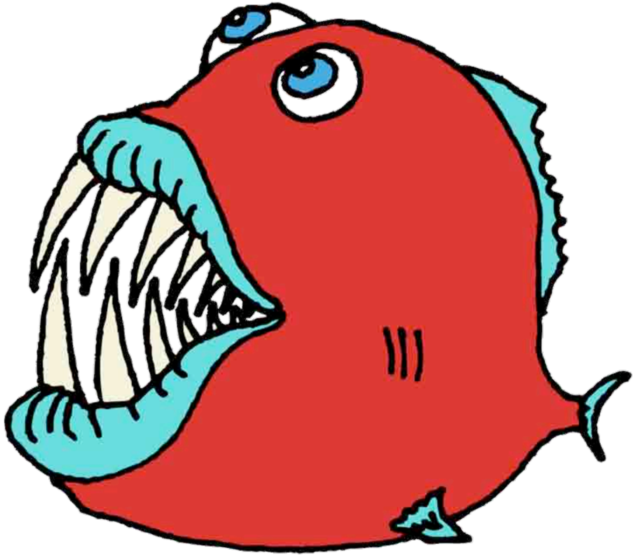
high-quality solutions to nonlinear elasticity energy minimizations:
~*seconds*
e.g. [Botsch et al. 2006]



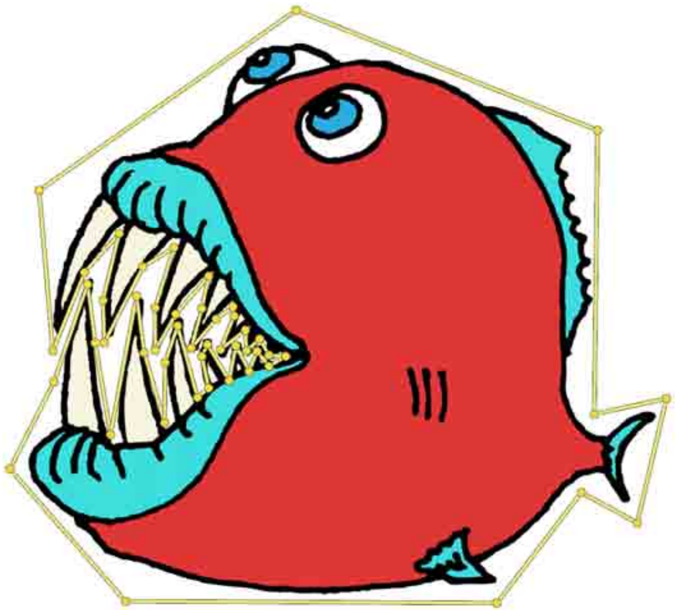
physically accurate muscle systems require off-line simulation
e.g. [Teran et al. 2005]

Image courtesy Joseph Teran

Other techniques limit user interface

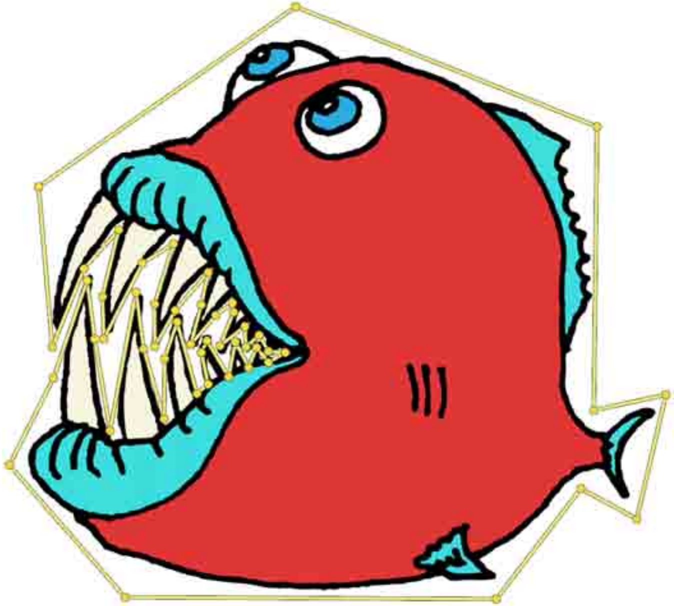


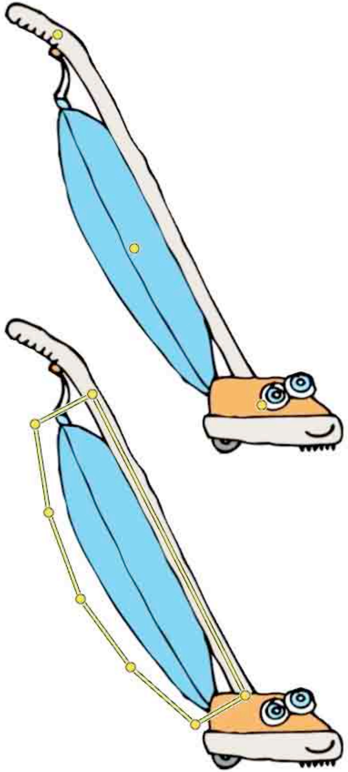
Other techniques limit user interface



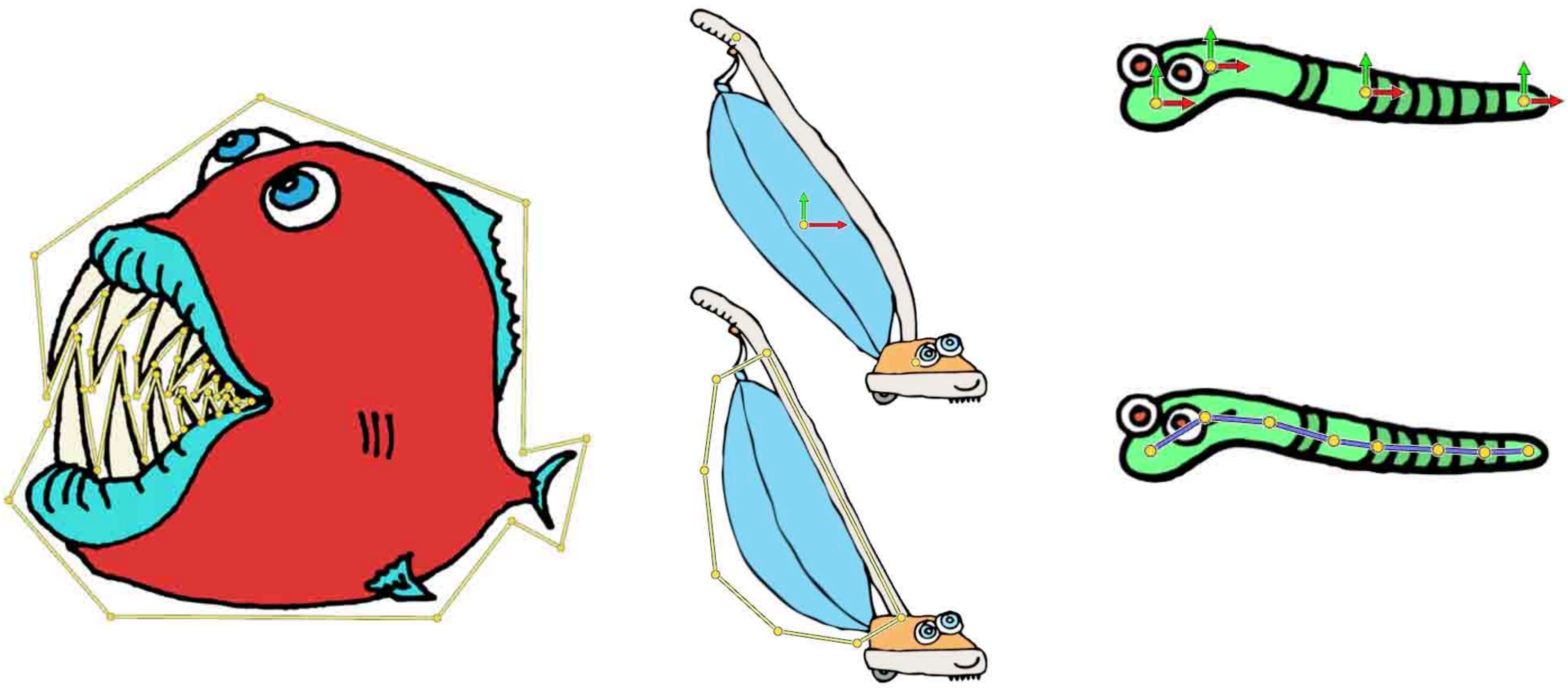
e.g. [Ju et al. 2005]

Other techniques limit user interface

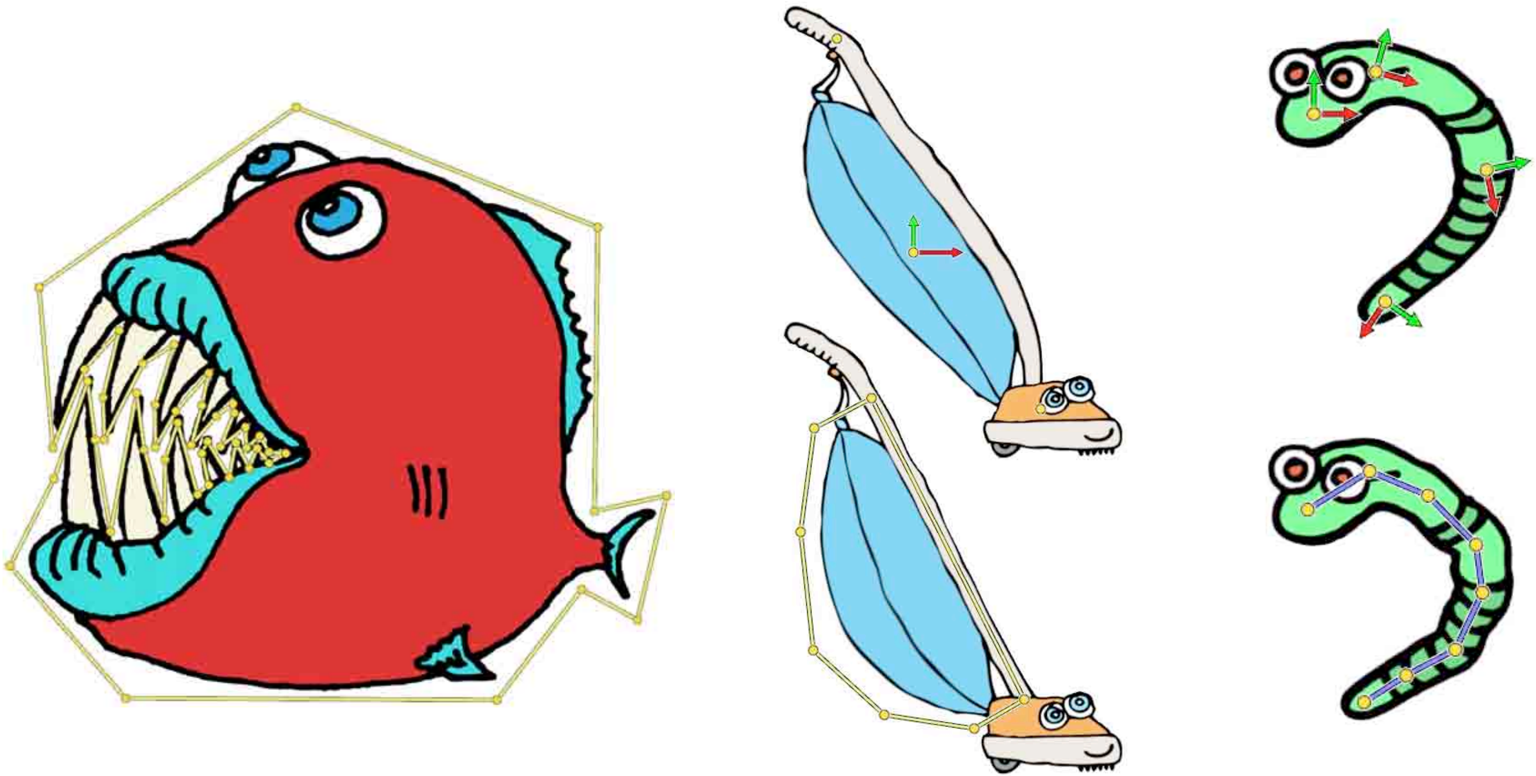




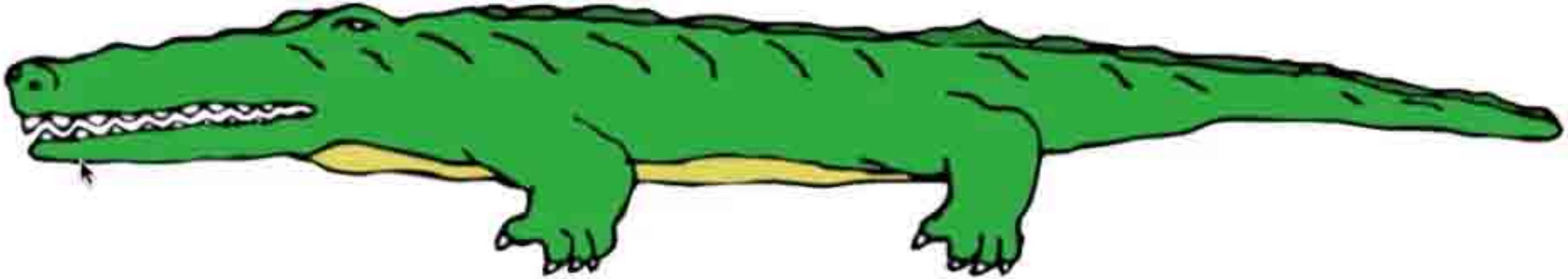
Other techniques limit user interface



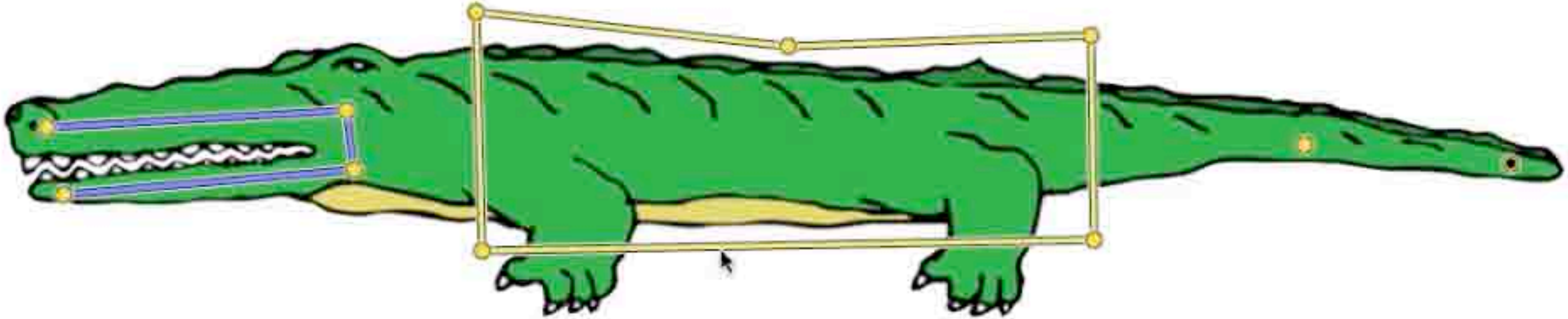
Other techniques limit user interface



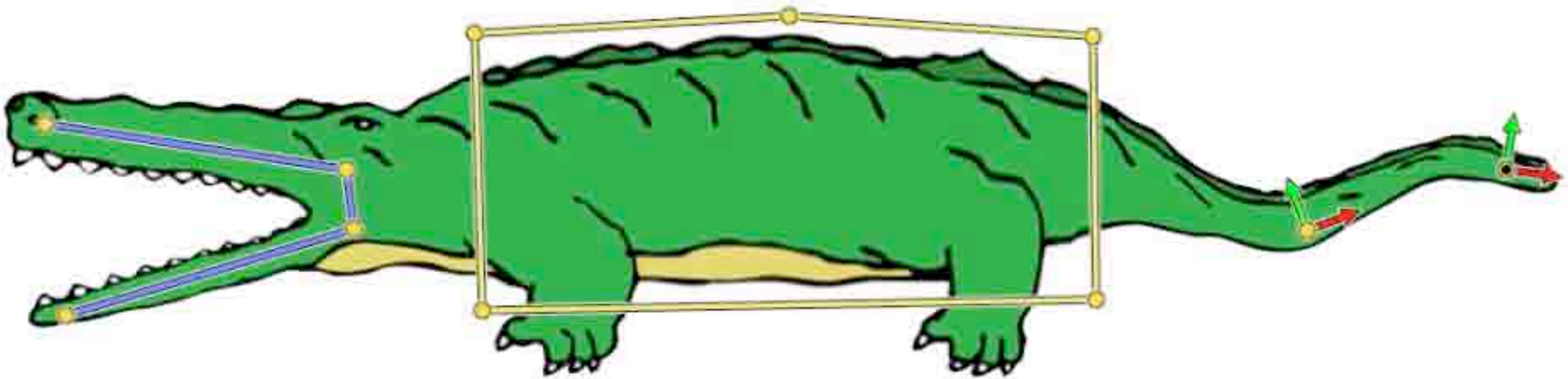
Each handle type has a specific task, more than just *different modeling metaphor*



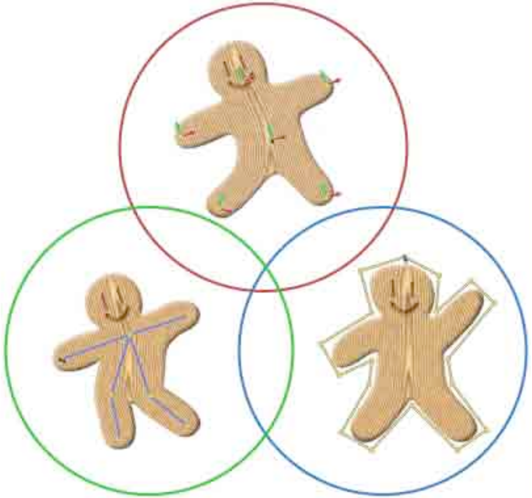
Each handle type has a specific task, more than just *different modeling metaphor*



Each handle type has a specific task, more than just *different modeling metaphor*

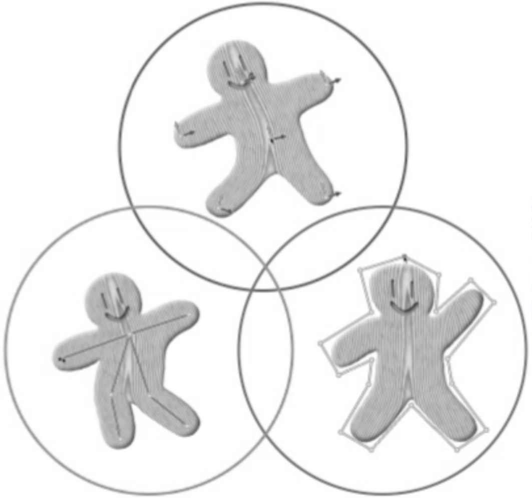


Real-time applications deserve ample interfaces and high-quality deformations



SGP 2010
SIGGRAPH 2011
SGP 2012

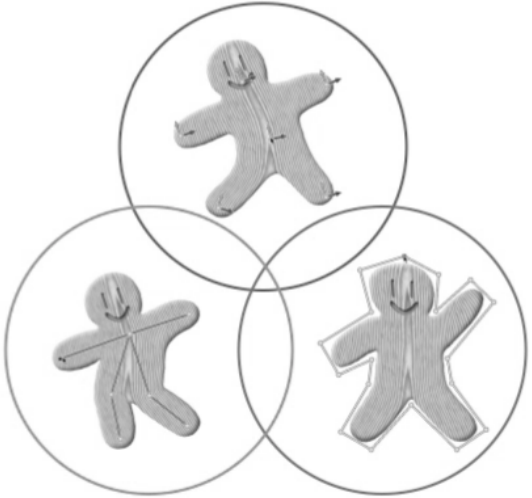
Real-time applications deserve ample interfaces and high-quality deformations

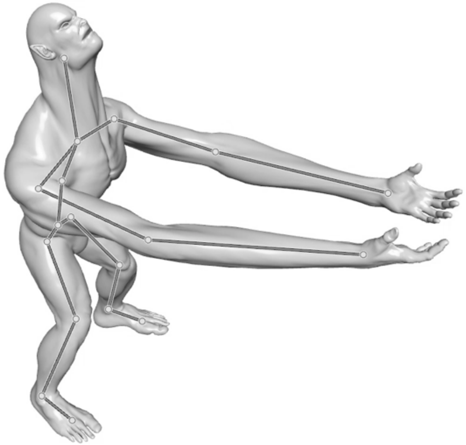


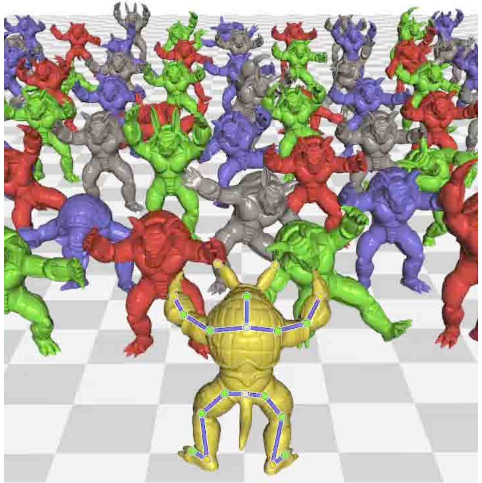
SGP 2010
SIGGRAPH 2011
SGP 2012

SIGGRAPH Asia 2011

Real-time applications deserve ample interfaces and high-quality deformations





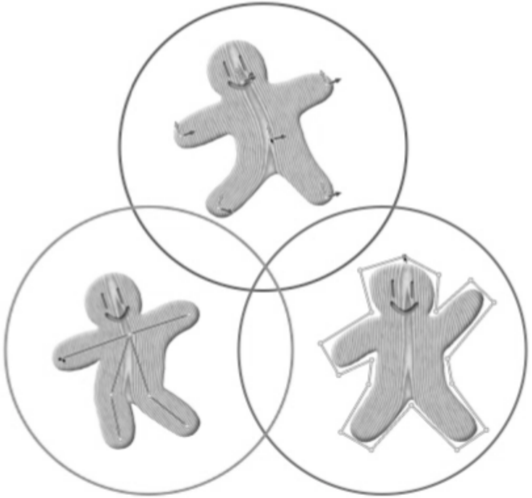


SGP 2010
SIGGRAPH 2011
SGP 2012

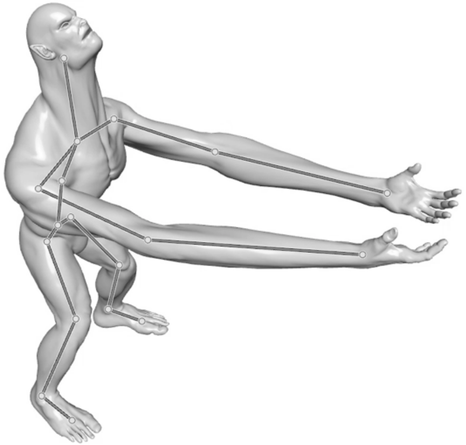
SIGGRAPH Asia 2011

SIGGRAPH 2012

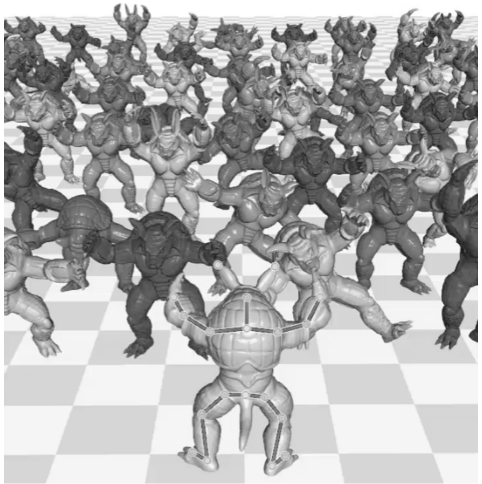
Real-time applications deserve ample interfaces and high-quality deformations



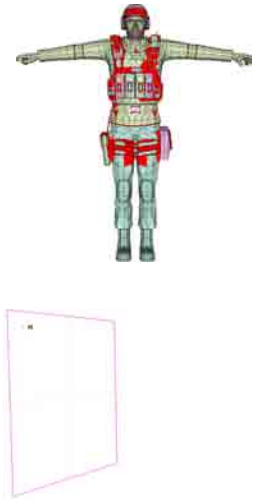
SGP 2010
SIGGRAPH 2011
SGP 2012



SIGGRAPH Asia 2011

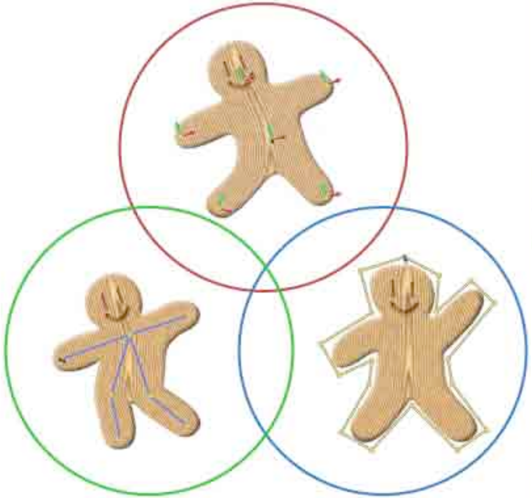


SIGGRAPH 2012

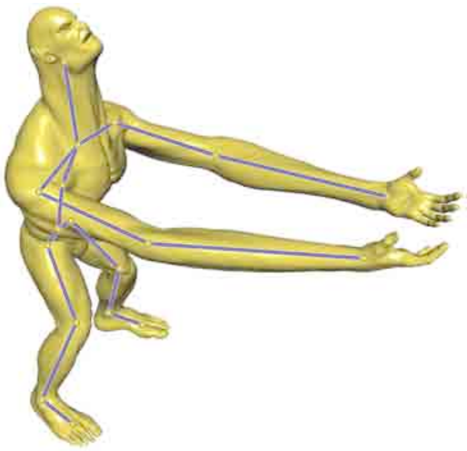


SIGGRAPH 2013

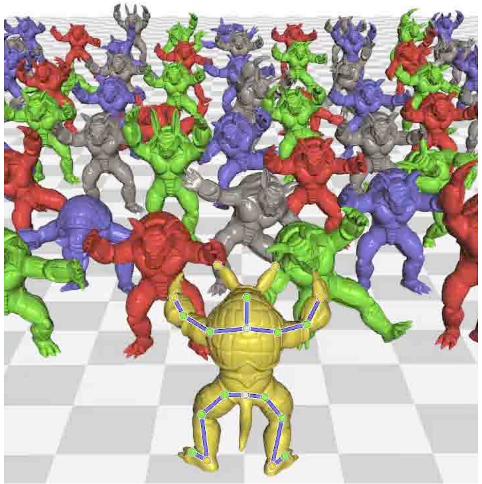
Real-time applications deserve ample interfaces and high-quality deformations



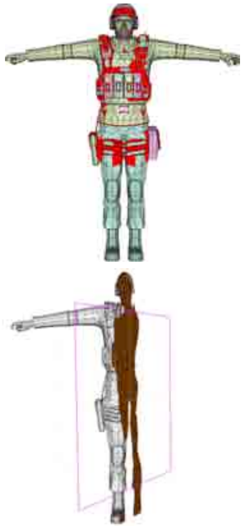
SGP 2010
SIGGRAPH 2011
SGP 2012



SIGGRAPH Asia 2011



SIGGRAPH 2012



SIGGRAPH 2013

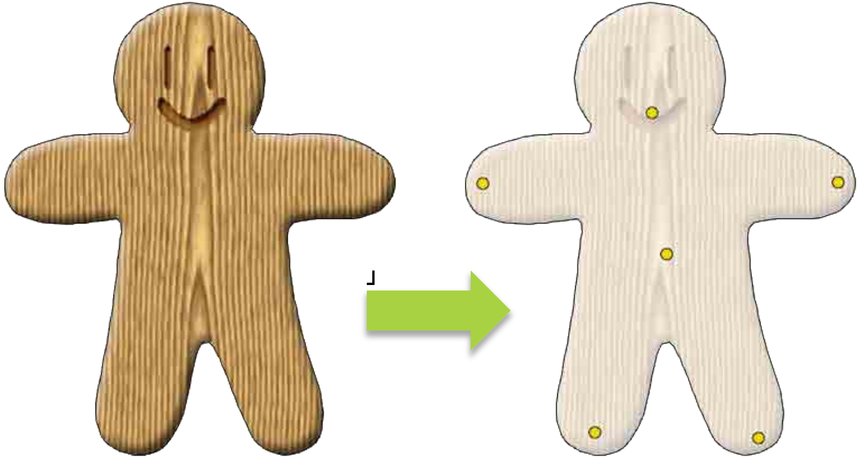
Linear Blend Skinning has known artifacts but makes up in real-time performance

load shape

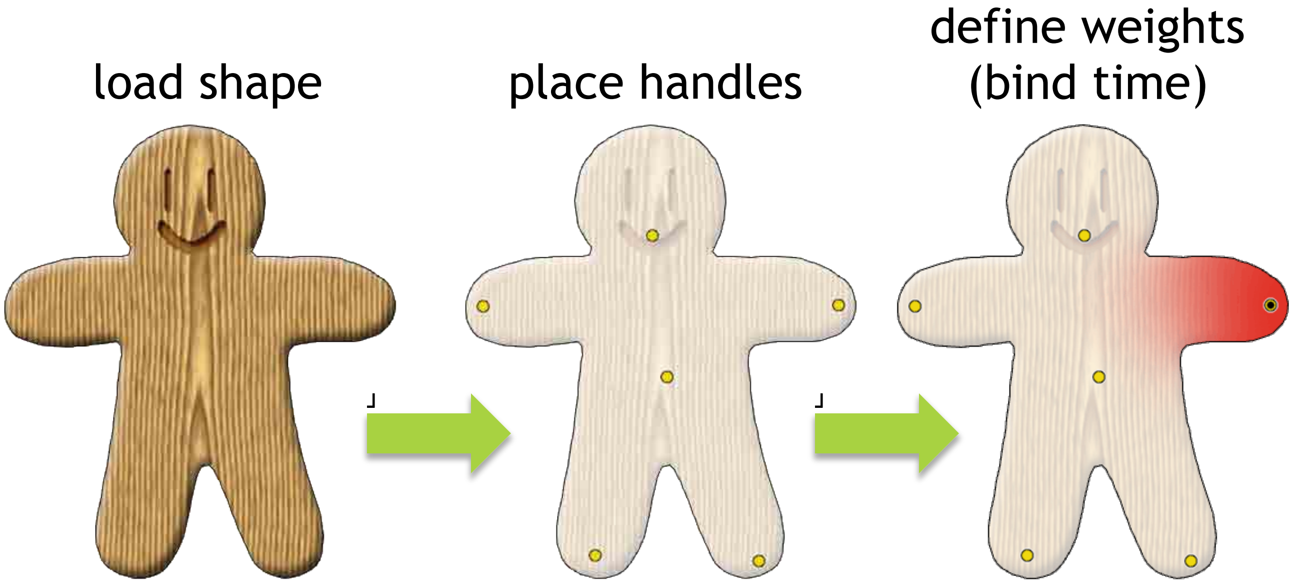
Linear Blend Skinning has known artifacts but makes up in real-time performance

load shape

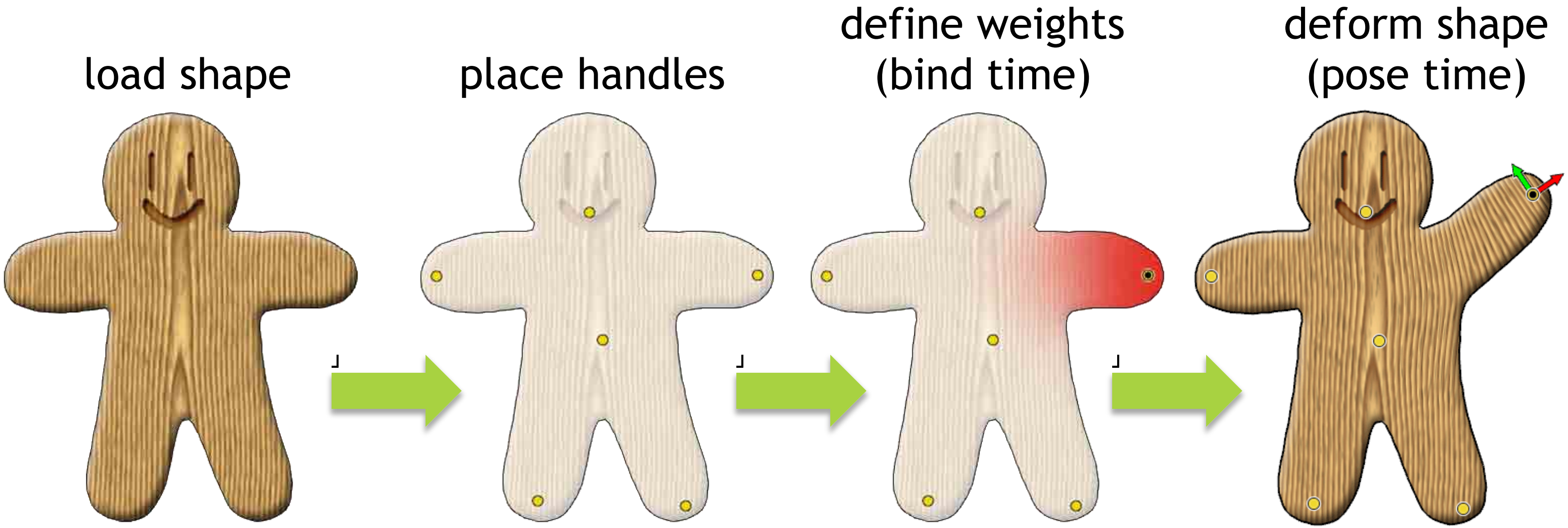
place handles



Linear Blend Skinning has known artifacts but makes up in real-time performance



Linear Blend Skinning has known artifacts but makes up in real-time performance



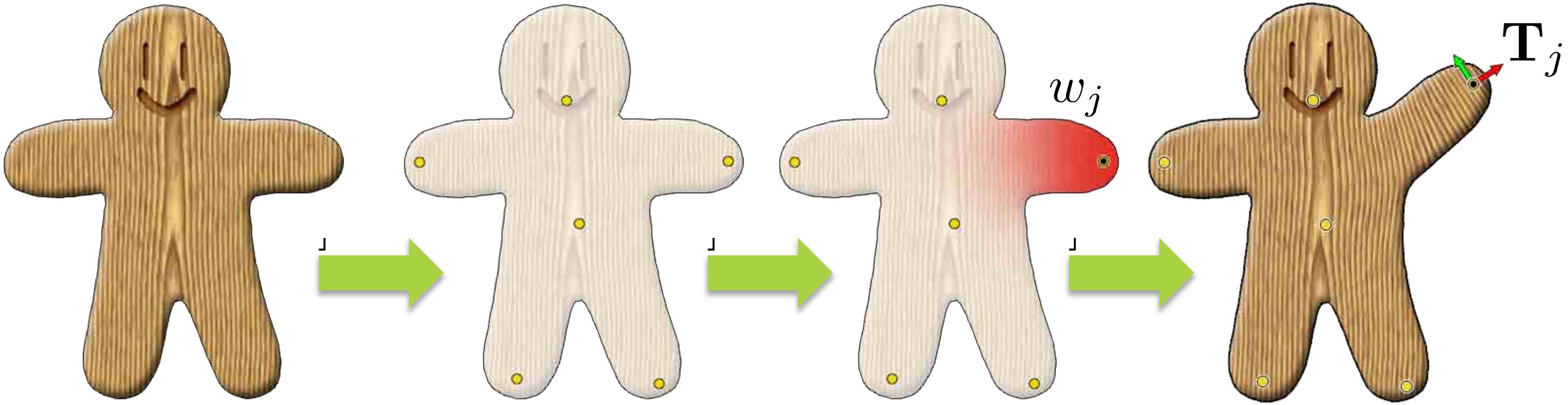
Linear Blend Skinning has known artifacts but makes up in real-time performance

$$\mathbf{v}'_i = \sum_{j=1}^{m} w_j(\mathbf{v}_i)\mathbf{T}_j \begin{pmatrix} \mathbf{v}_i \\ 1 \end{pmatrix}$$

load shape

place handles

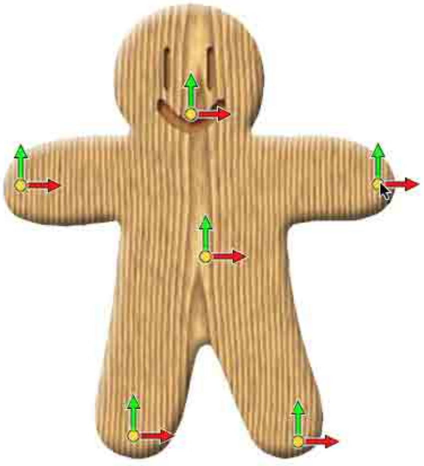
define weights (bind time)



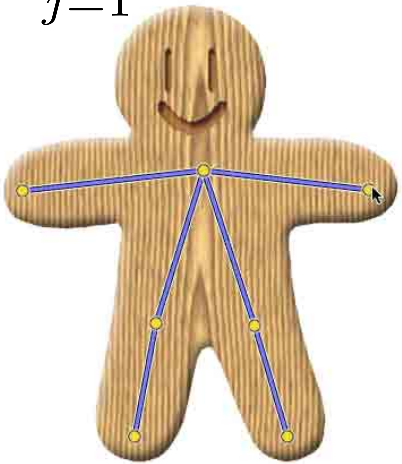
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Warping and deformation tasks require different user interactions

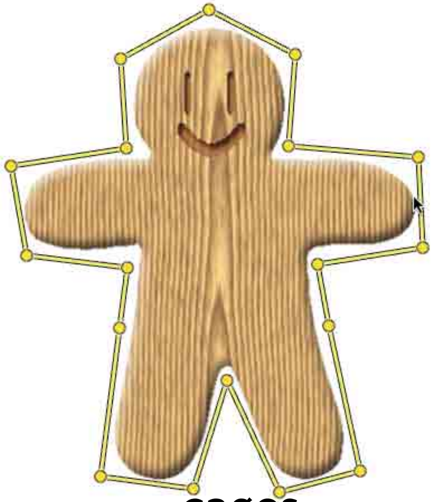
$$\mathbf{v}_i' = \sum_{j=1}^{m} w_j(\mathbf{v}_i)\mathbf{T}_j \begin{pmatrix} \mathbf{v}_i \\ 1 \end{pmatrix}$$



points

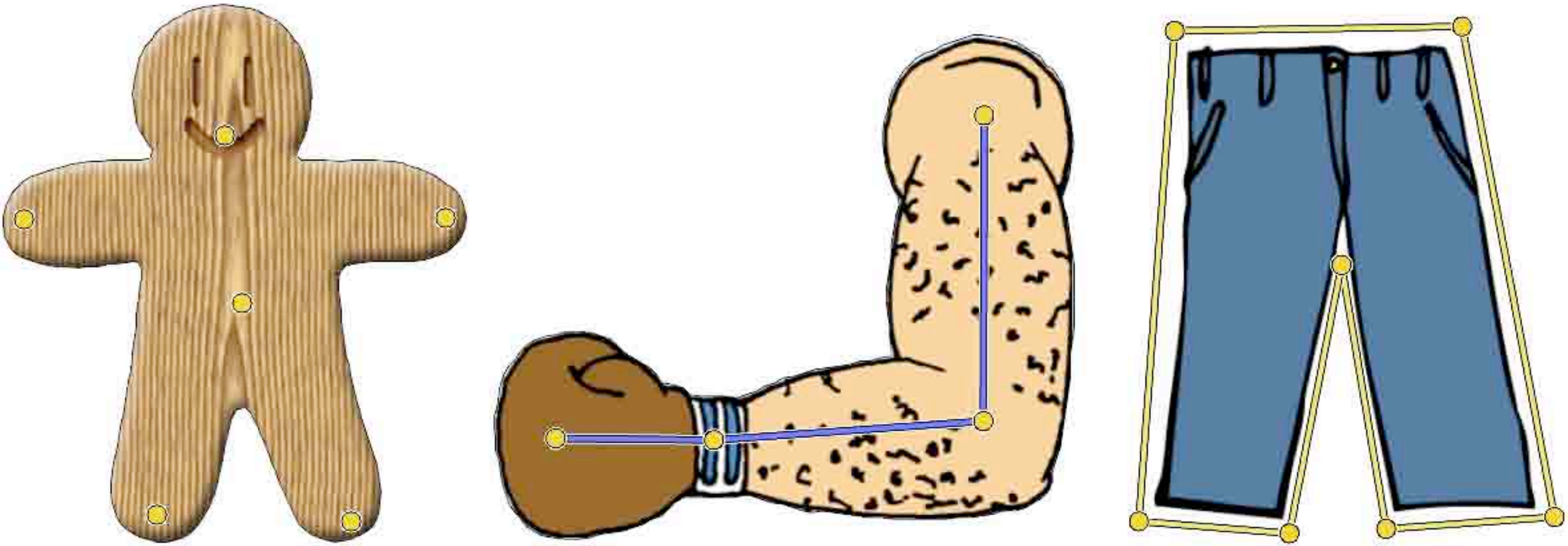


skeletons

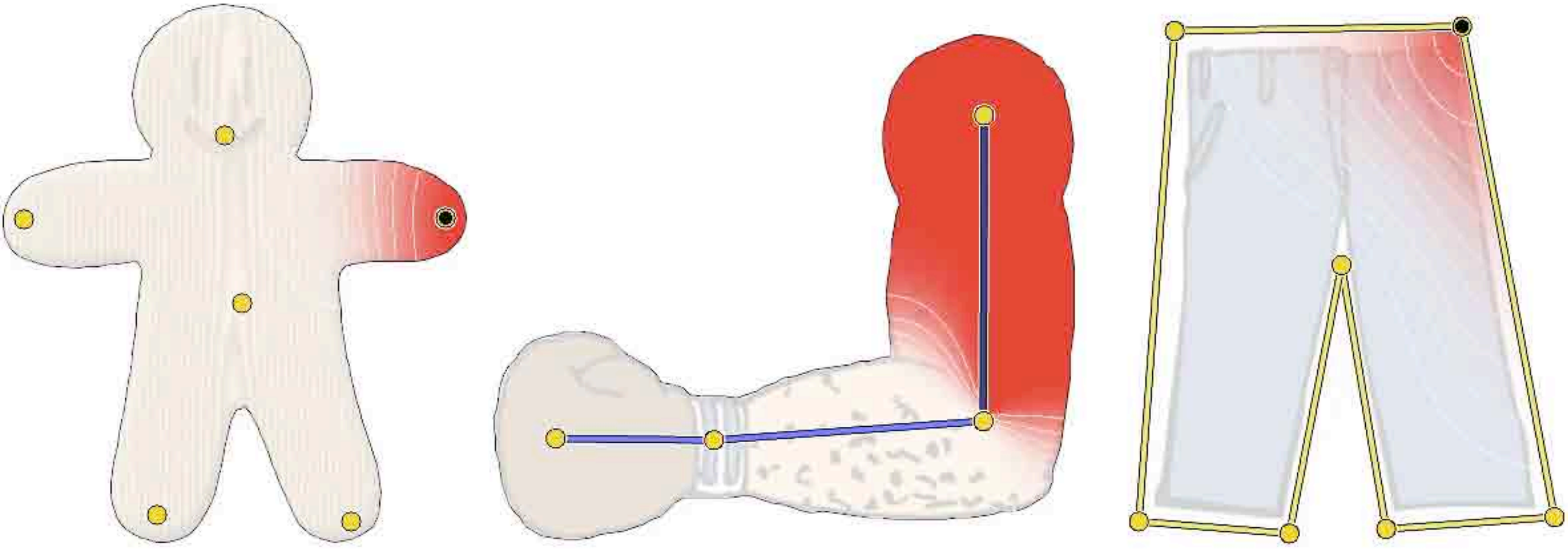


cages

Weights should be smooth, shape-aware, positive and *intuitive*



Weights should be smooth, shape-aware, positive and *intuitive*



Weights must be smooth everywhere, *especially* at handles

our method
[SIGGRAPH 2011]

extension of Harmonic Coordinates
[Joshi et al. 2005]

Weights must be smooth everywhere, *especially* at handles

our method
[SIGGRAPH 2011]

extension of Harmonic Coordinates
[Joshi et al. 2005]

Weights must be smooth everywhere, *especially* at handles

our method
[SIGGRAPH 2011]

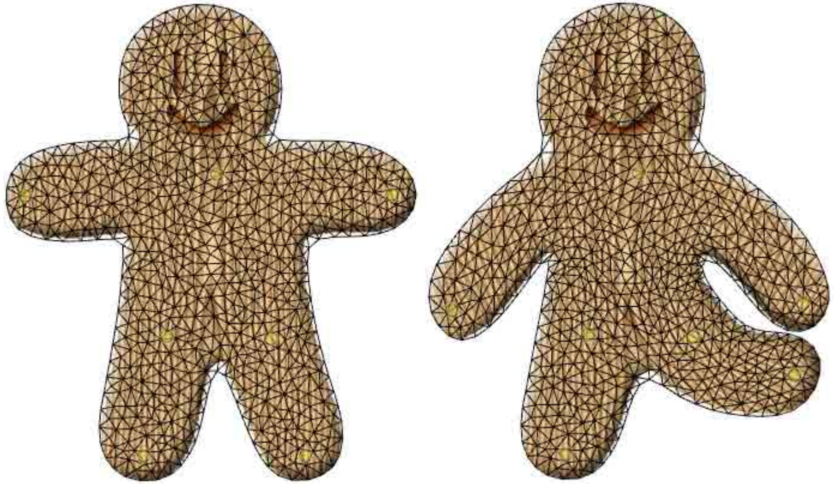
extension of Harmonic Coordinates
[Joshi et al. 2005]

Shape-awareness ensures respect of domain's features

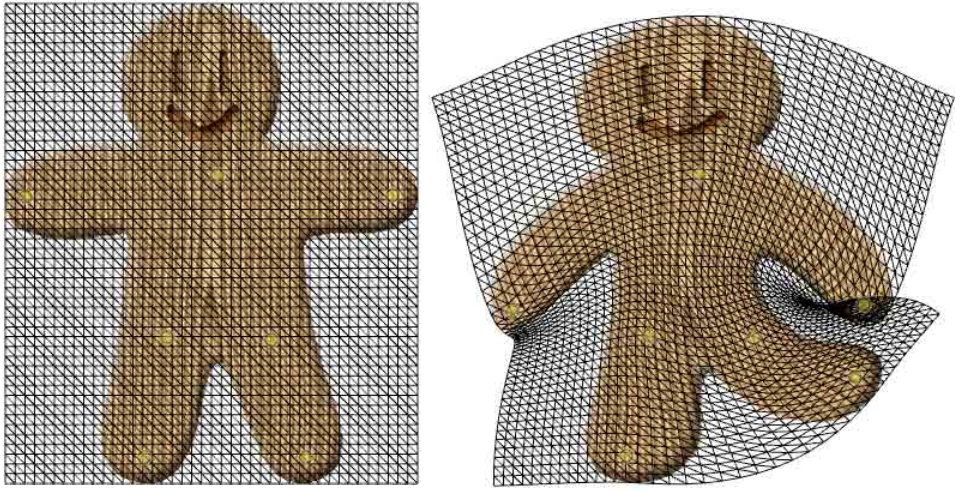
our method

non-shape-aware methods
e.g. [Schaefer et al. 2006]

Shape-awareness ensures respect of domain's features

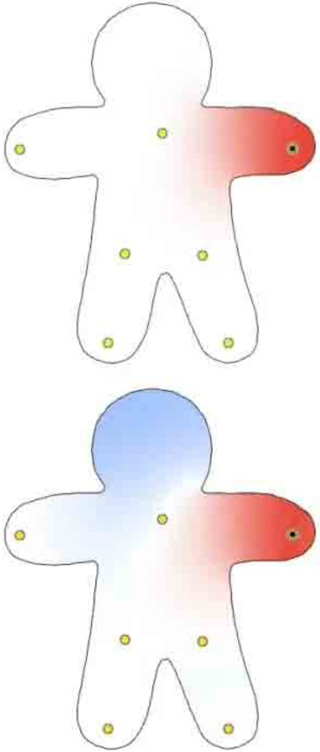


our method



non-shape-aware methods
e.g. [Schaefer et al. 2006]

Non-negative weights are mandatory

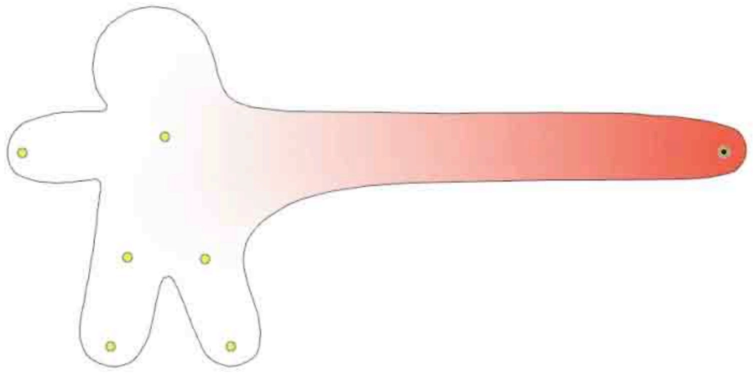


our method

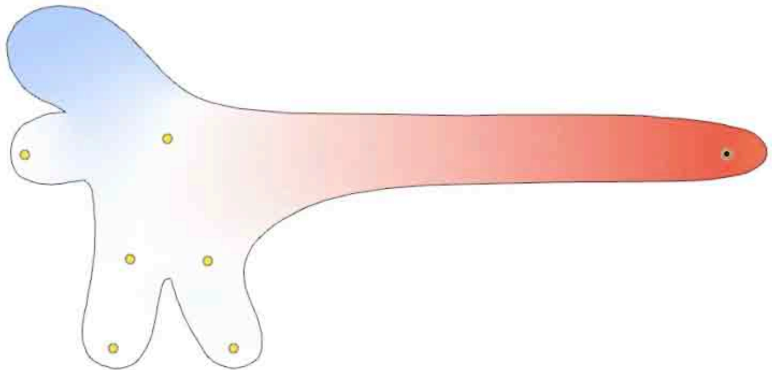
unconstrained biharmonic
[Botsch & Kobbelt 2004]

Non-negative weights are mandatory

our method



unconstrained biharmonic
[Botsch & Kobbelt 2004]

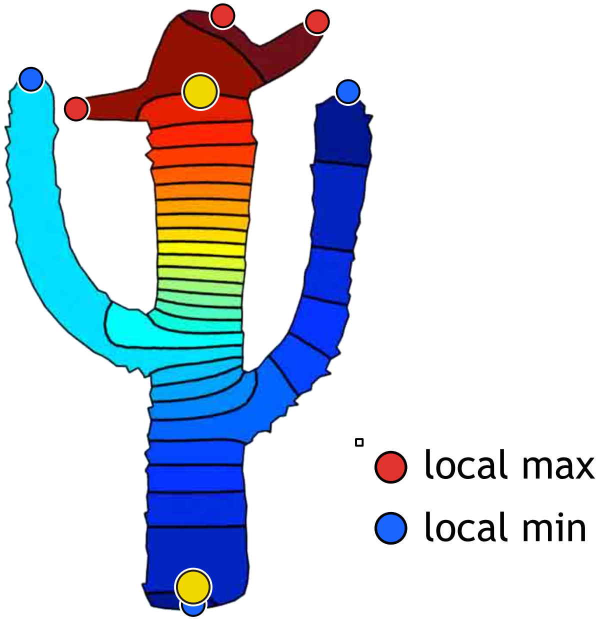


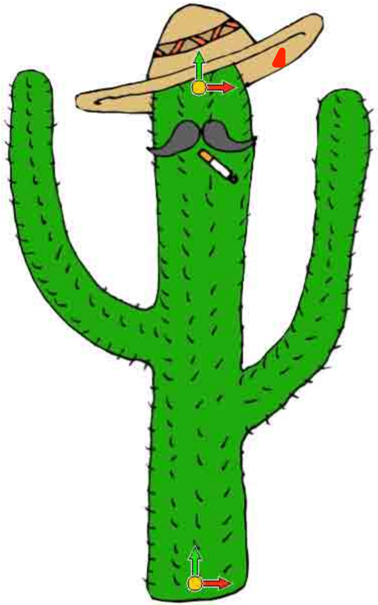
Spurious extrema cause distracting artifacts

unconstrained Δ^2

ext. of [Botsch & Kobbelt 2004]

$$\mathbf{v}_i' = \sum_{j=1}^{m} w_j(\mathbf{v}_i)\mathbf{T}_j \begin{pmatrix} \mathbf{v}_i \\ 1 \end{pmatrix}$$

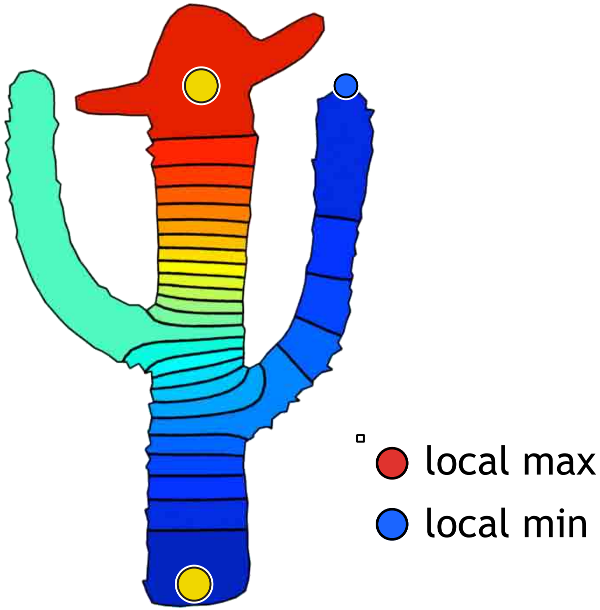


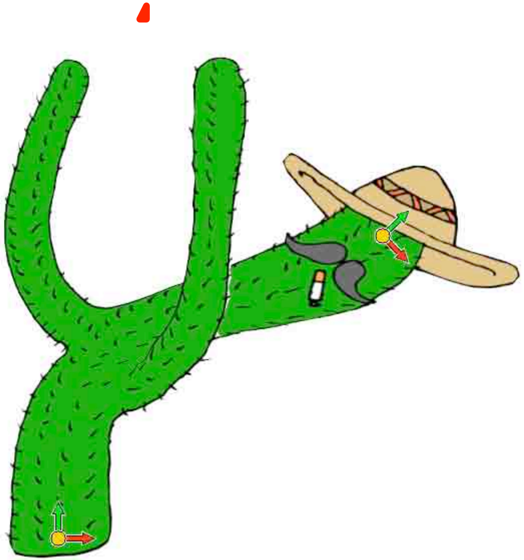


Spurious extrema cause distracting artifacts

bounded Δ^2
[SIGGRAPH 2011]

$$\mathbf{v}_i' = \sum_{j=1}^{m} w_j(\mathbf{v}_i)\mathbf{T}_j \begin{pmatrix} \mathbf{v}_i \\ 1 \end{pmatrix}$$

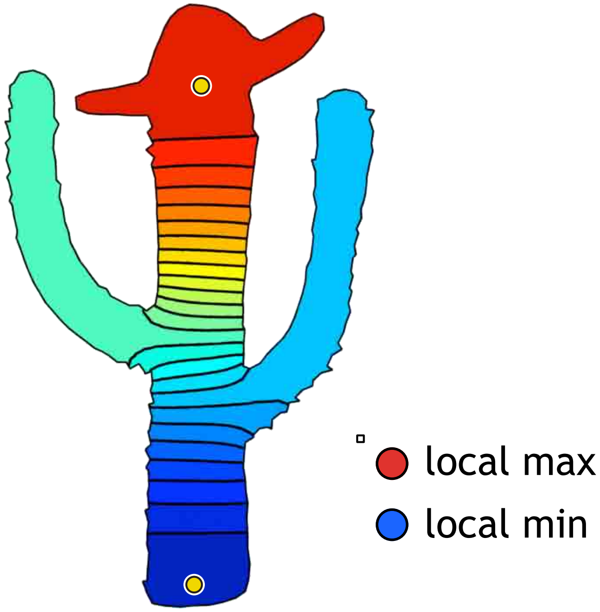


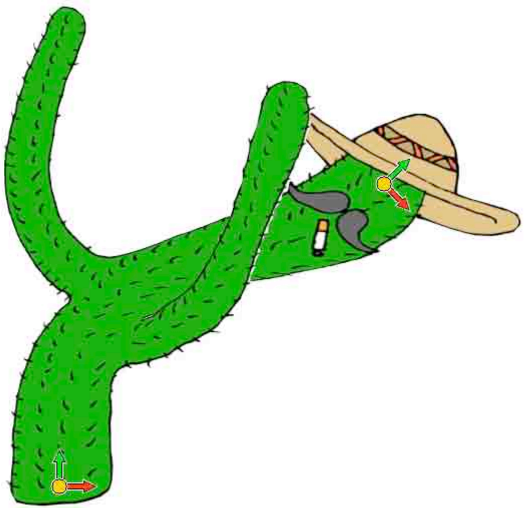


We explicitly prohibit spurious extrema

our improved Δ^2
[SGP 2012]

$$\mathbf{v}_i' = \sum_{j=1}^{m} w_j(\mathbf{v}_i)\mathbf{T}_j \begin{pmatrix} \mathbf{v}_i \\ 1 \end{pmatrix}$$





Constrained optimization ensures satisfaction of all properties

$$\underset{\mathbf{w}_j,\ j=1,\dots,m}{\operatorname{argmin}} \sum_{j=1}^{m} \|L\mathbf{w}_j\|^2$$

ext. of [Botsch & Kobbelt 2004]

+ shape-aware

Constrained optimization ensures satisfaction of all properties

$$\underset{w_j,\ j=1,\ldots,m}{\operatorname{argmin}} \sum_{j=1}^{m} \int_{\Omega} (\Delta w_j)^2 \, dV$$

[SGP 2010]

+ shape-aware
+ smoothness
+ mesh independence

Constrained optimization ensures satisfaction of all properties

$$\underset{w_j,\ j=1,\ldots,m}{\operatorname{argmin}} \sum_{j=1}^{m} \int_{\Omega} (\Delta w_j)^2 \, dV$$

$$0 \leq w_j \leq 1, \ j = 1, \ldots, m$$

[SIGGRAPH 2011]

+ shape-aware
+ smoothness
+ mesh independence
+ non-negativity
+ locality
+ arbitrary handles

Constrained optimization ensures satisfaction of all properties

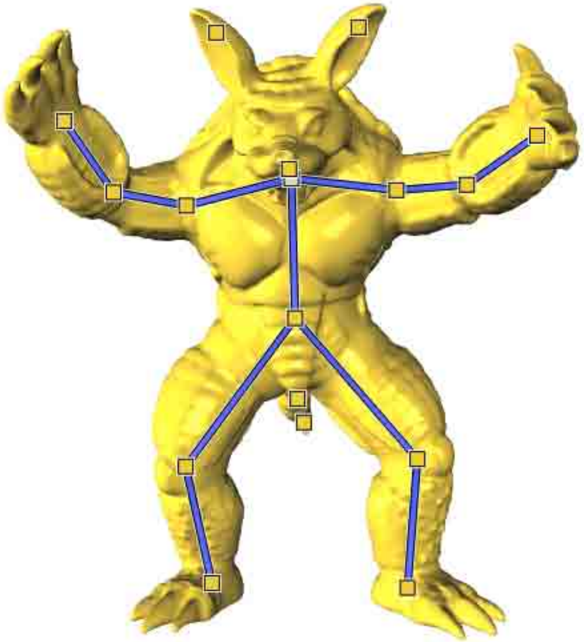
$$\underset{w_j,\ j=1,\ldots,m}{\operatorname{argmin}} \sum_{j=1}^{m} \int_{\Omega} (\Delta w_j)^2 \, dV$$

$$\nabla w_j \cdot \nabla u_j > 0,\ j = 1, \ldots, m$$

[SGP 2012]

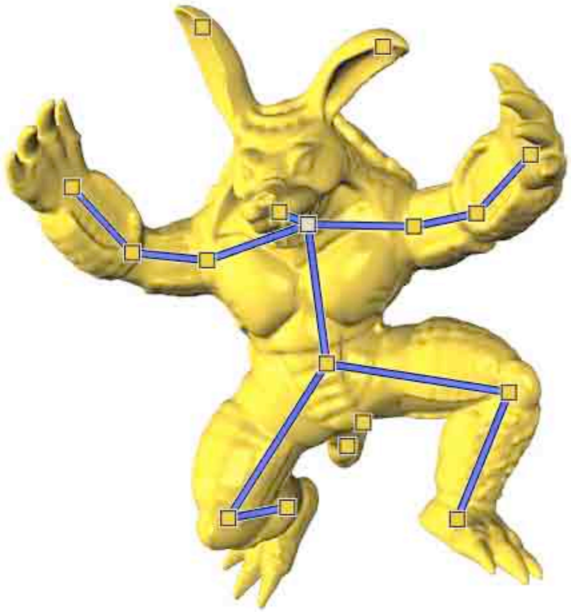
+ shape-aware
+ smoothness
+ mesh independence
+ non-negativity
+ locality
+ arbitrary handles
+ monotonicity

Weights in 3D retain nice properties



Demo

Weights in 3D retain nice properties



Demo

Variational formulation allows additional, problem-specific constraints

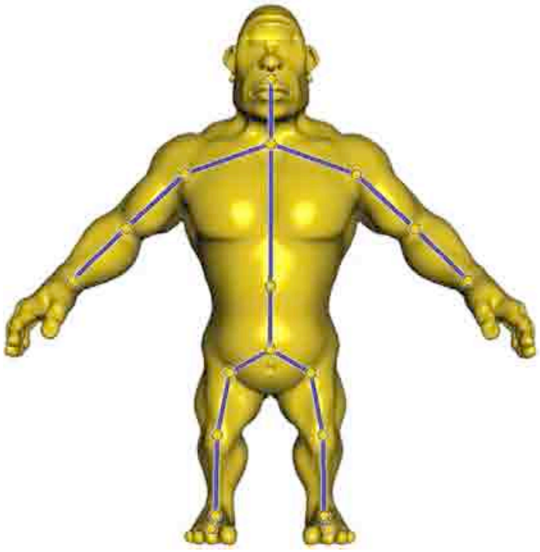
Variational formulation allows additional, problem-specific constraints

Linear blending subspace is *still* too small

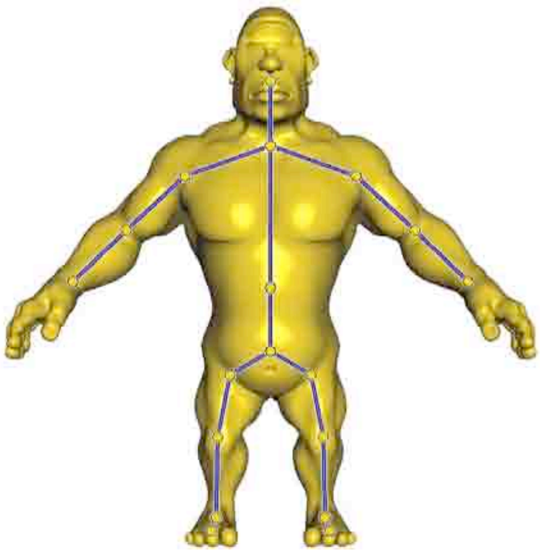
$$\mathbf{v}_i' = \sum_{j=1}^{m} w_j(\mathbf{v}_i)\mathbf{T}_j \begin{pmatrix}\mathbf{v}_i \\ 1\end{pmatrix}$$

Linear blend skinning etc. are not rich enough to stretch and twist along bones

LBS

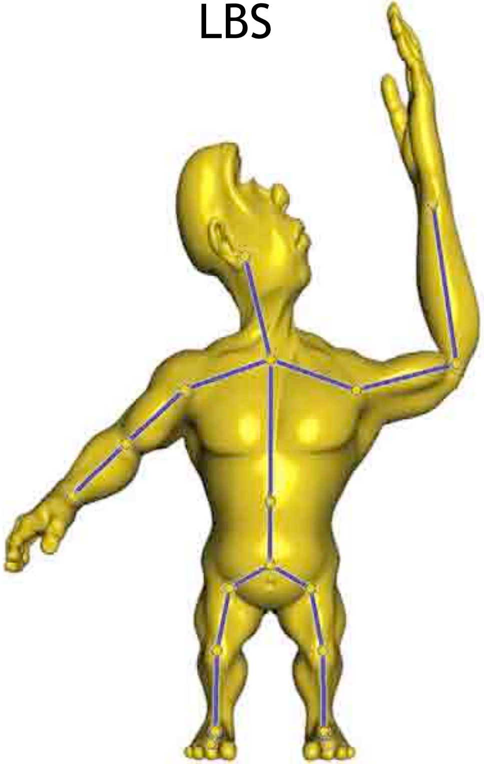


our extension
[SIGGRAPH Asia 2011]

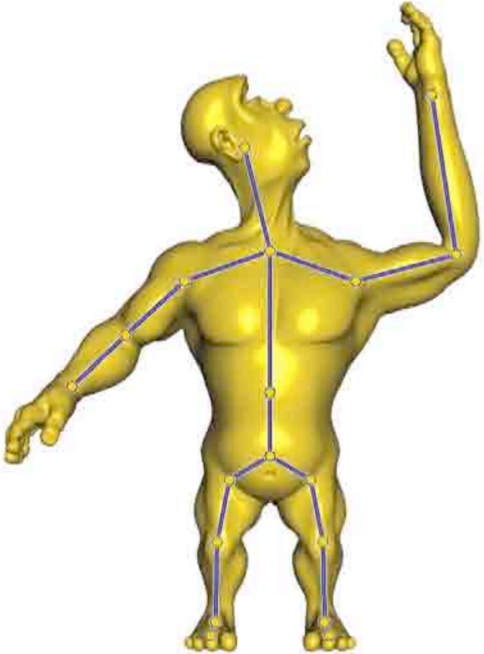


Linear blend skinning etc. are not rich enough to stretch and twist along bones

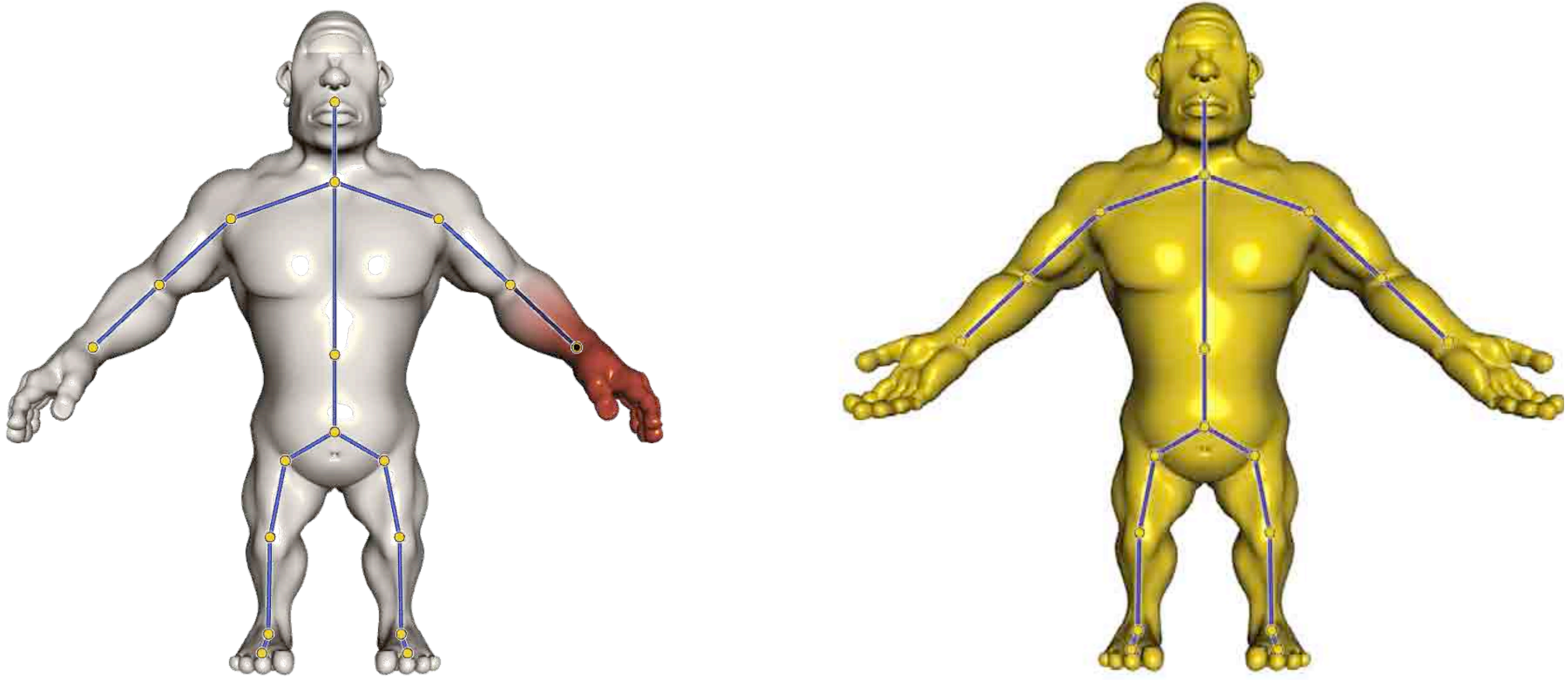
LBS



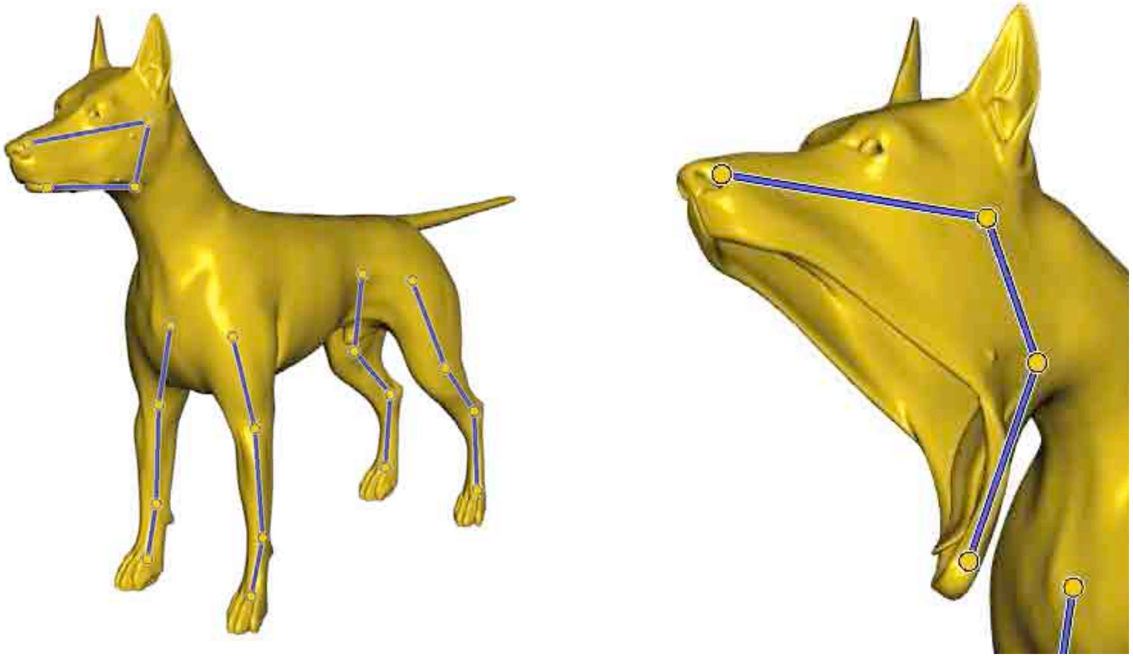
our extension
[SIGGRAPH Asia 2011]



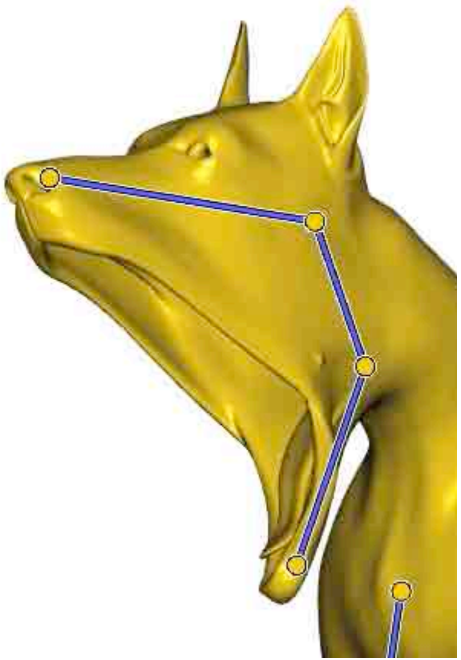
Second set of our weights expands skinning subspace



Stretching facilitates exaggeration, a basic principle of life-like animation

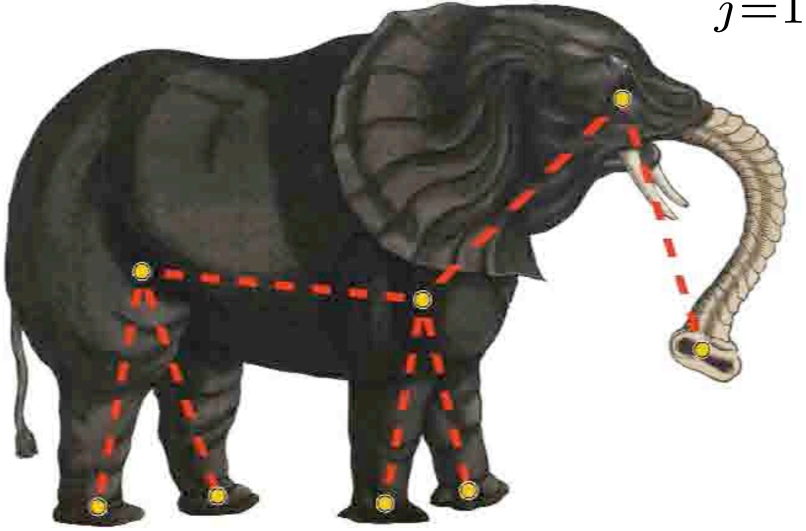


Stretching facilitates exaggeration, a basic principle of life-like animation



Good weights alone aren't enough to guarantee intuitive results

$$\mathbf{v}_i' = \sum_{j=1}^{m} w_j(\mathbf{v}_i)\boxed{\mathbf{T}_j}\begin{pmatrix}\mathbf{v}_i\\1\end{pmatrix}$$

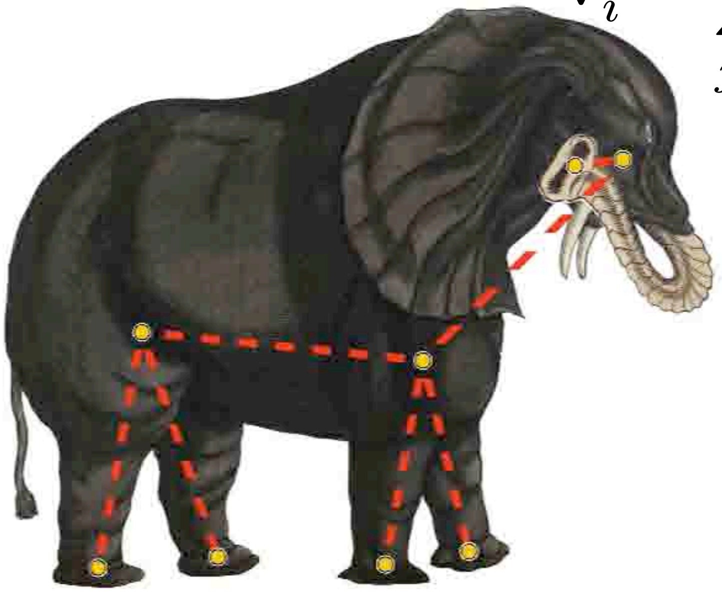


pseudo-edges [SIGGRAPH 2011]

our improved method [SIGGRAPH 2012]

Good weights alone aren’t enough to guarantee intuitive results

$$\mathbf{v}_i' = \sum_{j=1}^{m} w_j(\mathbf{v}_i)\mathbf{T}_j \begin{pmatrix} \mathbf{v}_i \\ 1 \end{pmatrix}$$

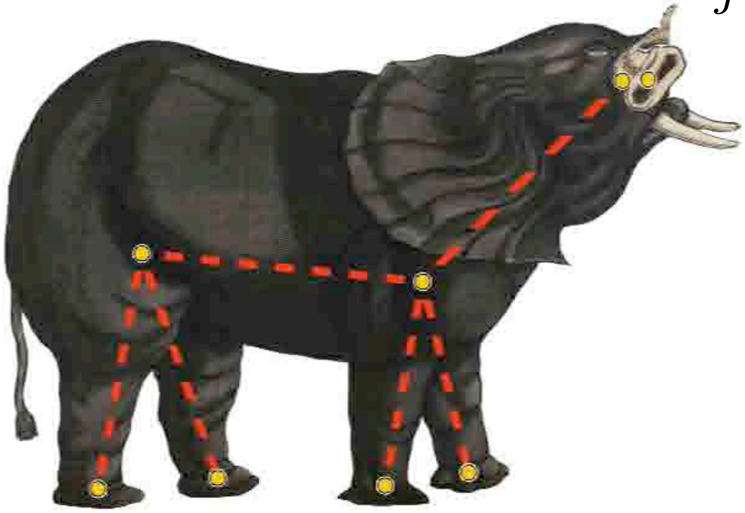


pseudo-edges [SIGGRAPH 2011]

our improved method [SIGGRAPH 2012]

Good weights alone aren't enough to guarantee intuitive results

$$\mathbf{v}_i' = \sum_{j=1}^{m} w_j(\mathbf{v}_i)\,\mathbf{T}_j \begin{pmatrix} \mathbf{v}_i \\ 1 \end{pmatrix}$$

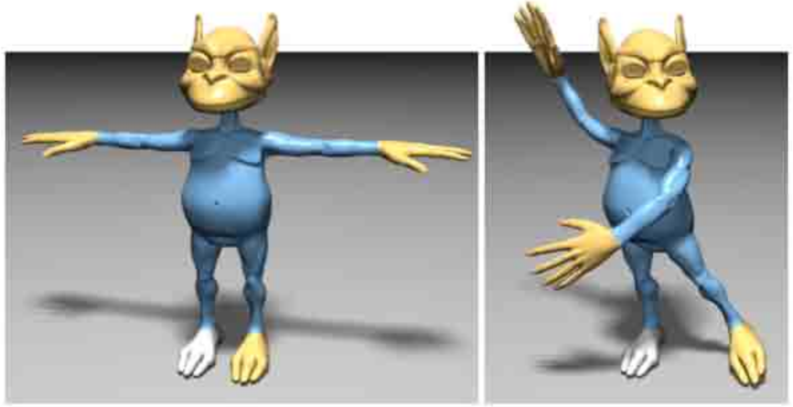


pseudo-edges [SIGGRAPH 2011]

our improved method [SIGGRAPH 2012]

Many previous techniques provide quality, but not speed

high-quality solutions to nonlinear elasticity energy minimizations: ~*seconds*
e.g. [Botsch et al. 2006]

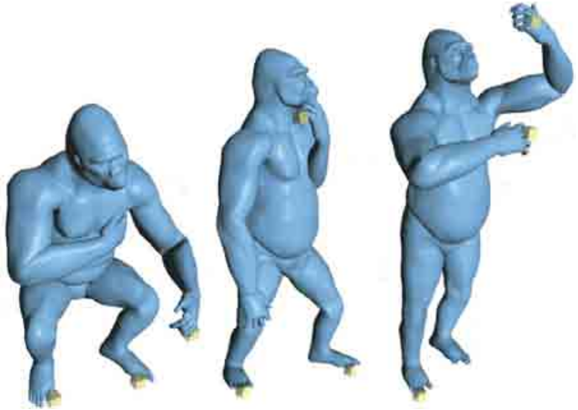


physically accurate muscle systems require off-line simulation
e.g. [Teran et al. 2005]

Image courtesy Joseph Teran

Other reduced models employ skinning, but still too slow

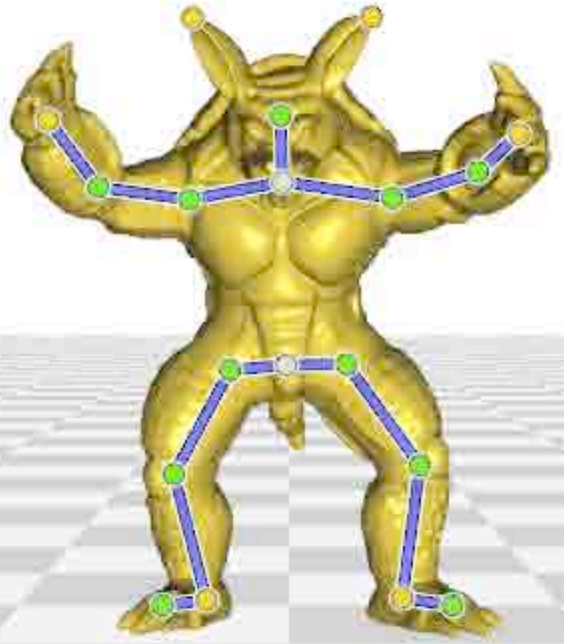
needs examples, choice of energy complicates per-frame computation
~milliseconds
e.g. [Der et al. 2006]



significant performance tuning, but grid determines complexity
~milliseconds
e.g. [McAdams et al. 2011]

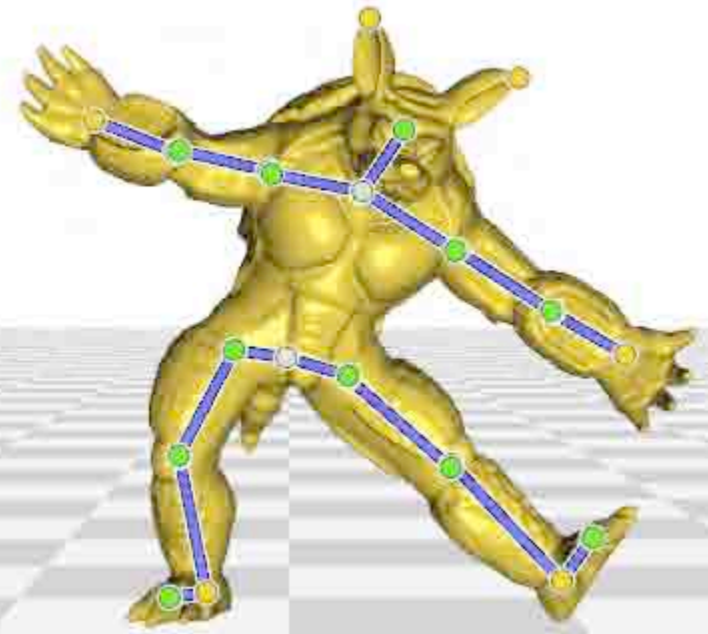
We achieve speeds measured in microseconds

80k triangles
20μs per iteration



We achieve speeds measured in microseconds

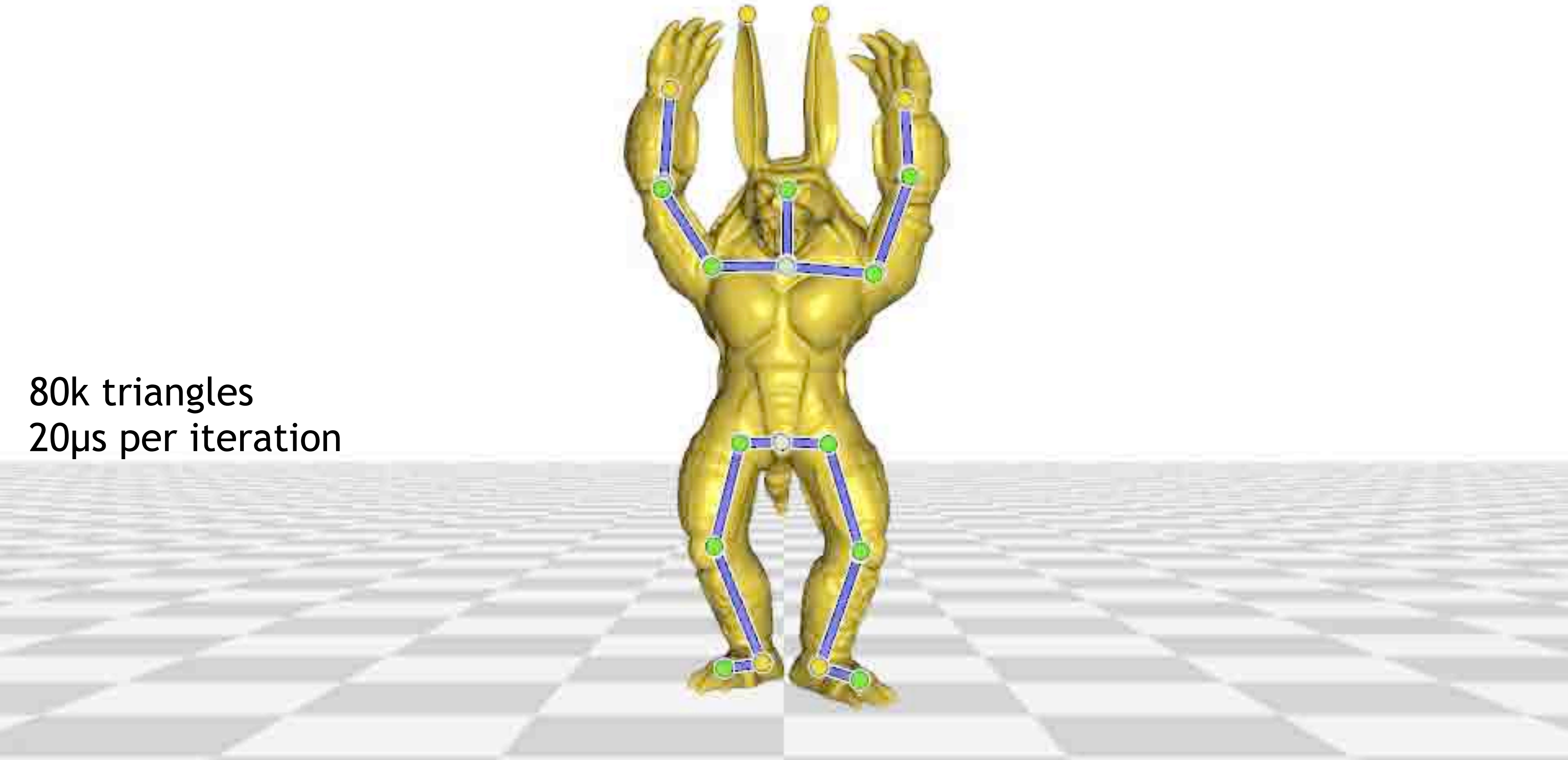
80k triangles
20μs per iteration



We achieve speeds measured in microseconds

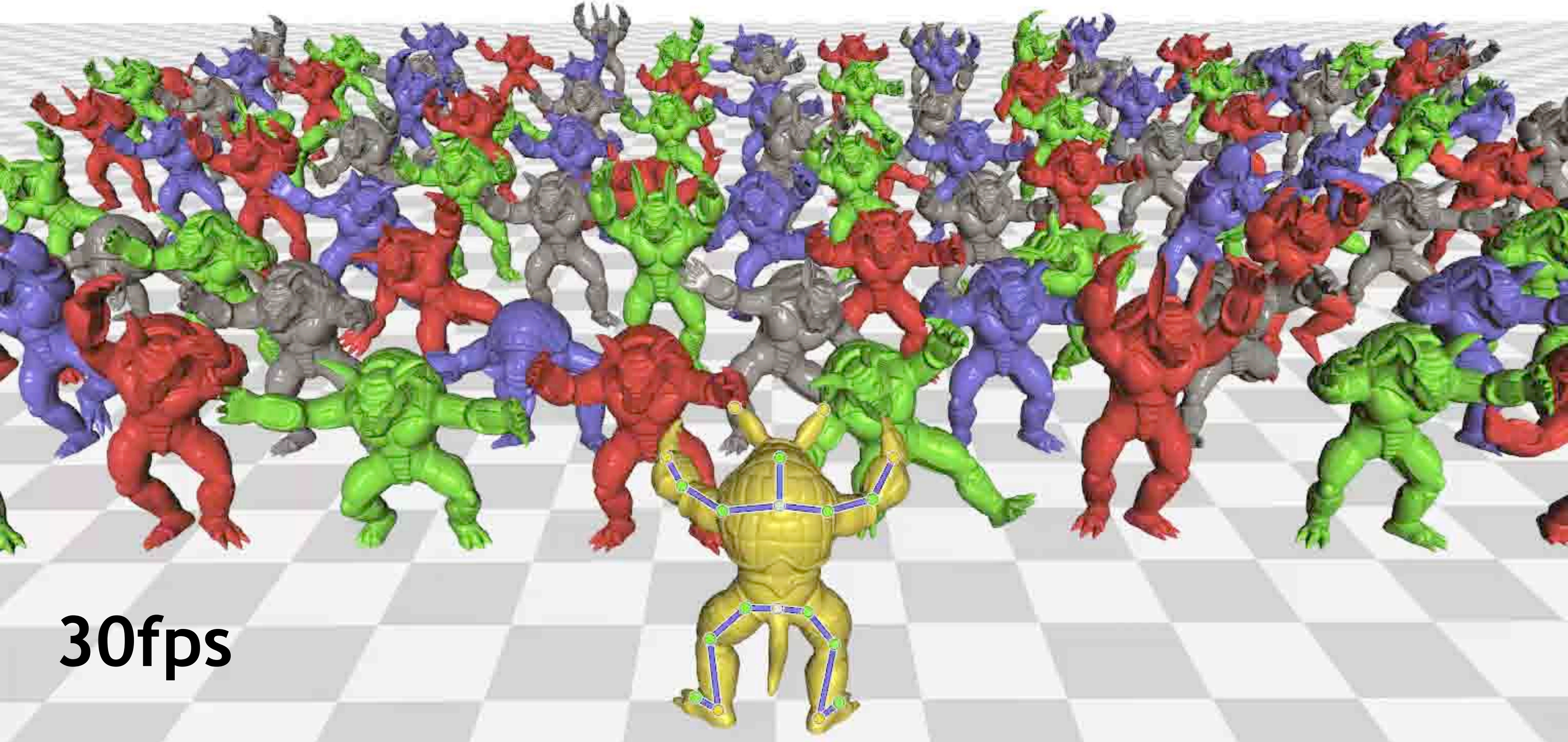
80k triangles
20μs per iteration

We achieve speeds measured in microseconds



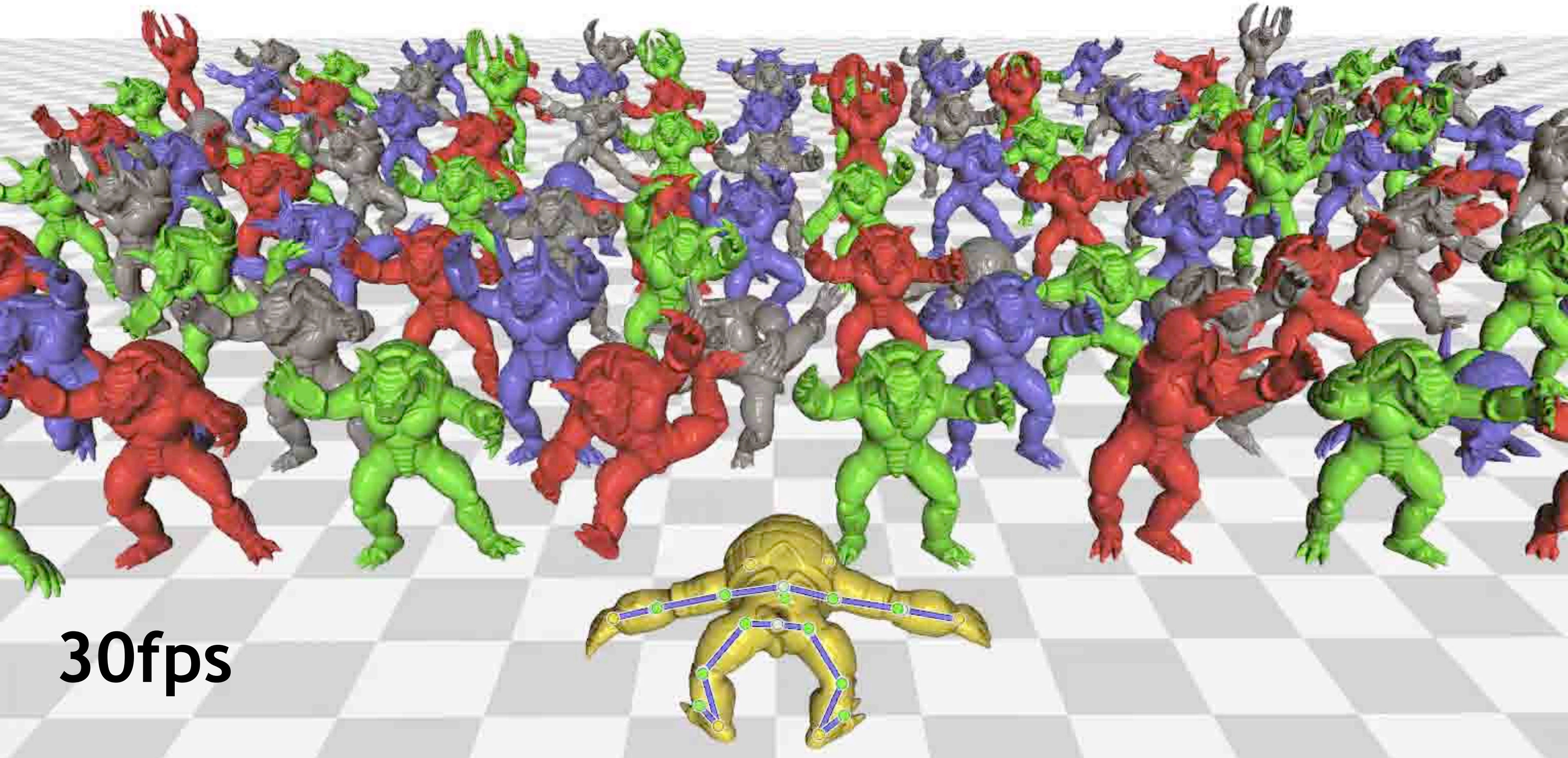
80k triangles
20μs per iteration

This means speed comparable to rendering



30fps

This means speed comparable to rendering



30fps

User specifies subset of parameters, optimize to find remaining ones

full optimization

$$\underset{\mathbf{V}'}{\arg\min}\ E(\mathbf{V}')$$

mesh vertex positions

User specifies subset of parameters, optimize to find remaining ones

full optimization

$$\operatorname*{arg\,min}_{\mathbf{V}'} \; E(\mathbf{V}')$$

reduced model

$$\mathbf{v}'_i = \sum_{j=1}^{m} w_j(\mathbf{v}_i)\mathbf{T}_j \begin{pmatrix} \mathbf{v}_i \\ 1 \end{pmatrix}$$

skinning degrees of freedom

User specifies subset of parameters, optimize to find remaining ones

full optimization

$$\underset{\mathbf{V}'}{\arg\min}\ E(\mathbf{V}')$$

reduced model

$$\mathbf{v}'_i = \sum_{j=1}^{m} w_j(\mathbf{v}_i)\mathbf{T}_j \begin{pmatrix} \mathbf{v}_i \\ 1 \end{pmatrix}$$

matrix form

$$\mathbf{V}' = \mathbf{M}\mathbf{T}$$

User specifies subset of parameters, optimize to find remaining ones

full optimization

$$\arg\min_{\mathbf{V}'} E(\mathbf{V}')$$

reduced model

$$\mathbf{v}'_i = \sum_{j=1}^{m} w_j(\mathbf{v}_i)\mathbf{T}_j \begin{pmatrix} \mathbf{v}_i \\ 1 \end{pmatrix}$$

matrix form

$$\mathbf{V}' = \mathbf{M}\mathbf{T}$$

reduced optimization

$$\arg\min_{\mathbf{T}} E(\mathbf{M}\mathbf{T})$$

Enforce user constraints as linear equalities

reduced optimization

$$\arg\min_{\mathbf{T}} \; E(\mathbf{MT})$$

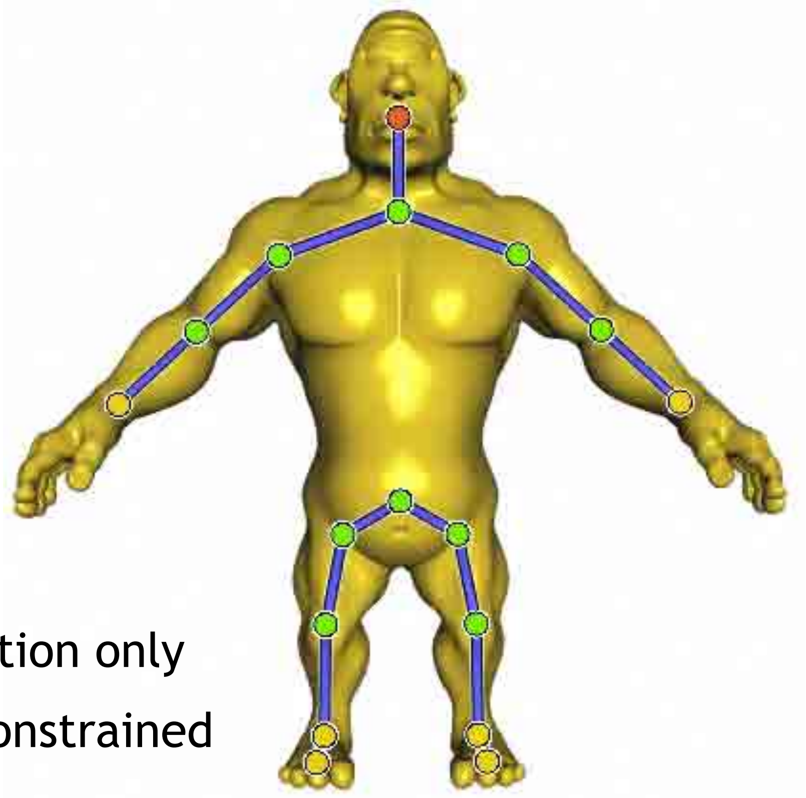
user constraints

$$\underbrace{\begin{bmatrix} \mathbf{I}_{\text{full}} \\ \mathbf{M}_{\text{pos}} \end{bmatrix}}_{\mathbf{M}_{\text{eq}}} \mathbf{T} = \underbrace{\begin{bmatrix} \mathbf{T}_{\text{full}} \\ \mathbf{P}_{\text{pos}} \end{bmatrix}}_{\mathbf{P}_{\text{eq}}}$$

full

position only

unconstrained



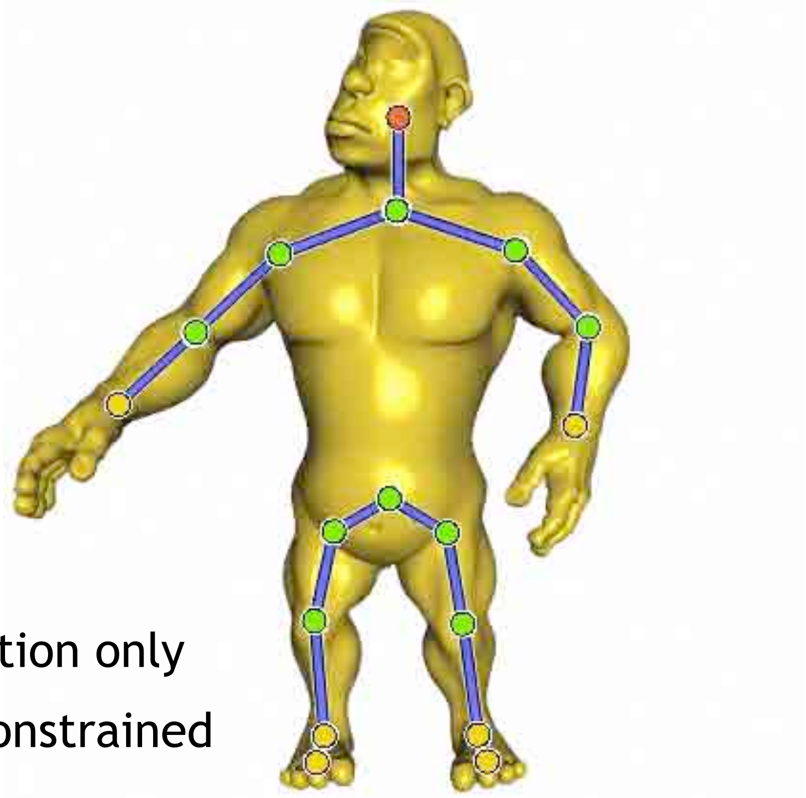
Enforce user constraints as linear equalities

reduced optimization

$$\underset{\mathbf{T}}{\arg\min}\ E(\mathbf{MT})$$

user constraints

$$\underbrace{\begin{bmatrix} \mathbf{I}_{\text{full}} \\ \mathbf{M}_{\text{pos}} \end{bmatrix}}_{\mathbf{M}_{\text{eq}}} \mathbf{T} = \underbrace{\begin{bmatrix} \mathbf{T}_{\text{full}} \\ \mathbf{P}_{\text{pos}} \end{bmatrix}}_{\mathbf{P}_{\text{eq}}}$$



full

position only

unconstrained

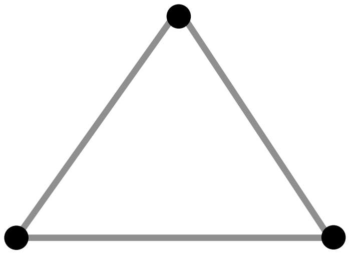
We reduce any *as-rigid-as-possible* energy

full energies $$E(\mathbf{V}', \mathbf{R}) = \frac{1}{2} \sum_{k=1}^{r} \sum_{(i,j) \in \mathcal{E}_k} c_{ijk} \left\| (\mathbf{v}'_i - \mathbf{v}'_j) - \mathbf{R}_k (\mathbf{v}_i - \mathbf{v}_j) \right\|^2$$

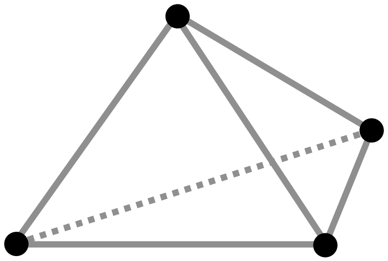
We reduce any *as-rigid-as-possible* energy

full energies

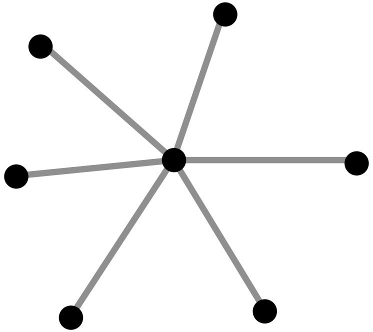
$$E(\mathbf{V}', \mathbf{R}) = \frac{1}{2} \sum_{k=1}^{r} \sum_{(i,j)\in\mathcal{E}_k} c_{ijk} \|(\mathbf{v}'_i - \mathbf{v}'_j) - \mathbf{R}_k(\mathbf{v}_i - \mathbf{v}_j)\|^2$$



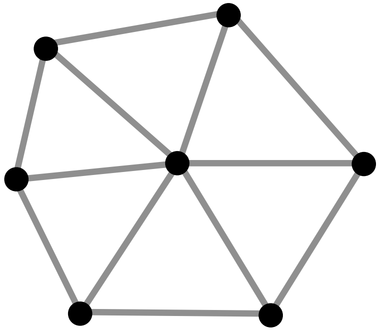
triangles
[Liu et al. 2008]



tetrahedra
[Chao et al. 2010]



“spokes”
[Sorkine & Alexa 2007]



“spokes and rims”
[Chao et al. 2010]

We reduce any *as-rigid-as-possible* energy

full energies $$E(\mathbf{V}', \mathbf{R}) = \frac{1}{2} \sum_{k=1}^{r} \sum_{(i,j)\in\mathcal{E}_k} c_{ijk} \|(\mathbf{v}'_i - \mathbf{v}'_j) - \mathbf{R}_k(\mathbf{v}_i - \mathbf{v}_j)\|^2$$

local/global optimization

global step: fix $\mathbf{R}$, minimize with respect to $\mathbf{V}'$

local step: fix $\mathbf{V}'$, minimize with respect to $\mathbf{R}$

We reduce any *as-rigid-as-possible* energy

full energies $$E(\mathbf{V}',\mathbf{R}) = \frac{1}{2}\sum_{k=1}^{r}\sum_{(i,j)\in\mathcal{E}_k} c_{ijk}\|(\mathbf{v}'_i - \mathbf{v}'_j) - \mathbf{R}_k(\mathbf{v}_i - \mathbf{v}_j)\|^2$$

local/global optimization

precompute

global step: large, sparse linear solve $\mathbf{V}' = \mathbf{A}^{-1}\mathbf{b}$

local step: fix $\mathbf{V}'$, minimize with respect to $\mathbf{R}$

We reduce any *as-rigid-as-possible* energy

full energies $$E(\mathbf{V}',\mathbf{R}) = \frac{1}{2}\sum_{k=1}^{r}\sum_{(i,j)\in\mathcal{E}_k} c_{ijk}\|(\mathbf{v}'_i-\mathbf{v}'_j)-\mathbf{R}_k(\mathbf{v}_i-\mathbf{v}_j)\|^2$$

local/global optimization

global step: large, sparse linear solve $\mathbf{V}' = \mathbf{A}^{-1}\mathbf{b}$

local step: 3x3 SVD for each rotation in $\mathbf{R}$

We reduce any *as-rigid-as-possible* energy

full energies $$E(\mathbf{V}',\mathbf{R}) = \frac{1}{2}\sum_{k=1}^{r}\sum_{(i,j)\in\mathcal{E}_k} c_{ijk}\|(\mathbf{v}'_i - \mathbf{v}'_j) - \mathbf{R}_k(\mathbf{v}_i - \mathbf{v}_j)\|^2$$

local/global optimization

precompute

global step: small, dense linear solve $\mathbf{T} = \tilde{\mathbf{A}}^{-1}\tilde{\mathbf{b}}$

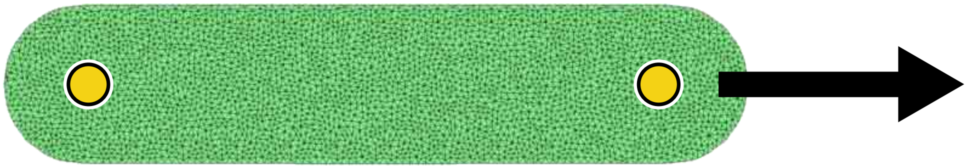
local step: 3x3 SVD for each rotation in $\mathbf{R}$

substitute

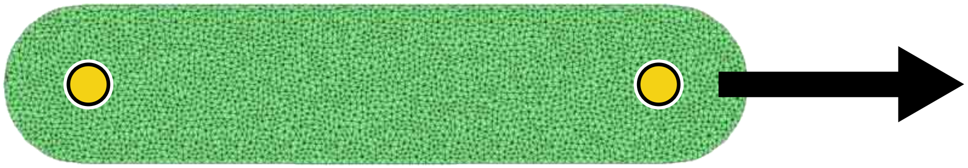
$$\mathbf{V}' = \mathbf{M}\mathbf{T}$$

similar to:
[Huang et al. 06]
[Der et al. 06]
[Au et al. 07]
[Hildebrandt et al. 12]

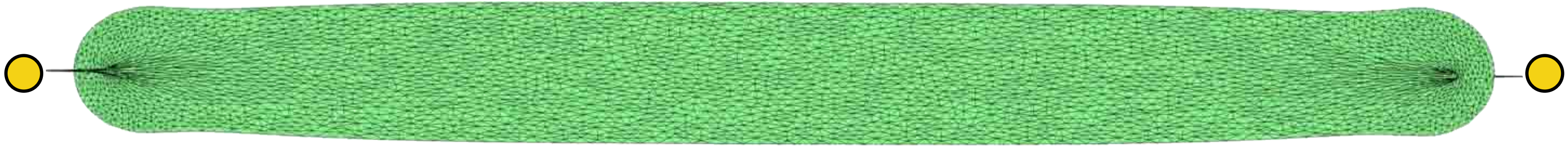
Direct reduction of elastic energies brings speed up and regularization…



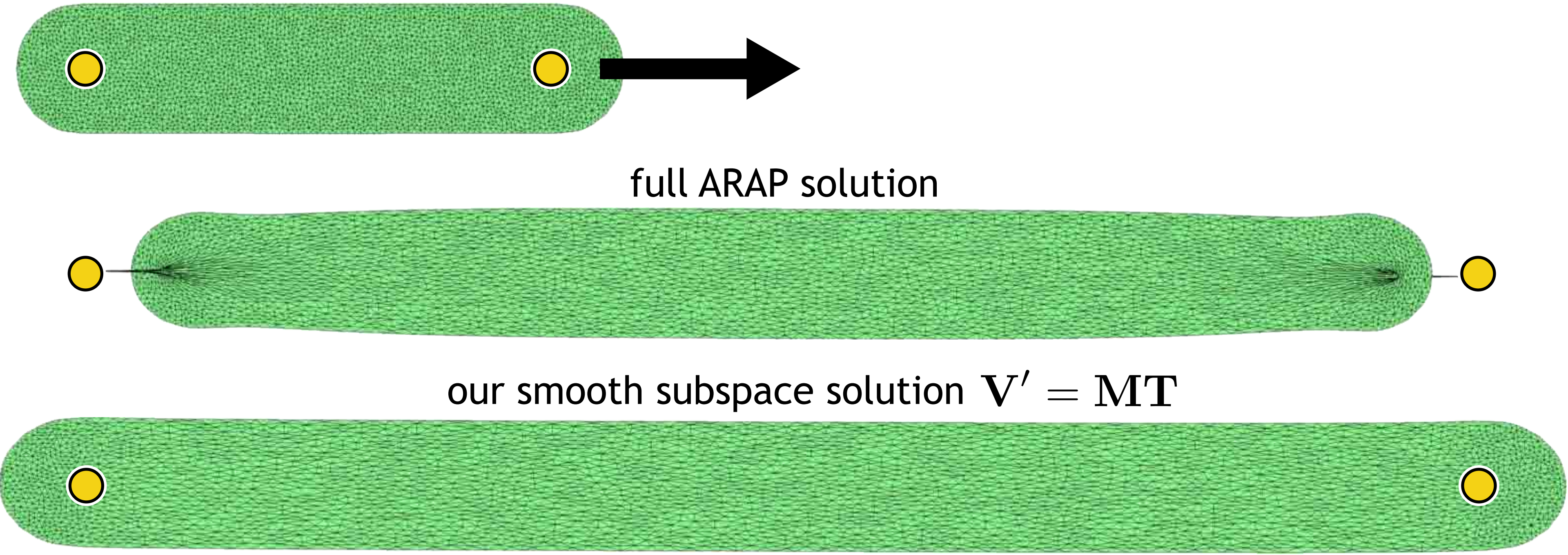
Direct reduction of elastic energies brings speed up and regularization…



full ARAP solution



Direct reduction of elastic energies brings speed up and regularization…



We reduce any *as-rigid-as-possible* energy

full energies $$E(\mathbf{V}', \mathbf{R}) = \frac{1}{2} \sum_{k=1}^{r} \sum_{(i,j) \in \mathcal{E}_k} c_{ijk} \|(\mathbf{v}'_i - \mathbf{v}'_j) - \mathbf{R}_k(\mathbf{v}_i - \mathbf{v}_j)\|^2$$

local/global optimization

global step: small, dense linear solve $\mathbf{T} = \tilde{\mathbf{A}}^{-1}\tilde{\mathbf{b}}$

local step: 3x3 SVD for each rotation in $\mathbf{R}$

but #rotations ~ full mesh discretization

substitute
$$\mathbf{V}' = \mathbf{M}\mathbf{T}$$

We reduce any *as-rigid-as-possible* energy

full energies $$E(\mathbf{V}',\mathbf{R}) = \frac{1}{2}\sum_{k=1}^{r}\sum_{(i,j)\in\mathcal{E}_k} c_{ijk}\|(\mathbf{v}'_i-\mathbf{v}'_j)-\mathbf{R}_k(\mathbf{v}_i-\mathbf{v}_j)\|^2$$

local/global optimization

global step: small, dense linear solve $\mathbf{T} = \tilde{\mathbf{A}}^{-1}\tilde{\mathbf{b}}$

local step: 3x3 SVD for each rotation in $\mathbf{R}$

substitute

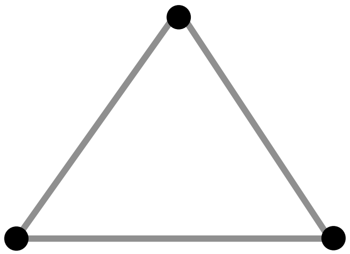
$$\mathbf{V}' = \mathbf{MT}$$

Cluster

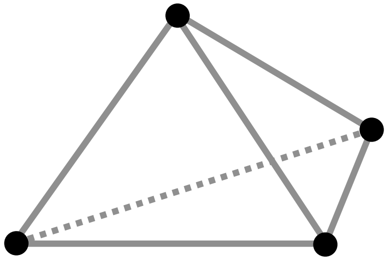
$$\mathcal{E}_k$$

Rotation evaluations may be reduced by clustering in *weight space*

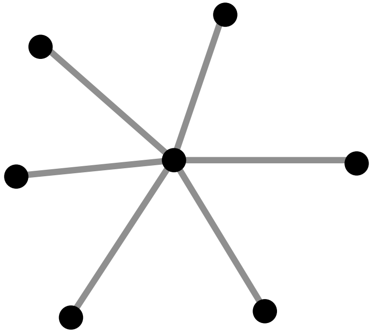
full energies $$E(\mathbf{V}',\mathbf{R}) = \frac{1}{2}\sum_{k=1}^{r}\boxed{\sum_{(i,j)\in\mathcal{E}_k}} c_{ijk}\left\|(\mathbf{v}'_i-\mathbf{v}'_j)-\mathbf{R}_k(\mathbf{v}_i-\mathbf{v}_j)\right\|^2$$



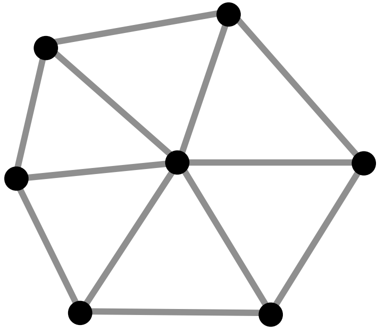
triangles
[Liu et al. 2008]



tetrahedra
[Chao et al. 2010]



"spokes"
[Sorkine & Alexa 2007]

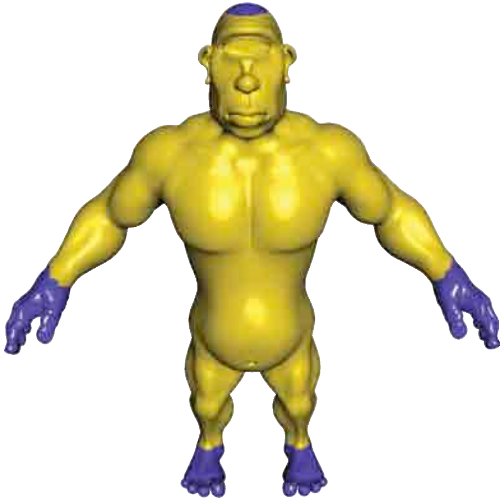


"spokes and rims"
[Chao et al. 2010]

Rotation evaluations may be reduced by k-means clustering in *weight space*

full energies

$$E(\mathbf{V}',\mathbf{R}) = \frac{1}{2}\sum_{k=1}^{r} \boxed{\sum_{(i,j)\in\mathcal{E}_k}} c_{ijk}\|(\mathbf{v}'_i - \mathbf{v}'_j) - \mathbf{R}_k(\mathbf{v}_i - \mathbf{v}_j)\|^2$$



weight space

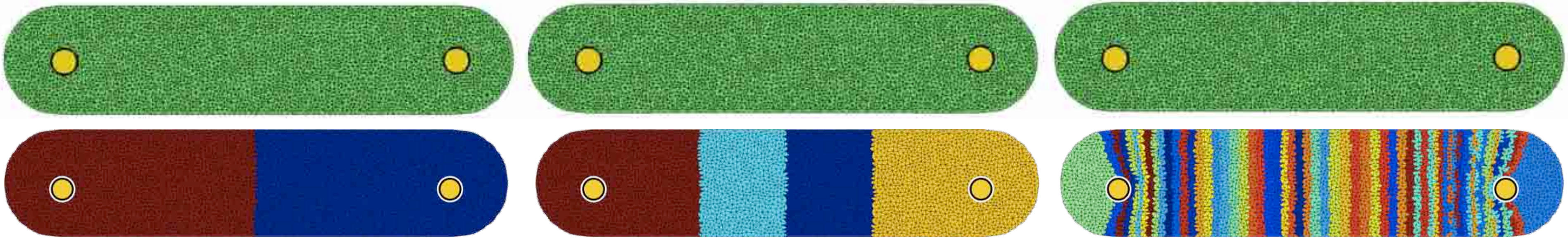
$$\mathbf{x}_j = \begin{bmatrix} w_1(\mathbf{v}_j) \\ w_2(\mathbf{v}_j) \\ \vdots \\ w_m(\mathbf{v}_j) \end{bmatrix}$$

Rotation evaluations may be reduced by clustering in *weight space*

full energies $$E(\mathbf{V}', \mathbf{R}) = \frac{1}{2}\sum_{k=1}^{r} \boxed{\sum_{(i,j)\in\mathcal{E}_k}} c_{ijk}\left\|(\mathbf{v}_i' - \mathbf{v}_j') - \mathbf{R}_k(\mathbf{v}_i - \mathbf{v}_j)\right\|^2$$

Rotation evaluations may be reduced by clustering in *weight space*

full energies $$E(\mathbf{V}',\mathbf{R}) = \frac{1}{2}\sum_{k=1}^{r}\sum_{(i,j)\in\mathcal{E}_k} c_{ijk}\|(\mathbf{v}'_i-\mathbf{v}'_j)-\mathbf{R}_k(\mathbf{v}_i-\mathbf{v}_j)\|^2$$



We reduce any *as-rigid-as-possible* energy

full energies $$E(\mathbf{V}', \mathbf{R}) = \frac{1}{2} \sum_{k=1}^{r} \sum_{(i,j)\in\mathcal{E}_k} c_{ijk} \|(\mathbf{v}'_i - \mathbf{v}'_j) - \mathbf{R}_k(\mathbf{v}_i - \mathbf{v}_j)\|^2$$

local/global optimization

global step: small, dense linear solve $\mathbf{T} = \tilde{\mathbf{A}}^{-1}\tilde{\mathbf{b}}$

local step: 3x3 SVD for each rotation in $\mathbf{R}$

#rotations ~ #T,
independent of full mesh resolution

substitute
$$\mathbf{V}' = \mathbf{MT}$$
Cluster
$$\mathcal{E}_k$$

With more and more user constraints we fall back to standard skinning

With more and more user constraints we fall back to standard skinning

With more and more user constraints we fall back to standard skinning

Extra weights expand deformation subspace

no extra weights

15 extra weights

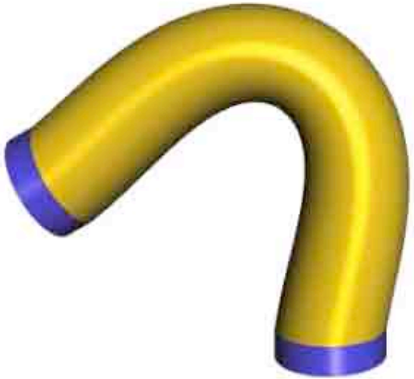
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Extra weights expand deformation subspace

no extra weights

15 extra weights

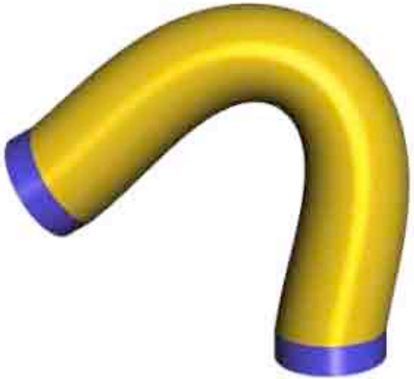
Subspace now rich enough for fast variational modeling



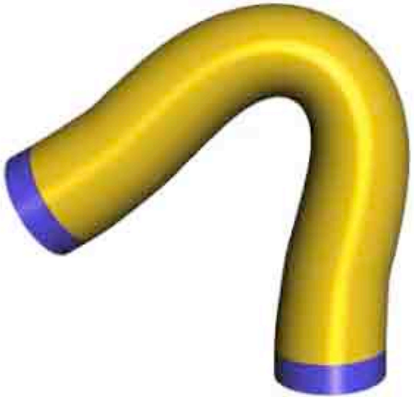
full non-linear optimization
[Botsch et al. 2006]

our reduced method

Subspace now rich enough for fast variational modeling

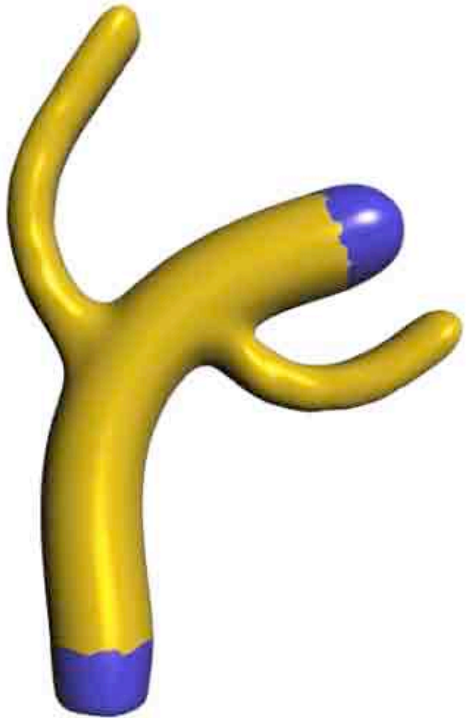


full non-linear optimization
[Botsch et al. 2006]



our reduced method

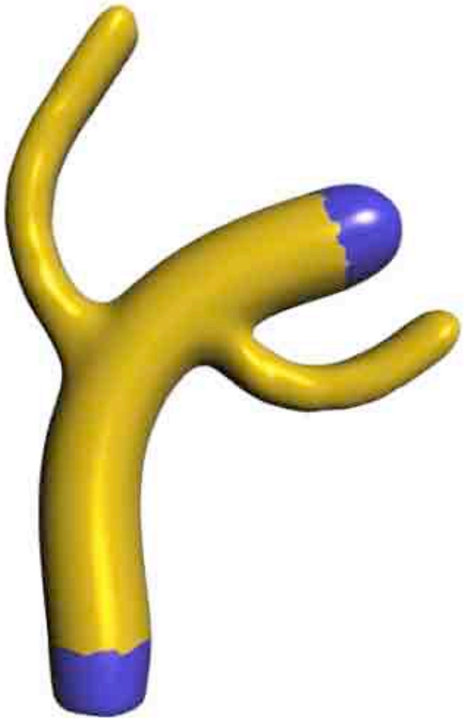
Subspace now rich enough for fast variational modeling



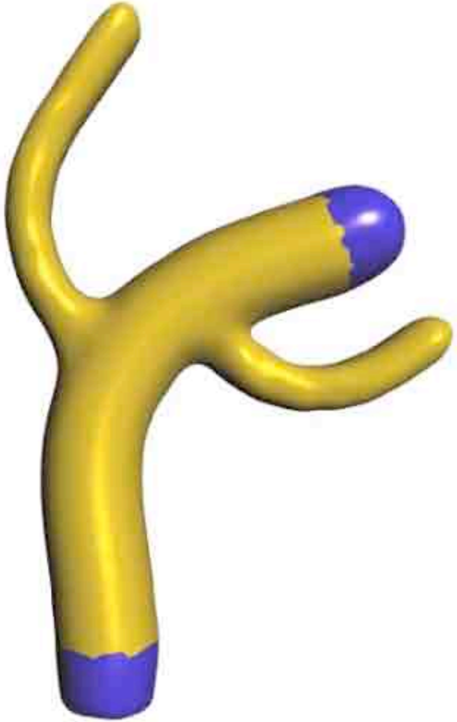
full non-linear optimization
[Botsch et al. 2006]

our reduced method

Subspace now rich enough for fast variational modeling

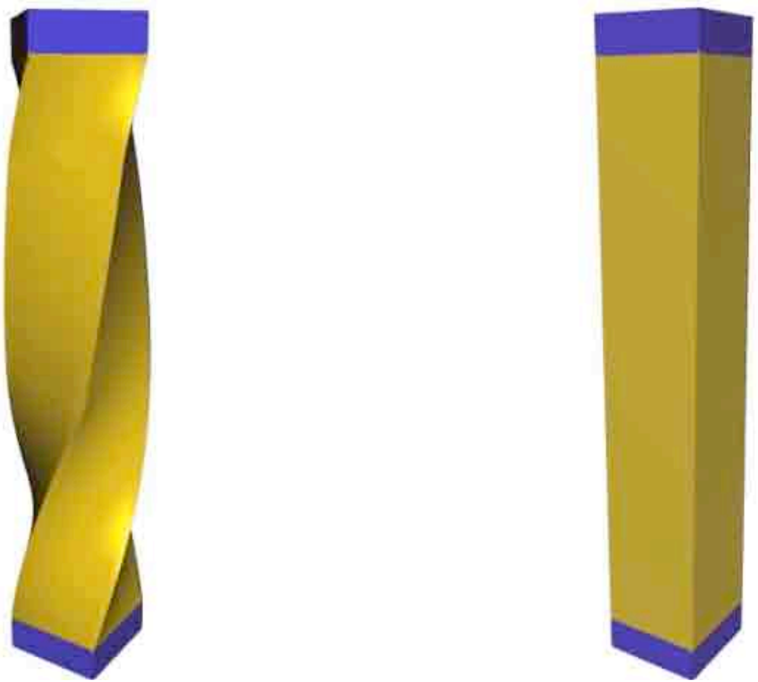


full non-linear optimization
[Botsch et al. 2006]



our reduced method

Subspace now rich enough for fast variational modeling



full non-linear optimization
[Botsch et al. 2006]

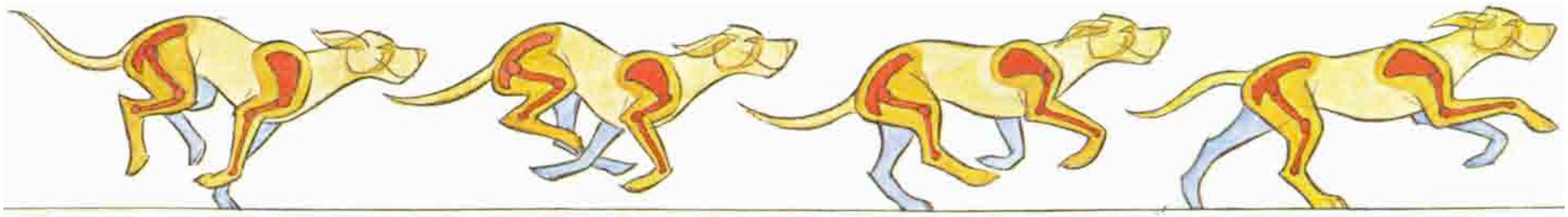
our reduced method

Subspace now rich enough for fast variational modeling

full non-linear optimization
[Botsch et al. 2006]

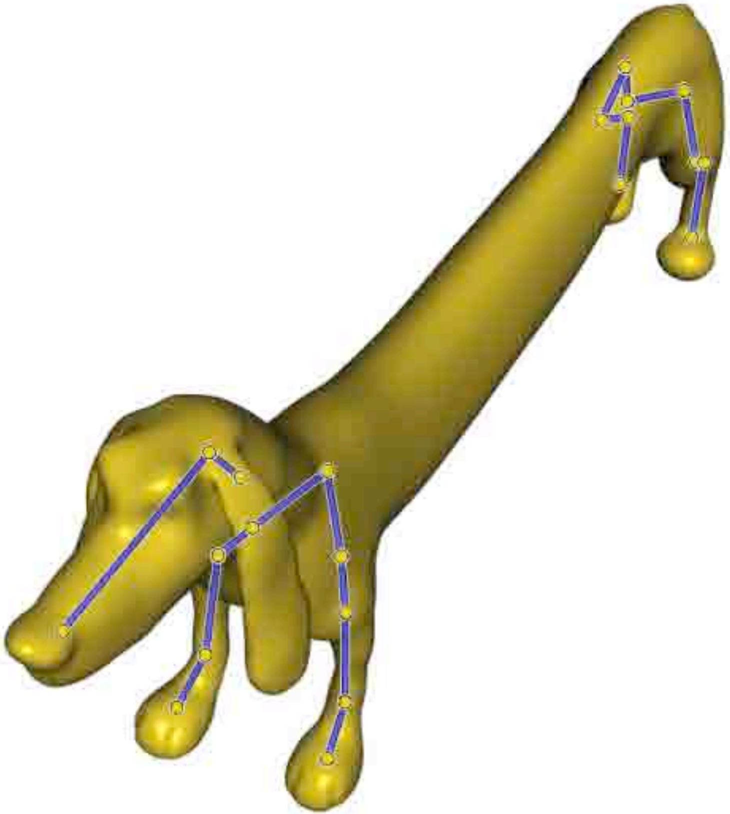
our reduced method

Extra weights and disjoint skeletons make flexible control easy

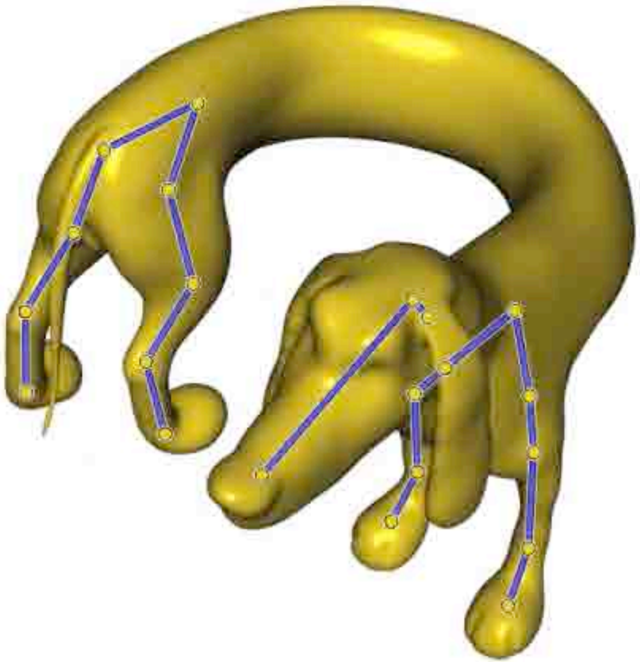


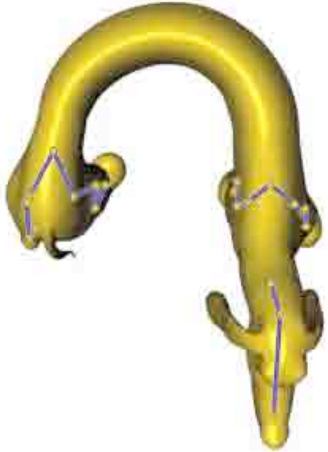
From *Cartoon Animation* by Preston Blair

Extra weights and disjoint skeletons make flexible control easy

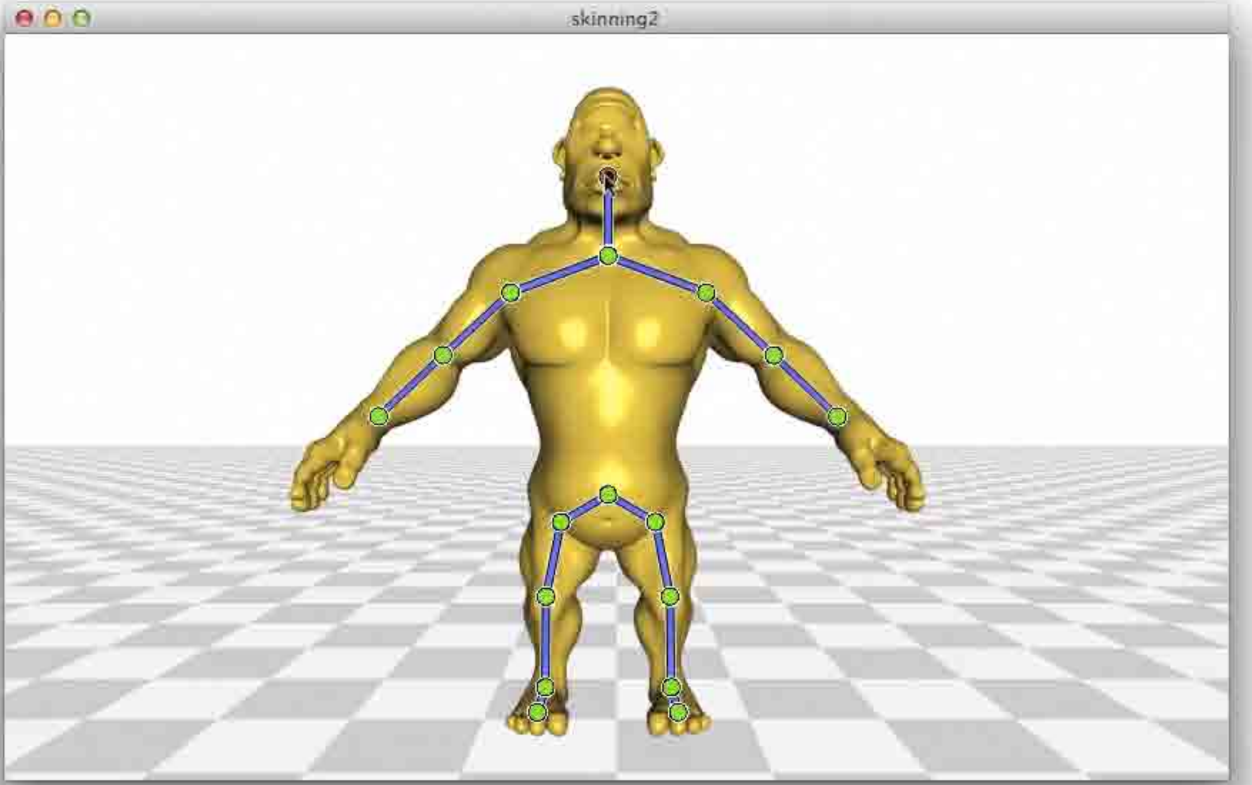


Extra weights and disjoint skeletons make flexible control easy



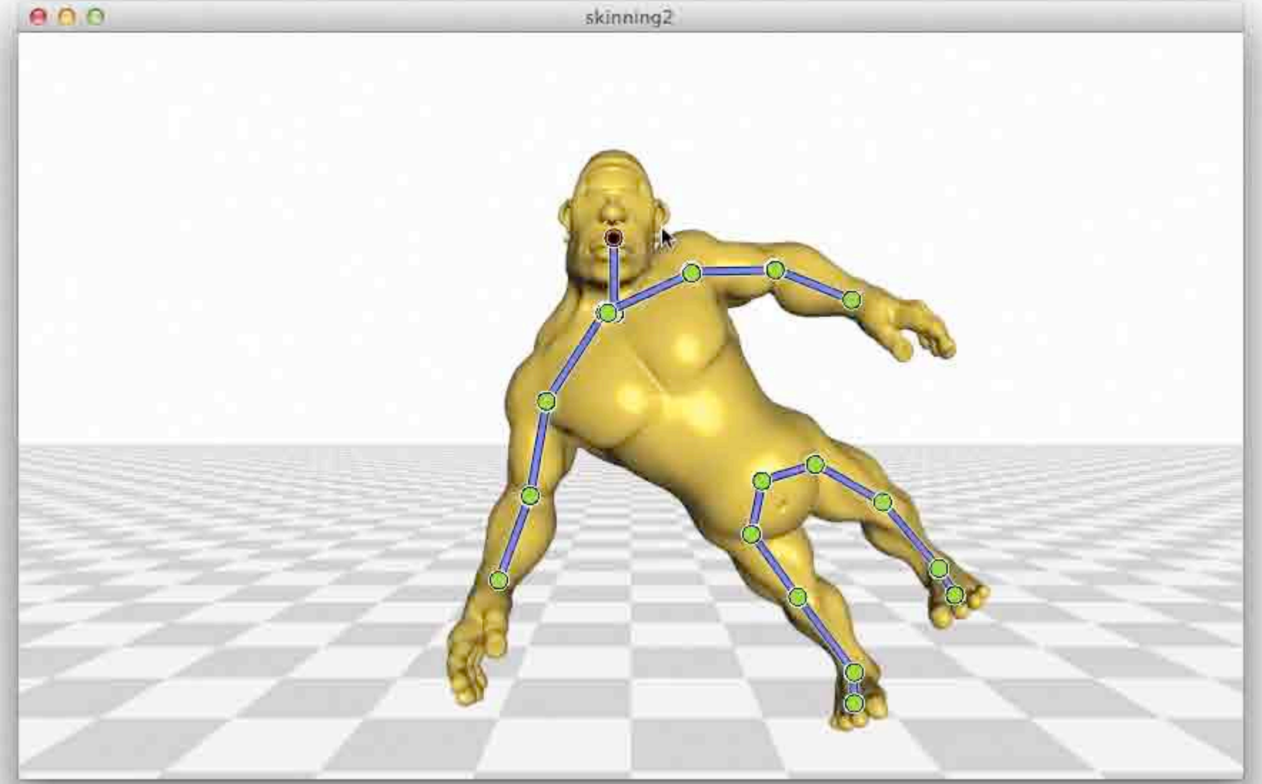


Physical dynamics also benefit from our reduction



Demo

Physical dynamics also benefit from our reduction



Demo

Our reduction preserves nature of different energies, at no extra cost

Surface ARAP

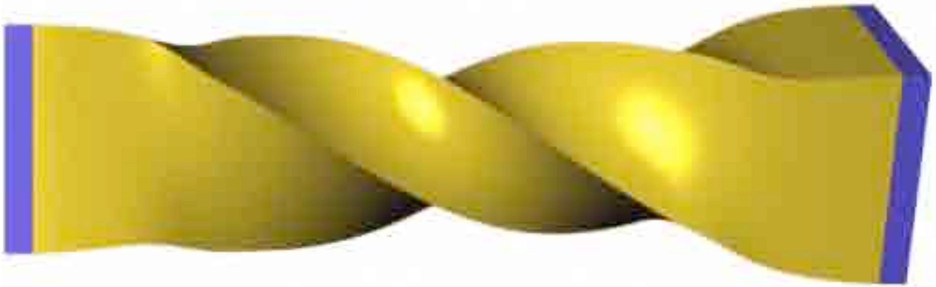
$$\mathbf{V}'_{\mathrm{surf}} = \mathbf{M}_{\mathrm{surf}} T$$

Volumetric ARAP

$$\mathbf{V}'_{\mathrm{vol}} = \mathbf{M}_{\mathrm{vol}} T$$

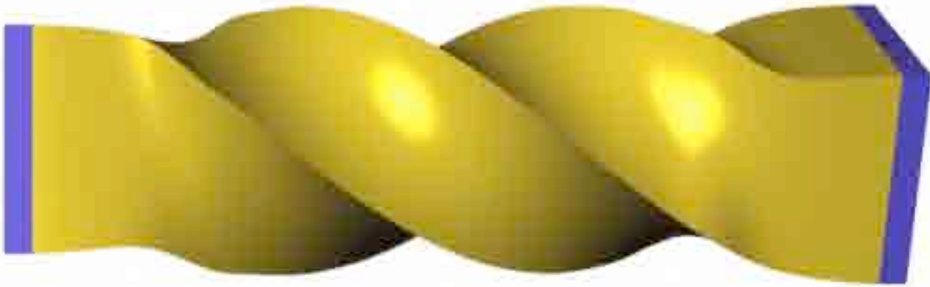
Our reduction preserves nature of different energies, at no extra cost

Surface ARAP



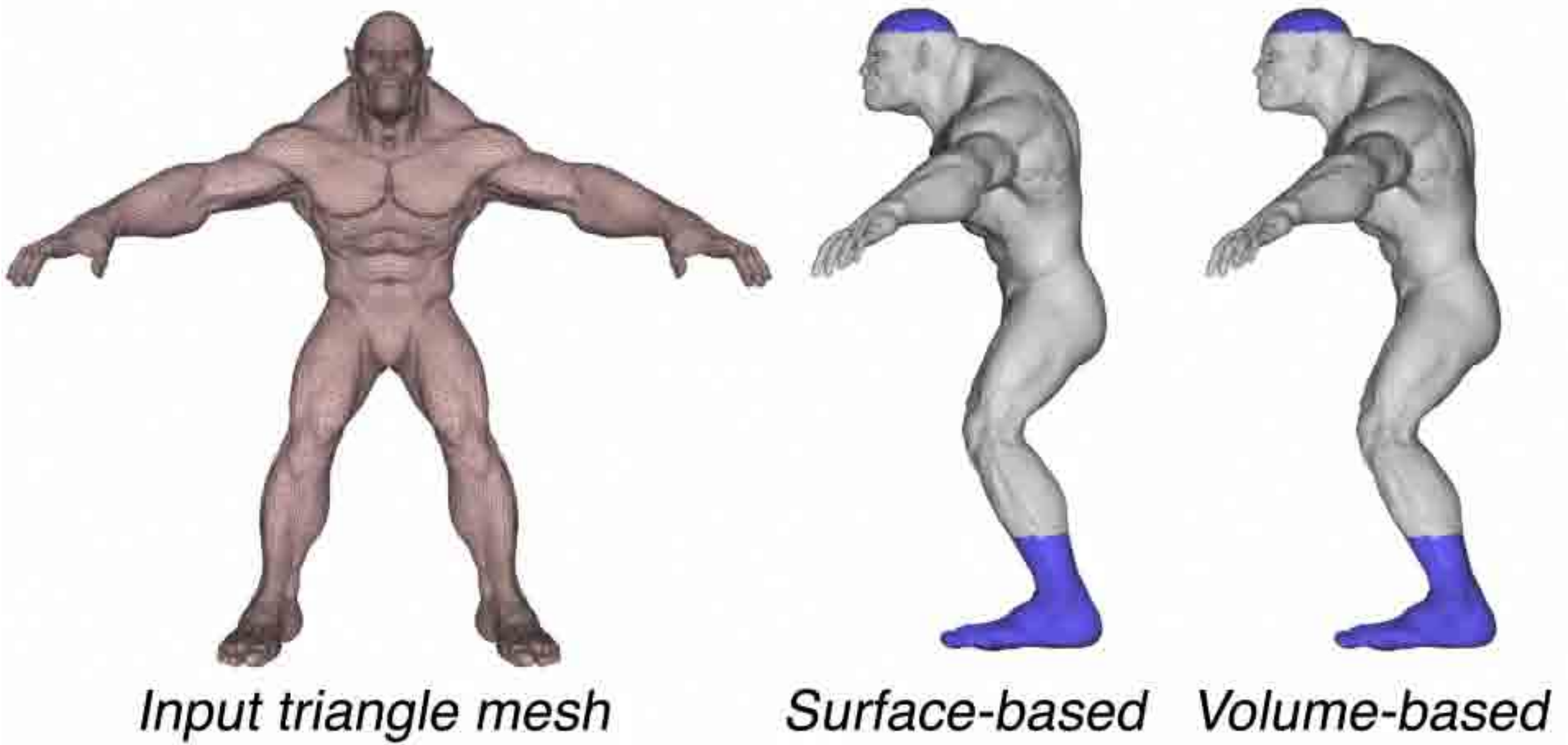
$$\mathbf{V}'_{\text{surf}} = \mathbf{M}_{\text{surf}} T$$

Volumetric ARAP

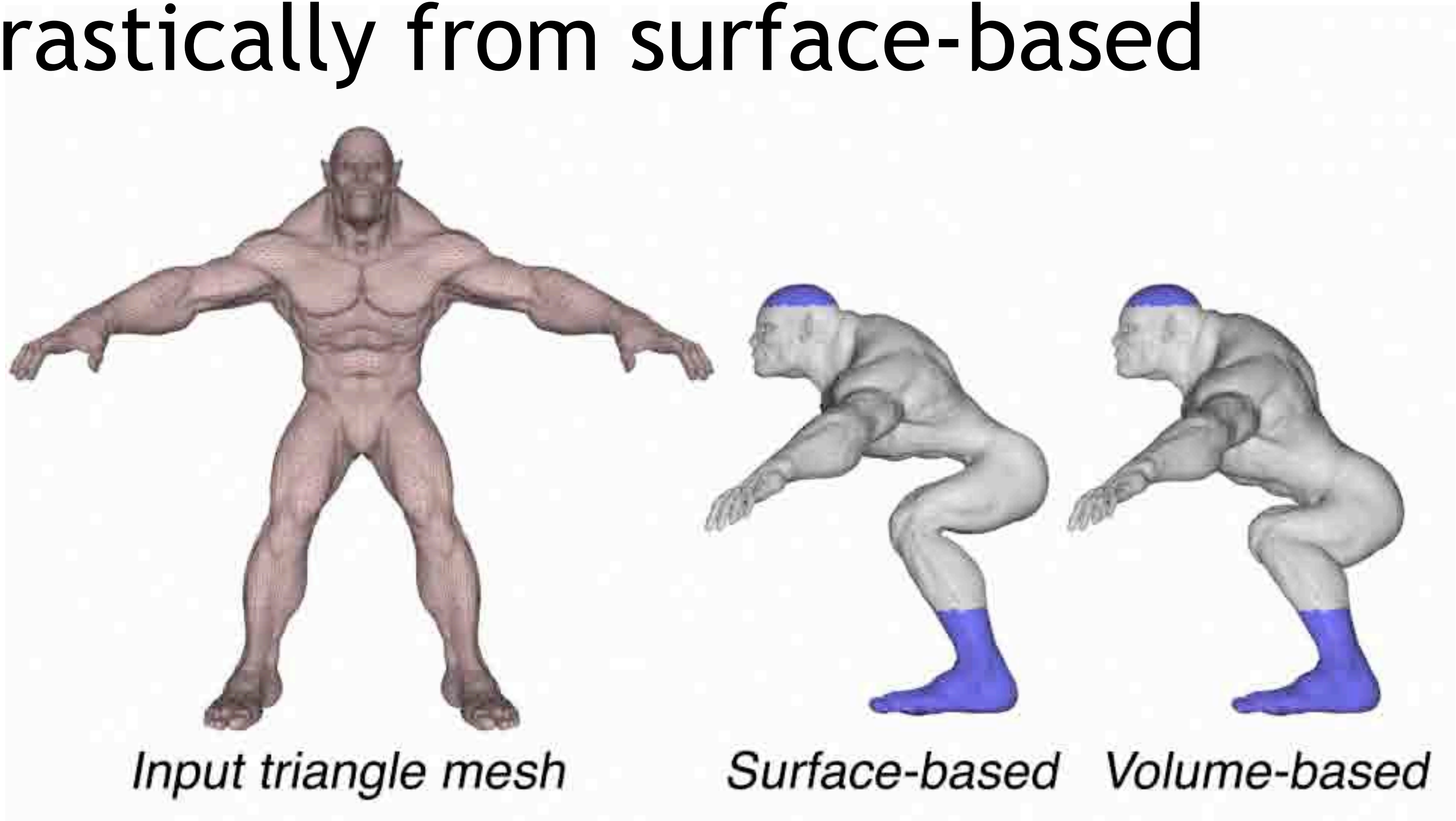


$$\mathbf{V}'_{\text{vol}} = \mathbf{M}_{\text{vol}} T$$

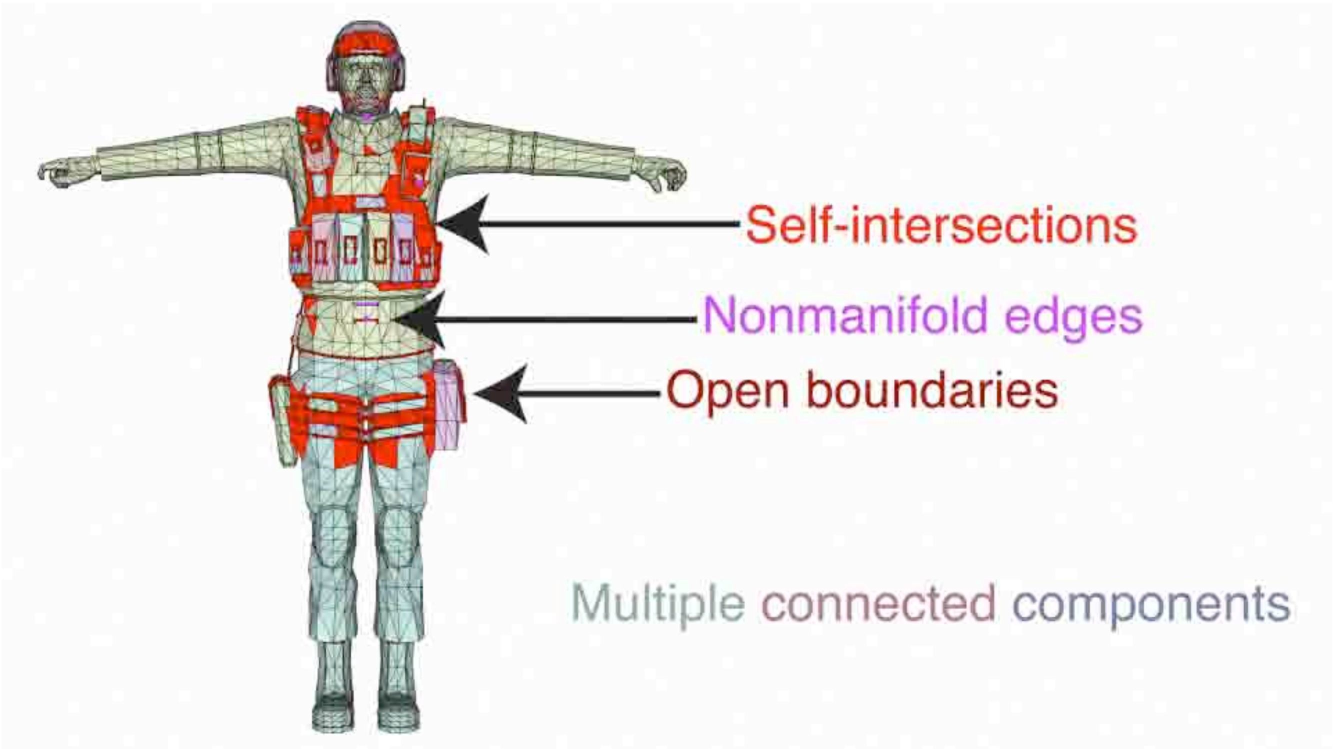
Volumetric deformation differs drastically from surface-based



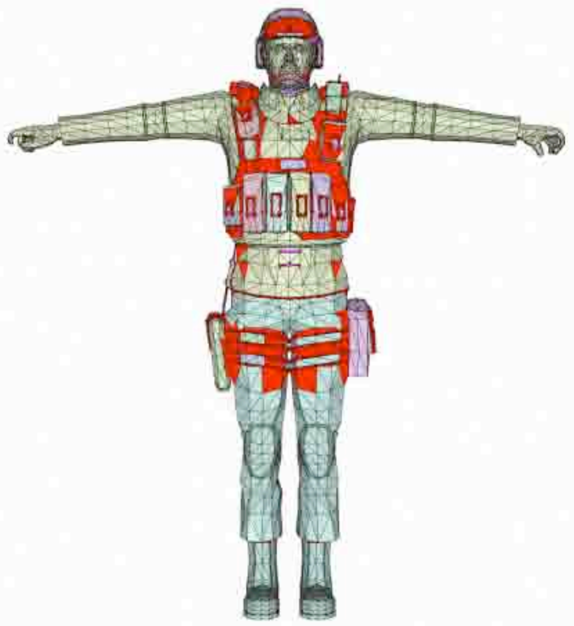
Volumetric deformation differs drastically from surface-based

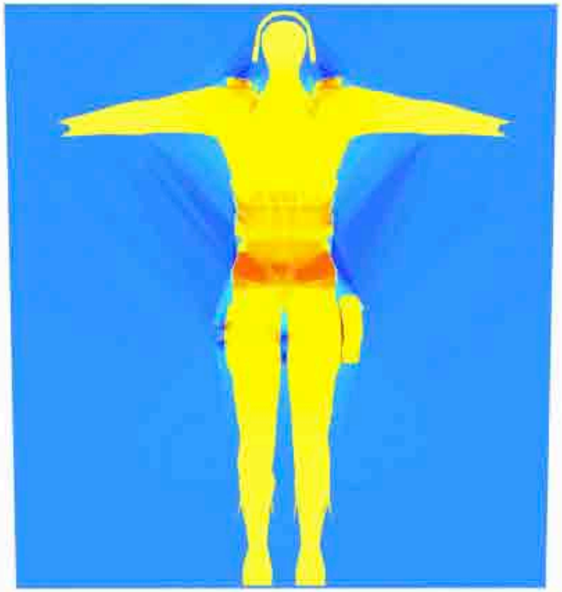


Surface artifacts prevent volume meshing, prevent volumetric processing



Continuous winding number for watertight surfaces generalizes to *unclean* triangle meshes



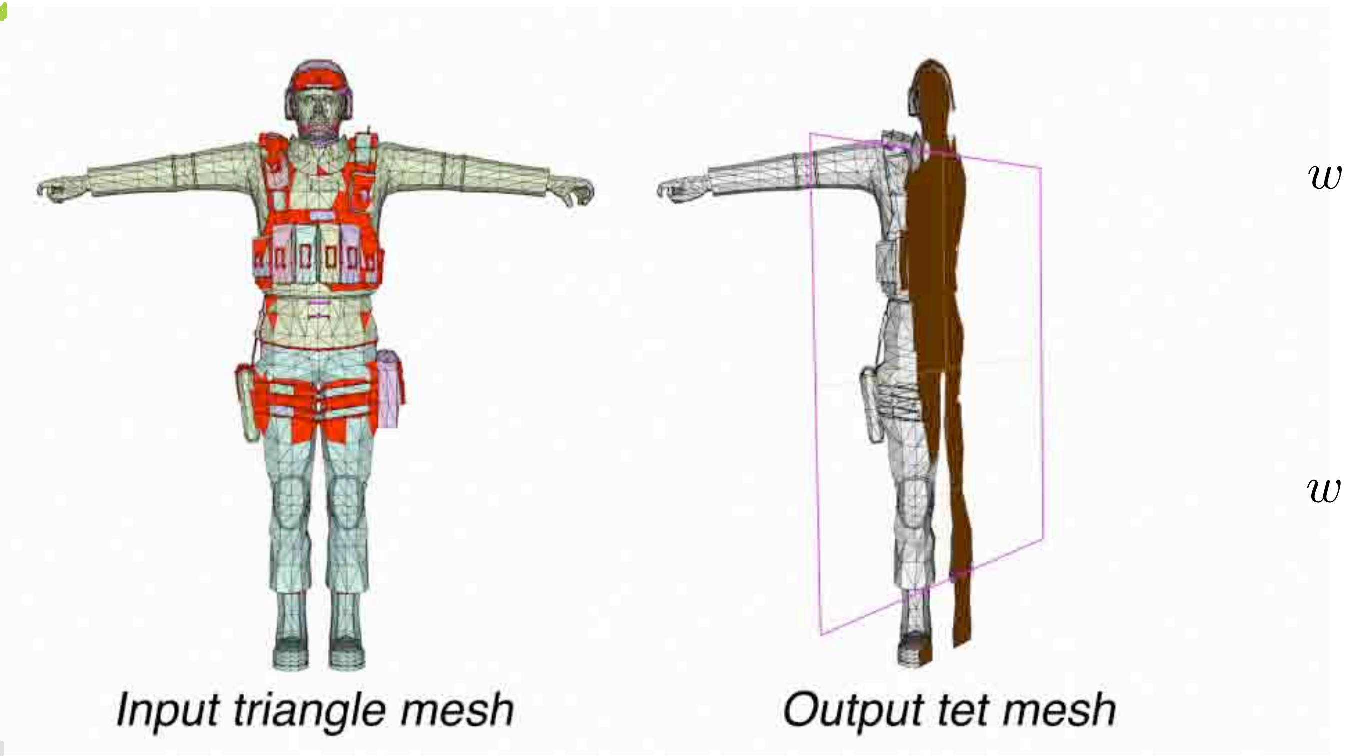


$$w(\mathbf{p}) = \frac{1}{4\pi} \iint_S \sin(\phi)\, d\theta\, d\phi$$

$$w(\mathbf{p}) = \frac{1}{4\pi} \sum_{f=1}^{m} \Omega_f$$

[SIGGRAPH 2013]

Generalized winding number is ideal indicator for graphcut segmentation of convex hull

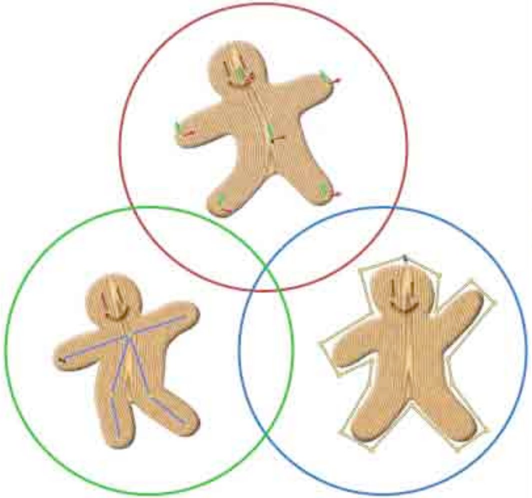


$$w(\mathbf{p}) = \frac{1}{4\pi} \iint_S \sin(\phi)\, d\theta\, d\phi$$

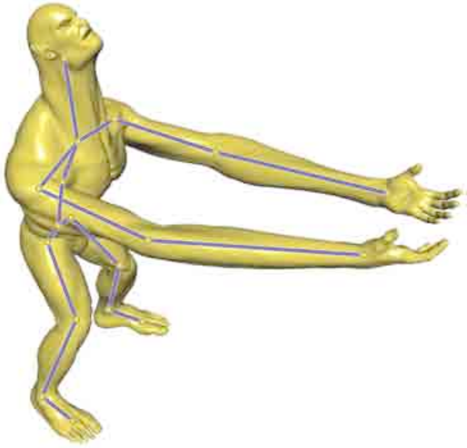
$$w(\mathbf{p}) = \frac{1}{4\pi} \sum_{f=1}^{m} \Omega_f$$

[SIGGRAPH 2013]

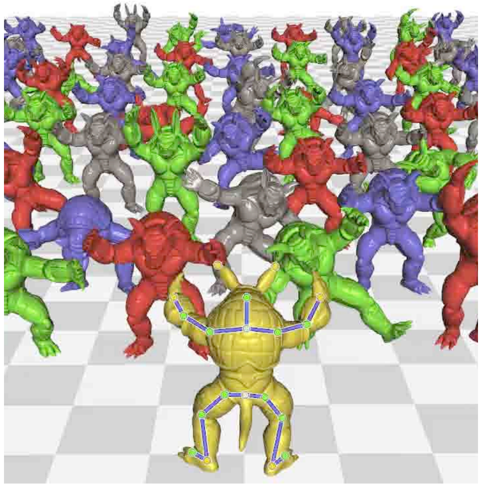
Each advance inspires new improvements, new interfaces



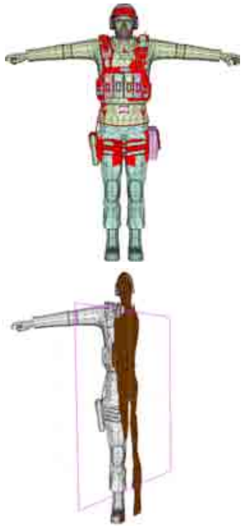
SGP 2010
SIGGRAPH 2011
SGP 2012



SIGGRAPH Asia 2011



SIGGRAPH 2012



SIGGRAPH 2013

We derive a powerful subspace via input shape and handle descriptions

- *Intrinsics* from shape's geometry
- *Semantics* from handles
- Fully automatic
- New interfaces
- Real-time as an invariant

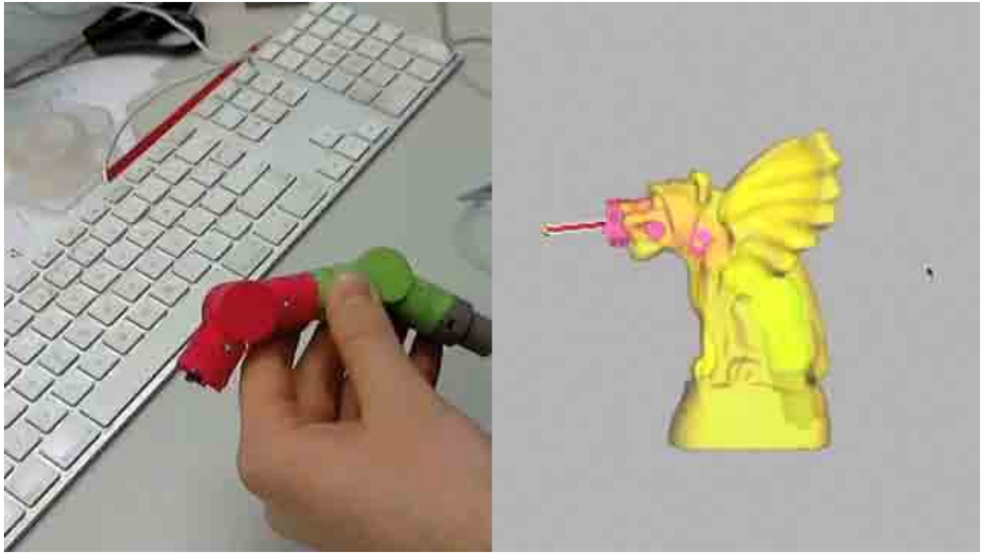
Future outlook

- More semantics: large data, collisions, etc.

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Future outlook

- More semantics: large data, collisions, etc.
- Physical interfaces



Acknowledgements

Coauthors: Ladislav Kavan, Ilya Baran, Jovan Popović,
Kaan Yücer, Alex Sorkine-Hornung, Tino Weinkauf, Denis Zorin,
Olga Sorkine-Hornung

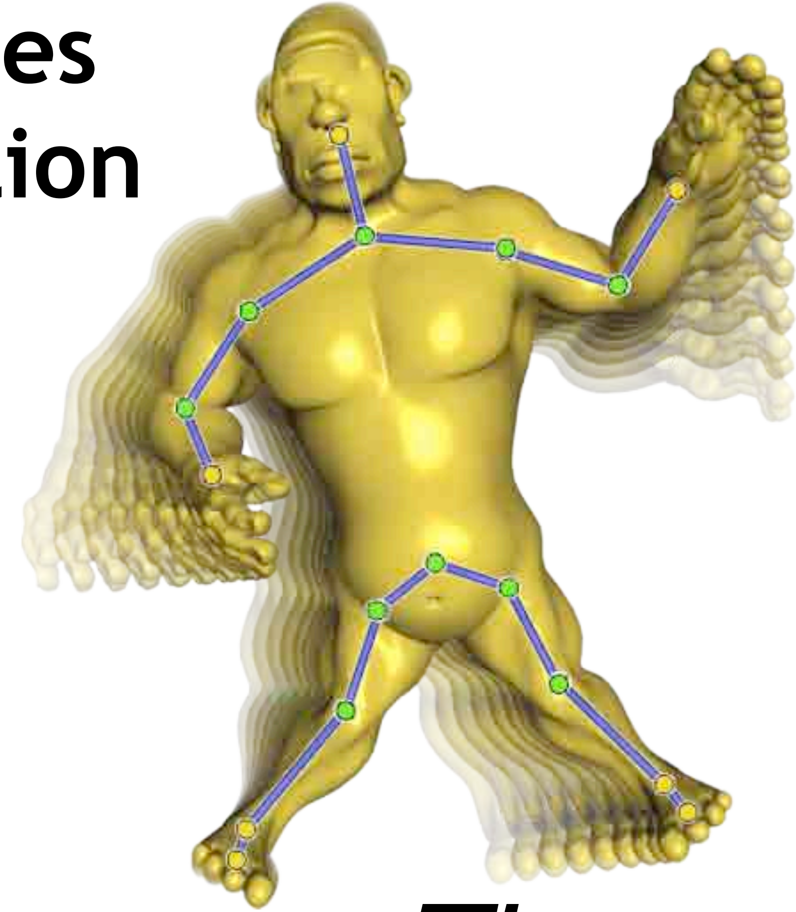
Colleagues at NYU and ETH

Friends and family

This thesis was supported in part by NSF award IIS-0905502,
ERC grant iModel (StG-2012-306877), SNF award 200021_137879,
the Intel Doctoral Fellowship and a gift from Adobe Systems.

Algorithms and Interfaces for Real-Time Deformation of 2D and 3D Shapes

Papers, videos, code: people.inf.ethz.ch/~jalec/



Alec Jacobson

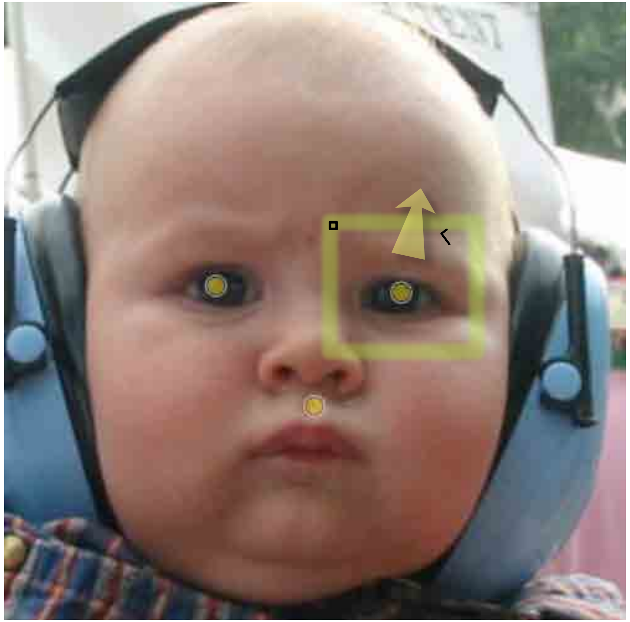
jacobson@inf.ethz.ch

INTERACTIVE GEOMETRY LAB

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

June 9, 2013

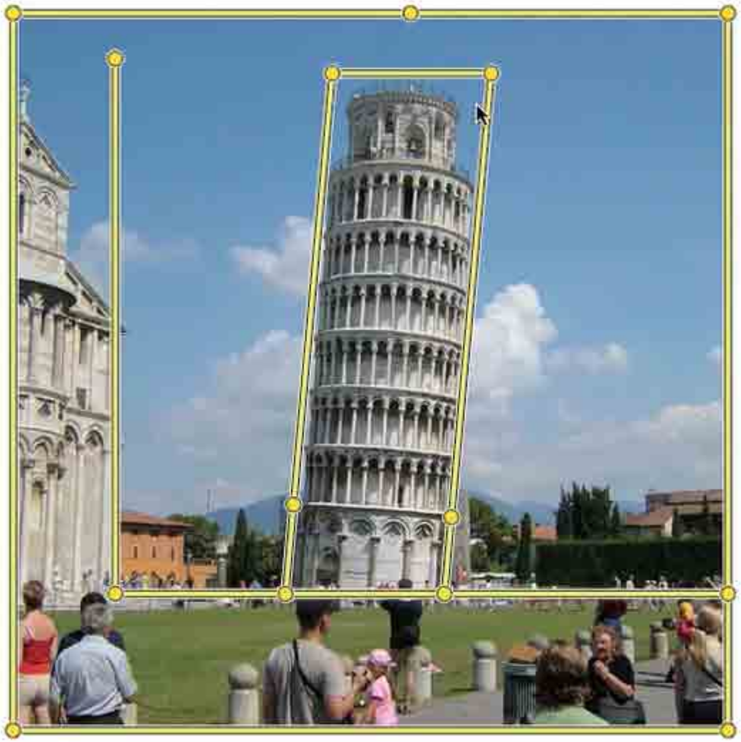
Weights should be smooth everywhere, *especially* at handles



Our method

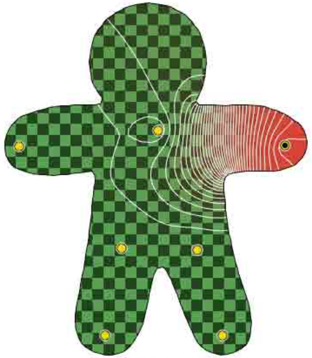
Extension of Harmonic Coordinates
[Joshi et al. 2005]

Open cages allow arbitrary line constraints

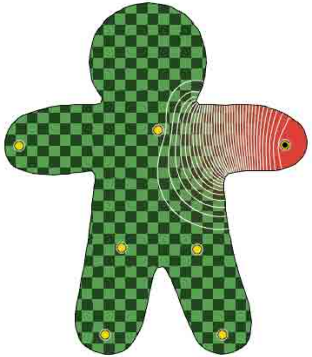


Sparsity helps maintain local influence

Smoothed extension of
Harmonic Coordinates
[Joshi et al. 2005]

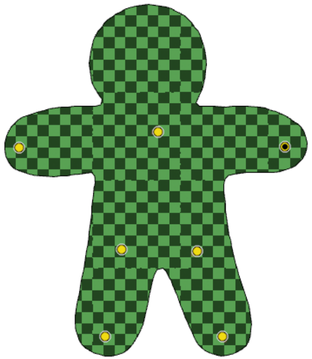


Our method

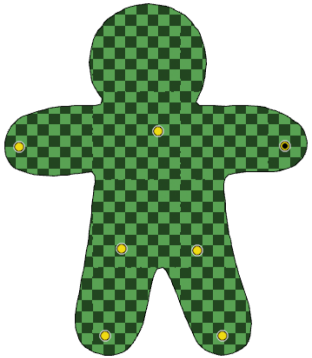


Sparsity helps maintain local influence

Smoothed extension of Harmonic Coordinates [Joshi et al. 2005]

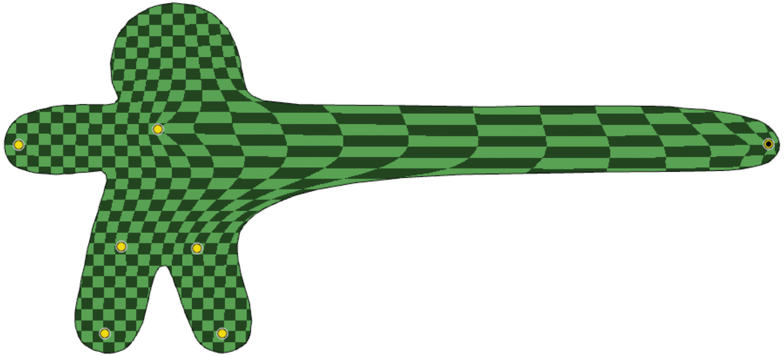


Our method

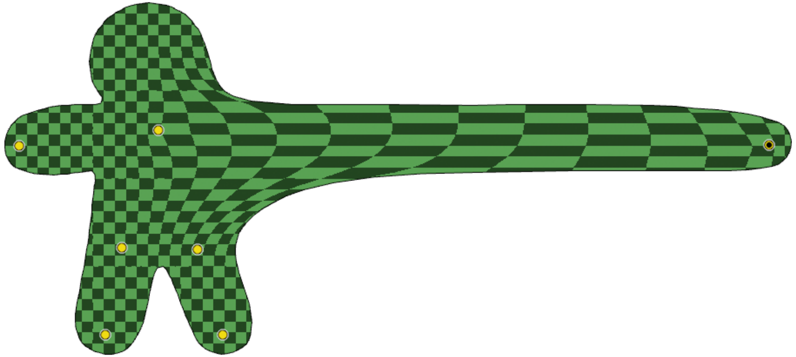


Sparsity helps maintain local influence

Smoothed extension of Harmonic Coordinates [Joshi et al. 2005]

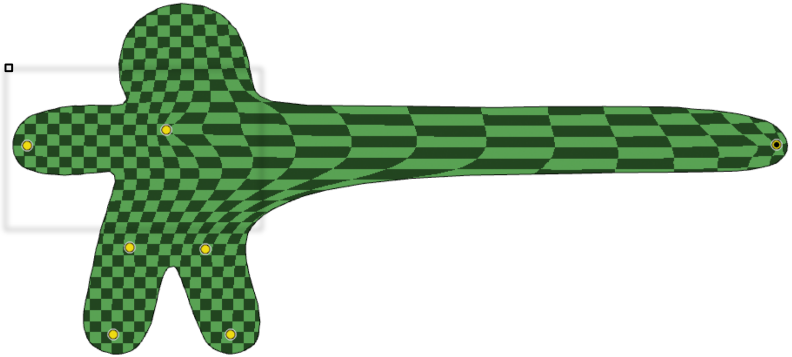


Our method

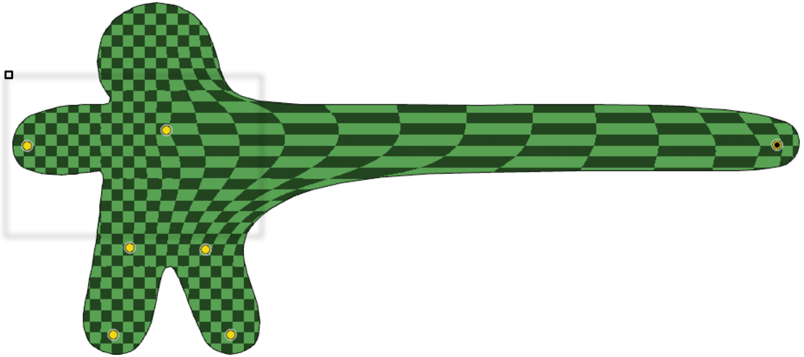


Sparsity helps maintain local influence

Smoothed extension of Harmonic Coordinates [Joshi et al. 2005]

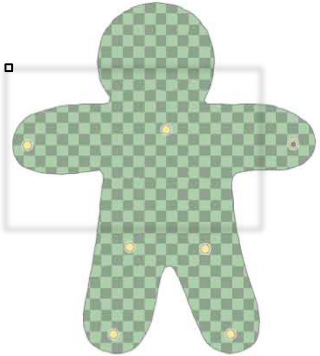


Our method

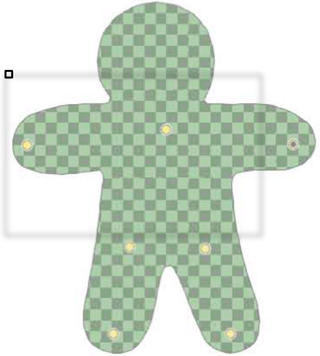


Sparsity helps maintain local influence

Smoothed extension of Harmonic Coordinates [Joshi et al. 2005]



Our method

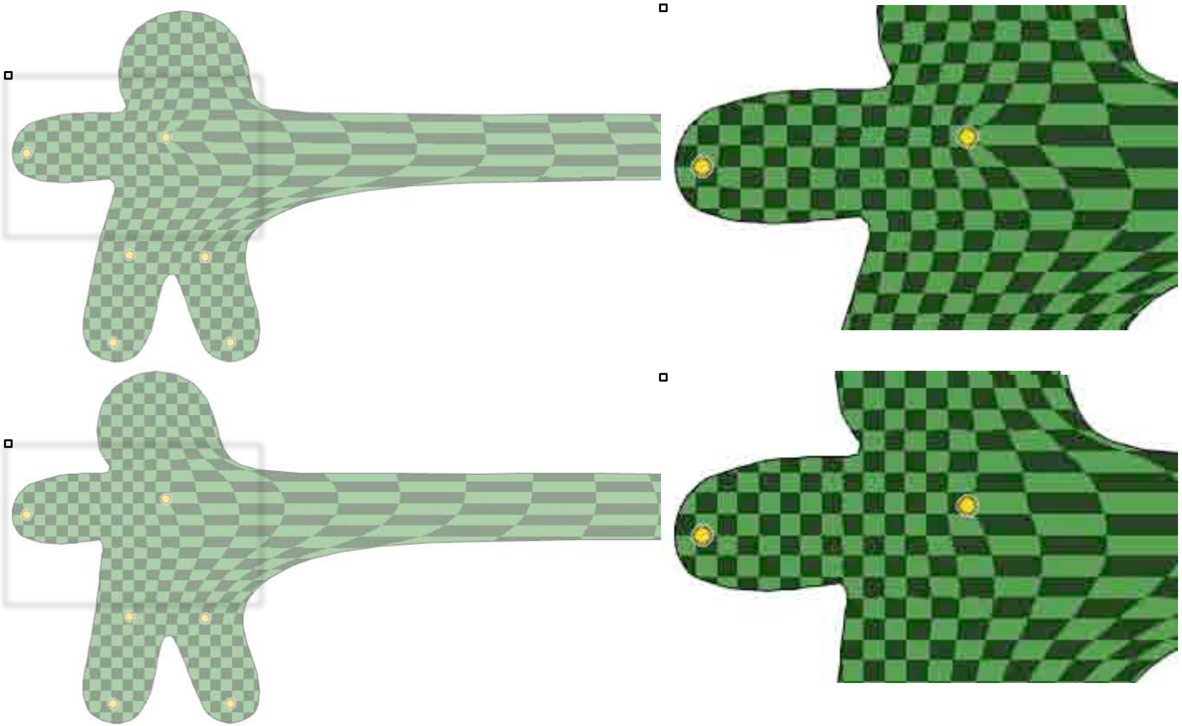


Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

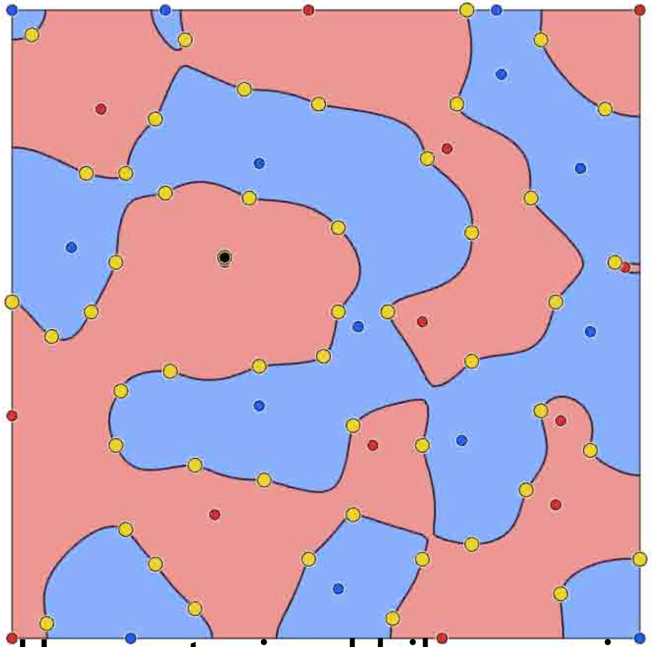
Sparsity helps maintain local influence

Smoothed extension of Harmonic Coordinates [Joshi et al. 2005]

Our method

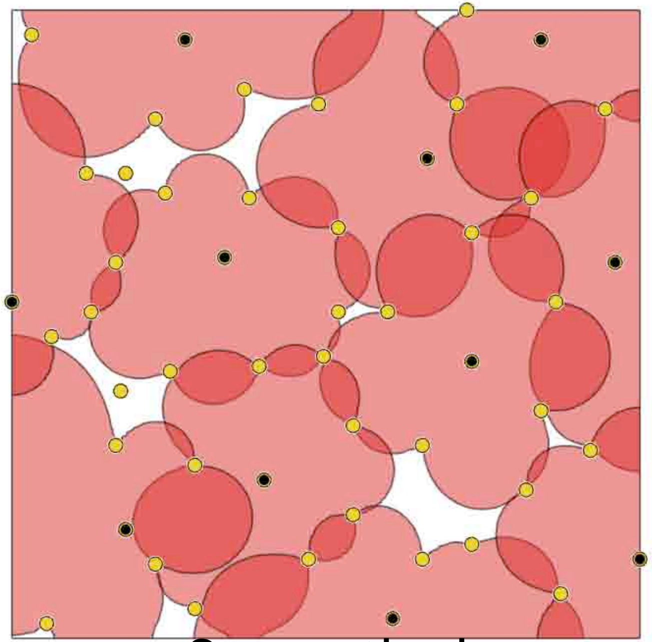


Boundedness also helps maintain local influence

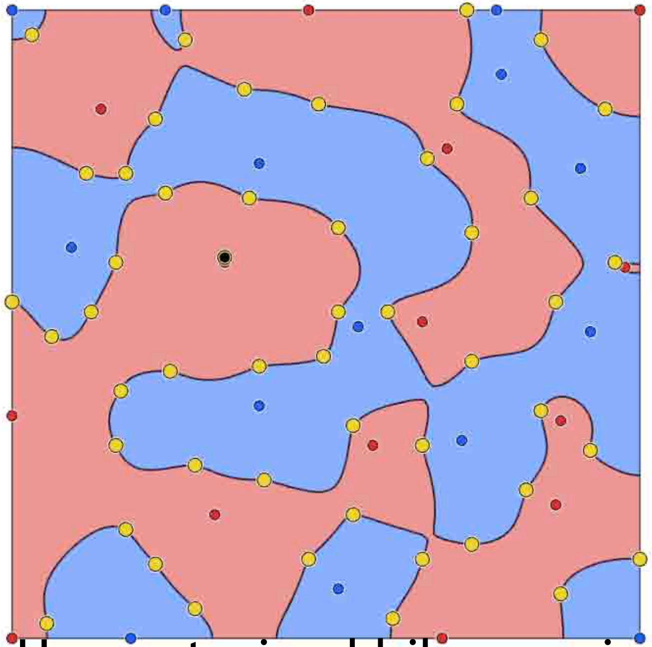


Unconstrained biharmonic
[Botsch and Kobbelt 2004]

Boundedness also helps maintain local influence

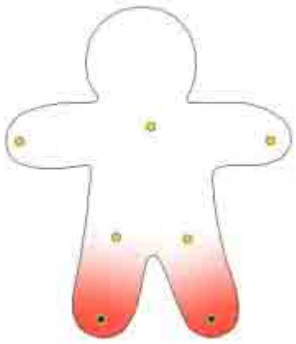


Our method

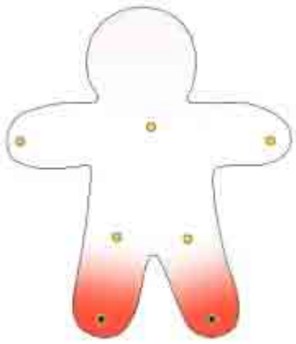


Unconstrained biharmonic
[Botsch and Kobbelt 2004]

Spurious local maxima also cause unintuitive response



Our method



Extension of unconstrained biharmonic
[Botsch and Kobbelt 2004]

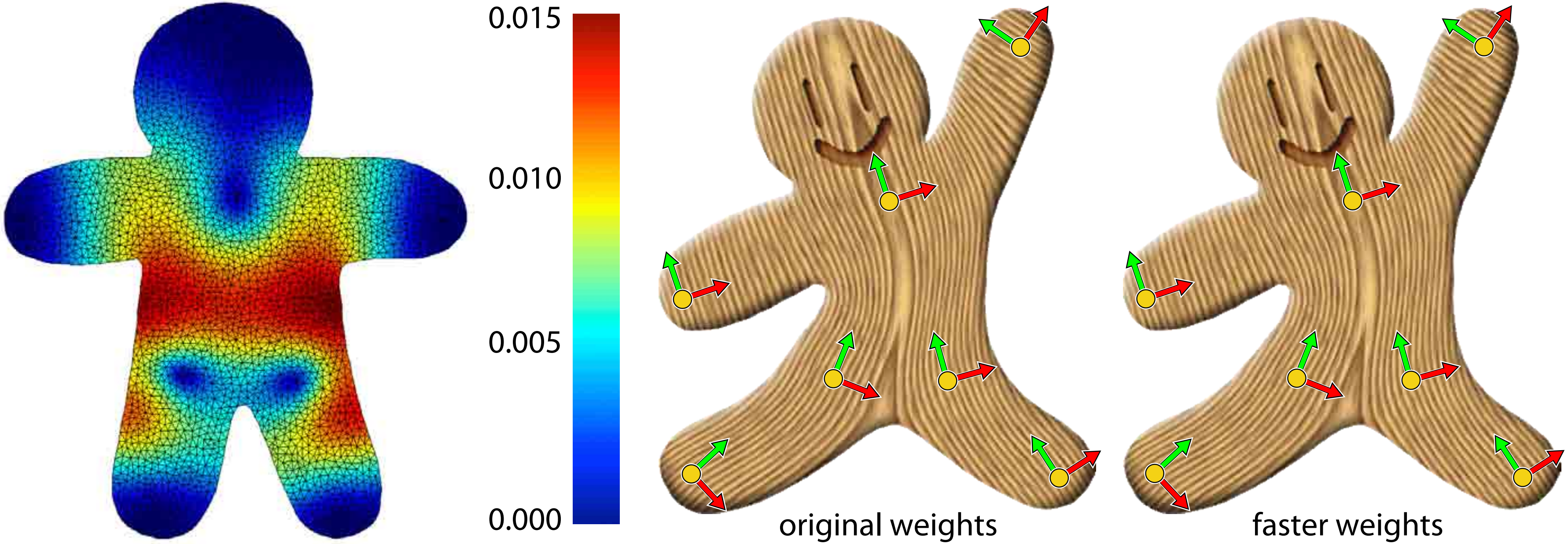
Weights propagate transformations at handles to shape in real-time

Translate

Rotate

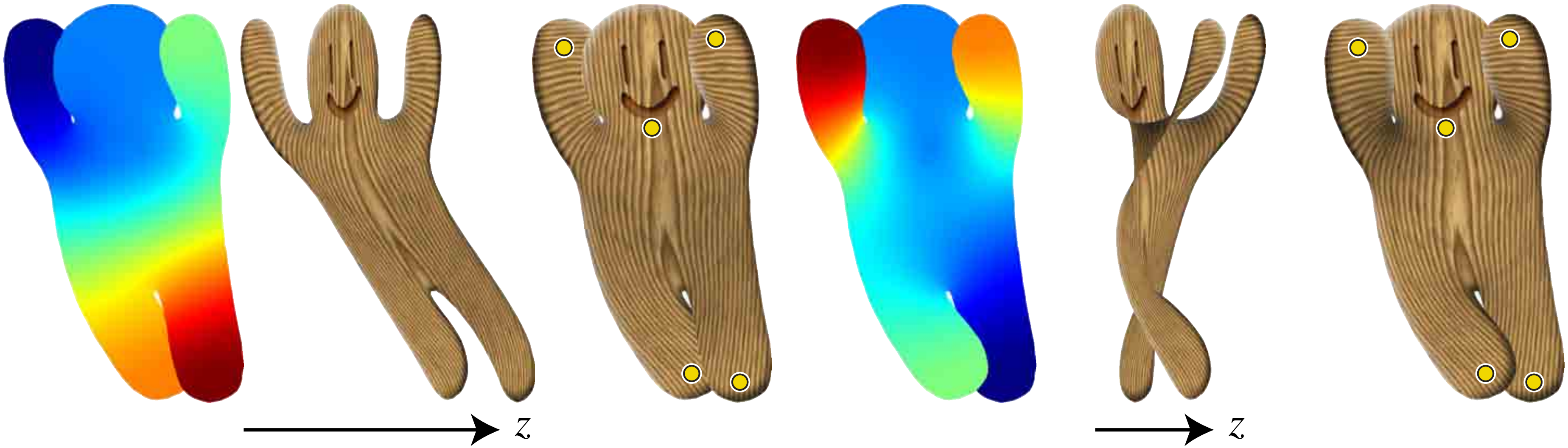
Scale

Dropping partition of unity as explicit constraint does not effect quality

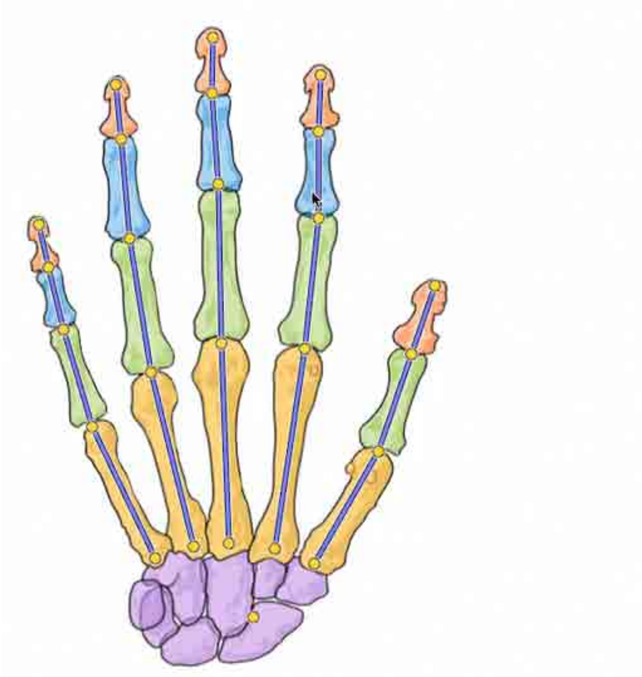


Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

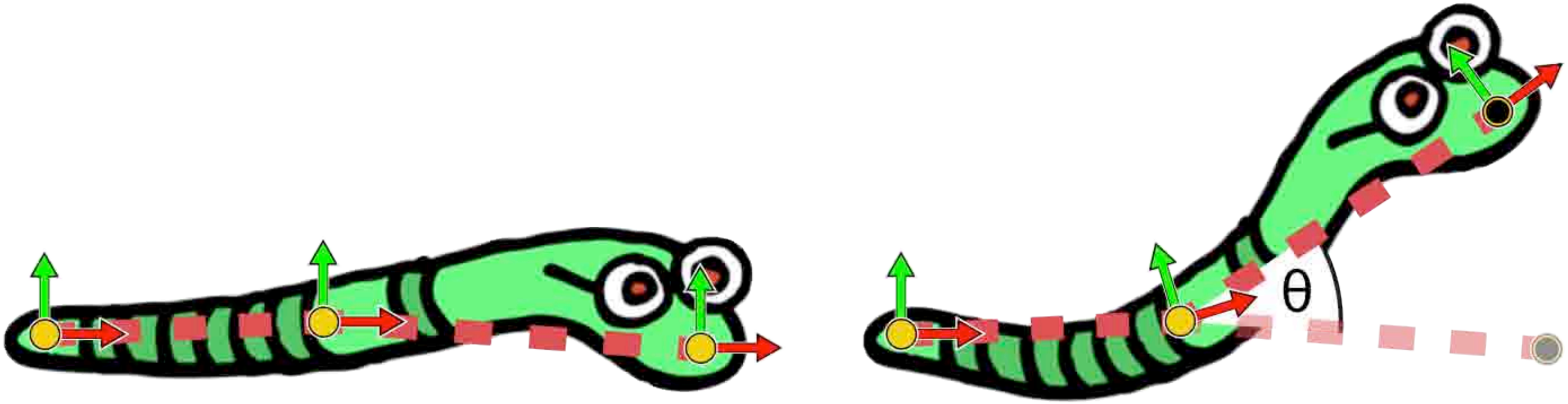
Weights may also define an intuitive, shape-aware depth ordering in 2D



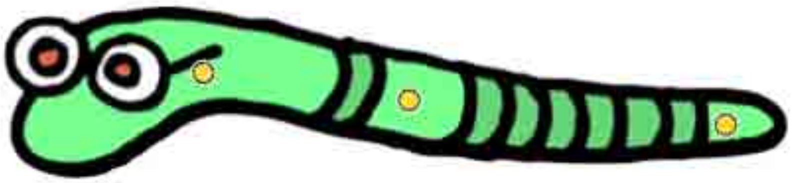
Weights may also define an intuitive, shape-aware depth ordering in 2D



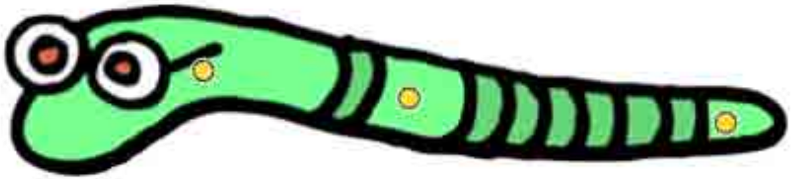
Rotations at point handles may be computed automatically based on translations



Alternative skinning methods may also take advantage of bounded biharmonic weights



Linear blend skinning



Dual quaternion skinning

Same weights can interpolate colors

$$\boxed{\mathbf{x}'} = \sum_{j=1}^{H} w_j(\mathbf{x}_i)\boxed{T_j\mathbf{x}_i}$$

Same weights can interpolate colors

$$\boxed{\mathbf{c}_i} = \sum_{j=1}^{H} w_j(\mathbf{x}_i)\boxed{\mathbf{c}_j}$$

Same functions used for color interpolation

unconstrained Δ^2
[Finch et al. 2011]

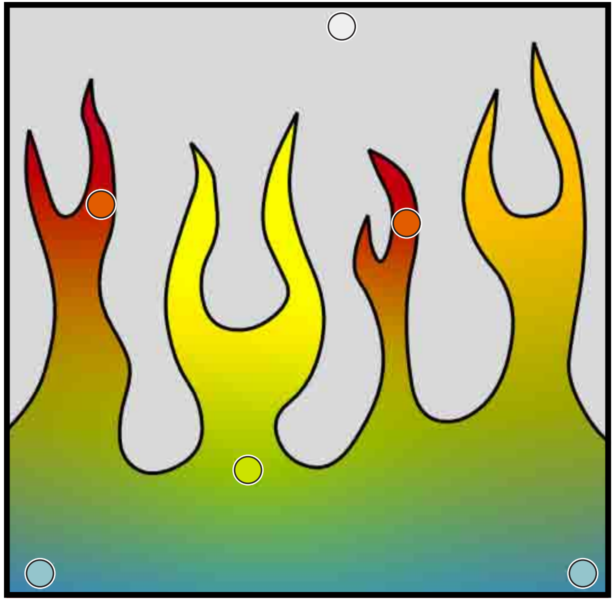
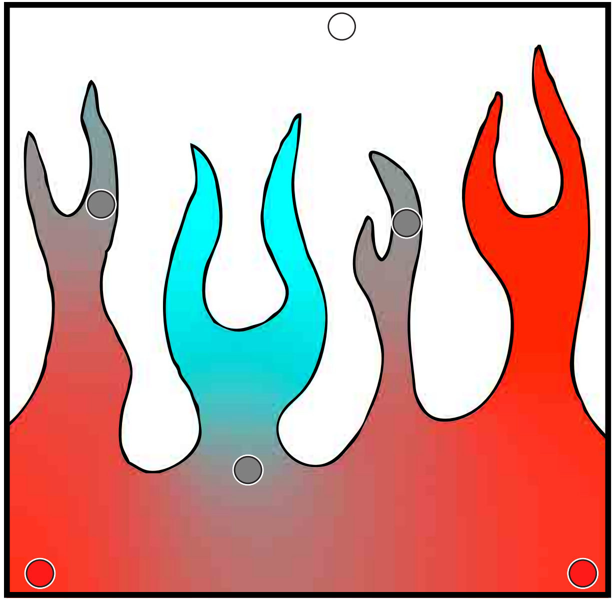


Image courtesy Mark Finch

$$\mathbf{c}_i = \sum_{j=1}^{H} w_j(\mathbf{x}_i)\mathbf{c}_j$$

Same functions used for color interpolation

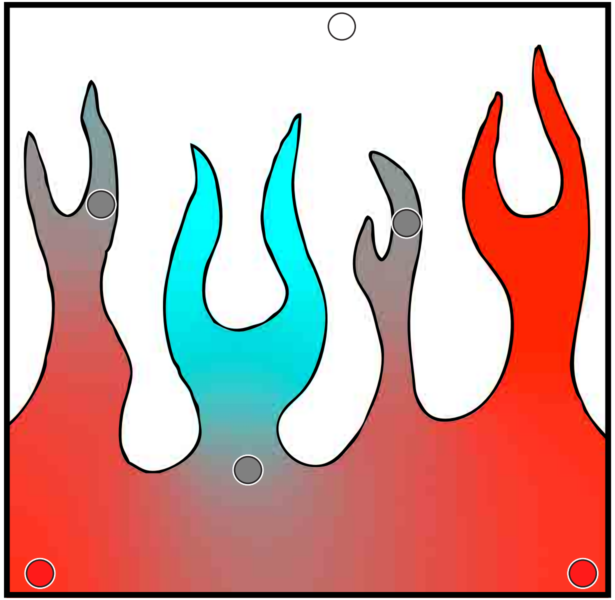
unconstrained Δ^2
[Finch et al. 2011]



$$\mathbf{c}_i = \sum_{j=1}^{H} w_j(\mathbf{x}_i)\mathbf{c}_j$$

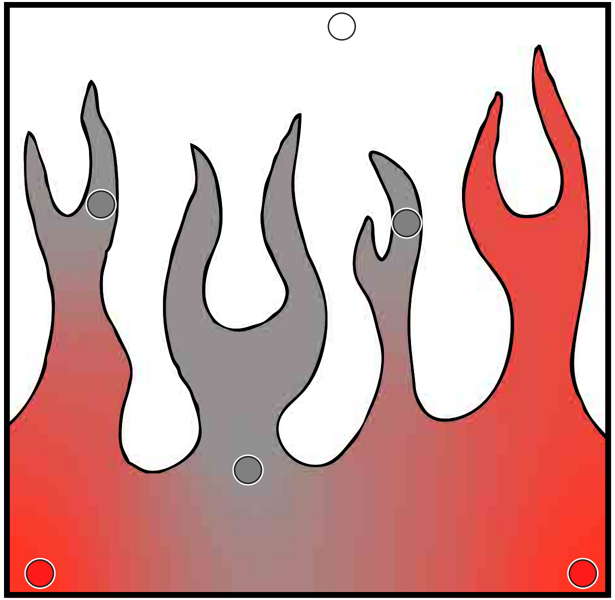
Same functions used for color interpolation

unconstrained Δ^2
[Finch et al. 2011]

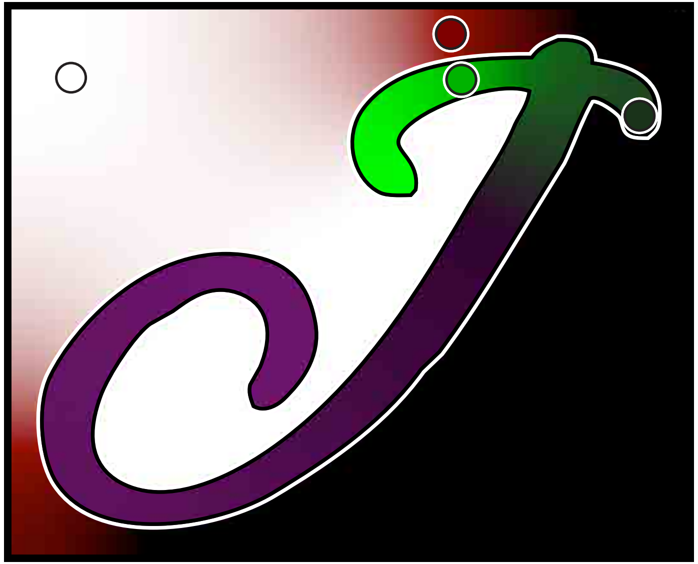


$$\mathbf{c}_i = \sum_{j=1}^{H} w_j(\mathbf{x}_i)\mathbf{c}_j$$

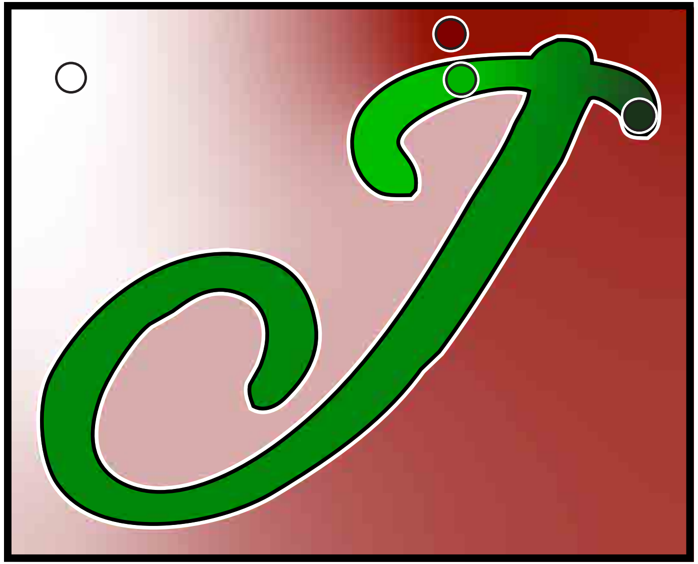
Our Δ^2
[SGP 2012]



Without constraints, biharmonic unintuitively interpolates colors

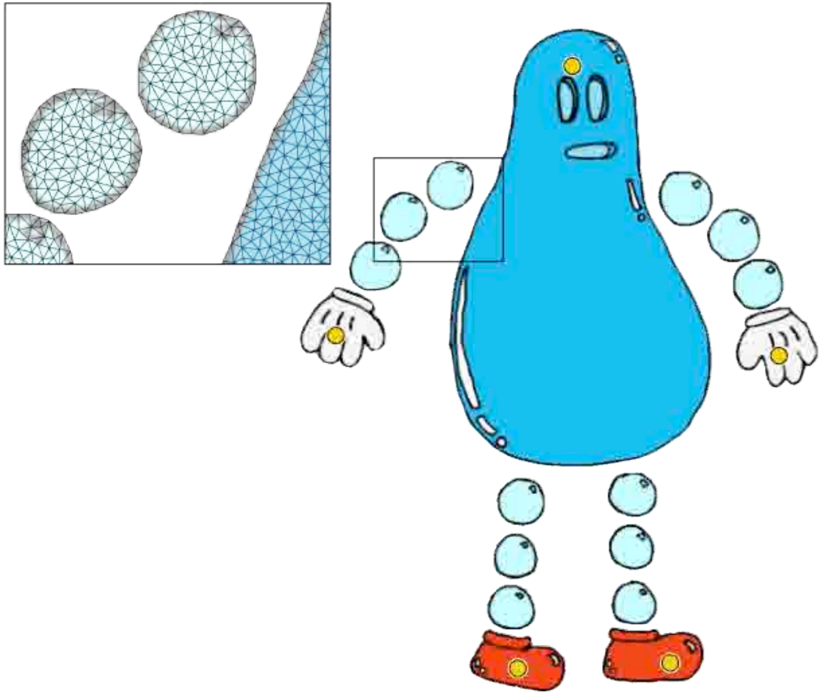


Unconstrained Δ^2
[Finch et al. 2011]



Our extended method
[SGP 2012]

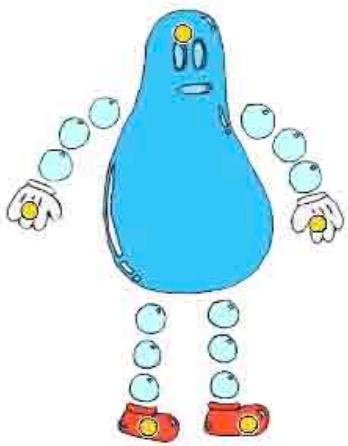
Skinning weights tether optimization over multiple-component meshes



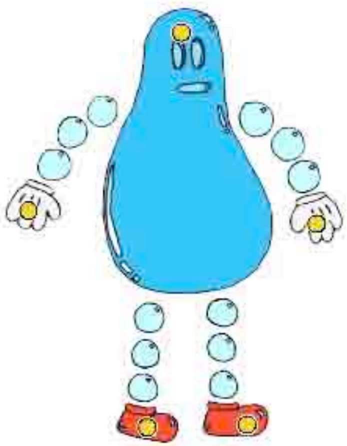
Skinning weights tether optimization over multiple-component meshes

Mesh-energy methods

e.g. Igarashi et al. 2004,
Sumner et al. 2005,
Botsch et al. 2006,
Sorkine and Alexa 2007,
Shi et al. 2007,
Solomon et al. 2011

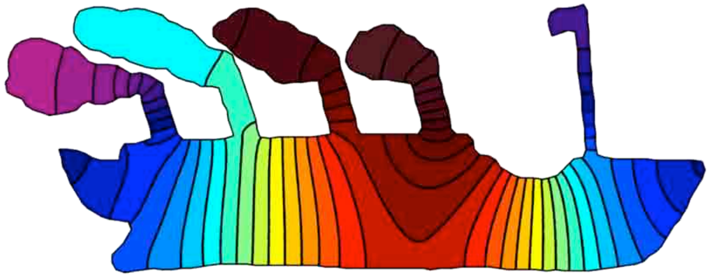


Our skinning-based method

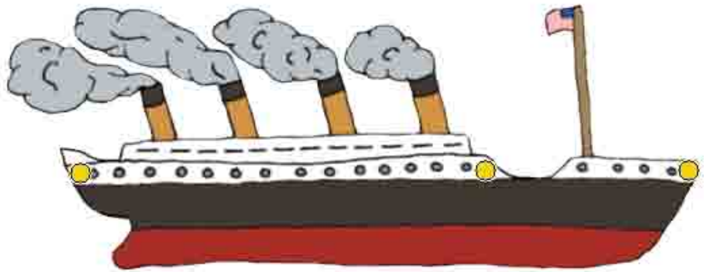


Extrema distort small features

Unconstrained Δ^2 [Botsch & Kobbelt 2004]

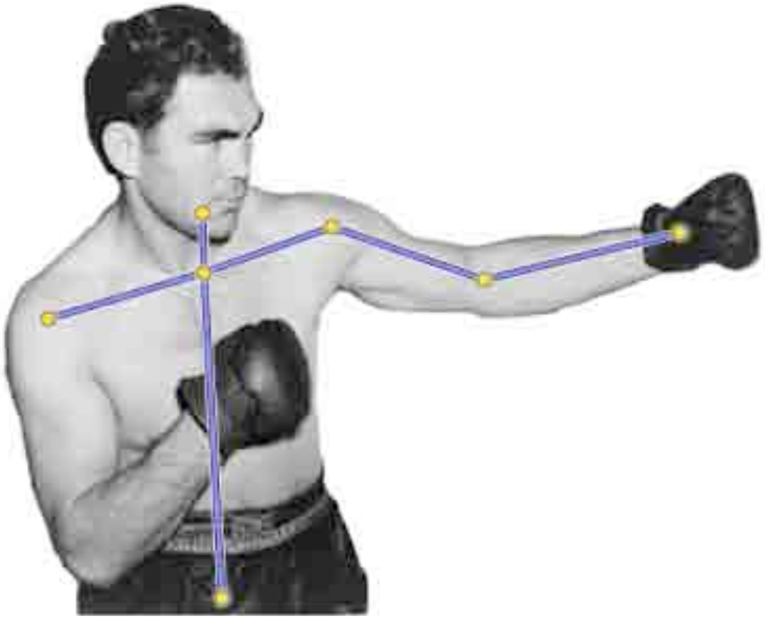


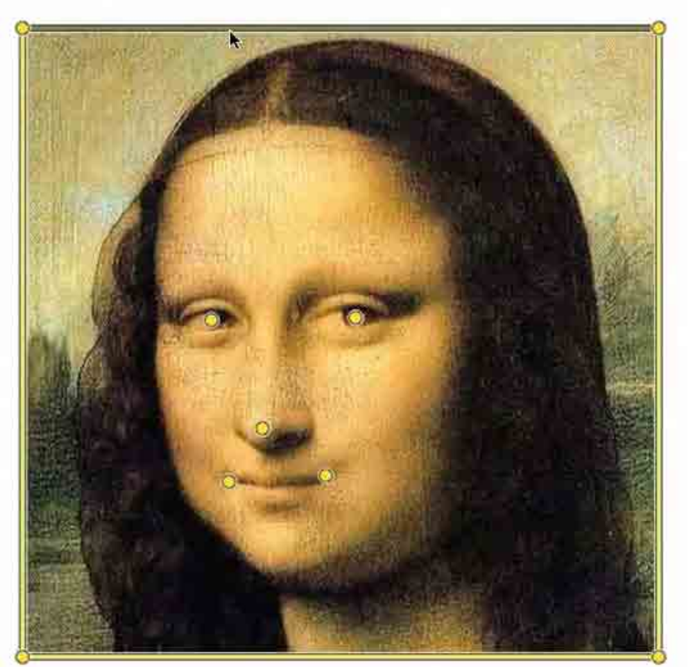
weight of middle point



Deformation applies to images as planar shapes

non-convex “cut-out” cartoons





entire image rectangle

Extra weights would expand subspace…

$$\mathbf{v}'_i = \sum_{j=1}^{m} w_j(\mathbf{v}_i)\mathbf{T}_j \begin{pmatrix}\mathbf{v}_i \\ 1\end{pmatrix}$$

$$\mathbf{V}' = \mathbf{M}\mathbf{T}$$

Extra weights would expand subspace…

$$\mathbf{v}'_i = \sum_{j=1}^{m} w_j(\mathbf{v}_i)\mathbf{T}_j \begin{pmatrix} \mathbf{v}_i \\ 1 \end{pmatrix} + \sum_{k=1}^{m_{\text{extra}}} w_k(\mathbf{v}_i)\mathbf{T}_k \begin{pmatrix} \mathbf{v}_i \\ 1 \end{pmatrix}$$

$$\mathbf{V}' = \mathbf{M}\mathbf{T}$$

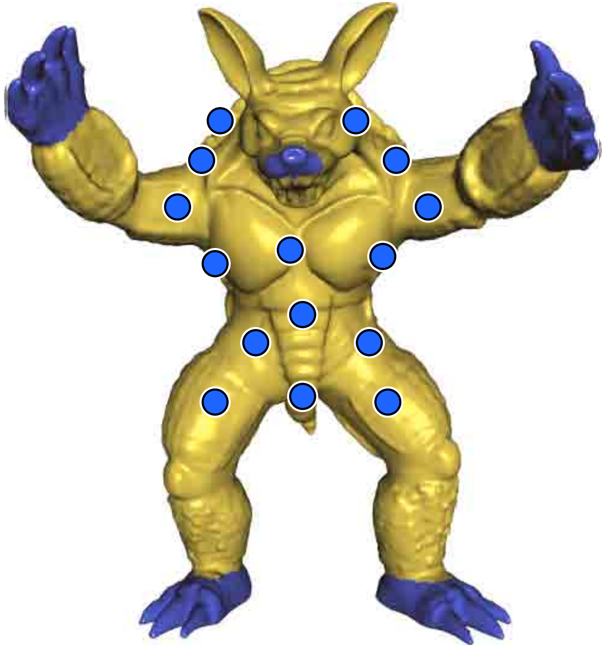
Extra weights would expand subspace...

$$\mathbf{v}'_i = \sum_{j=1}^{m} w_j(\mathbf{v}_i)\mathbf{T}_j \begin{pmatrix} \mathbf{v}_i \\ 1 \end{pmatrix} + \sum_{k=1}^{m_{\text{extra}}} w_k(\mathbf{v}_i)\mathbf{T}_k \begin{pmatrix} \mathbf{v}_i \\ 1 \end{pmatrix}$$

$$\mathbf{V}' = \mathbf{M}\mathbf{T} + \mathbf{M}_{\text{extra}}\mathbf{T}_{\text{extra}}$$

Real-time automatic degrees of freedom

Overlapping b-spline "bumps" in weight space



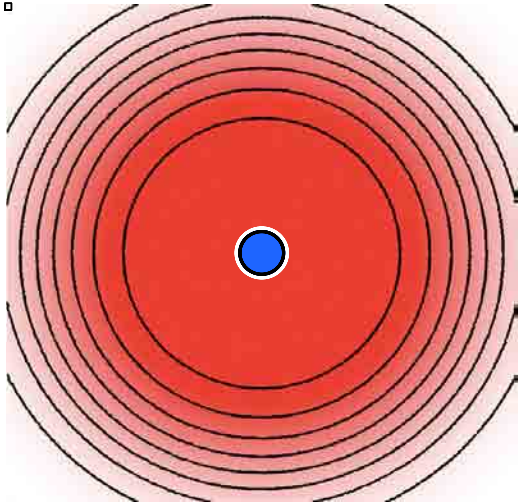
farthest point sampling

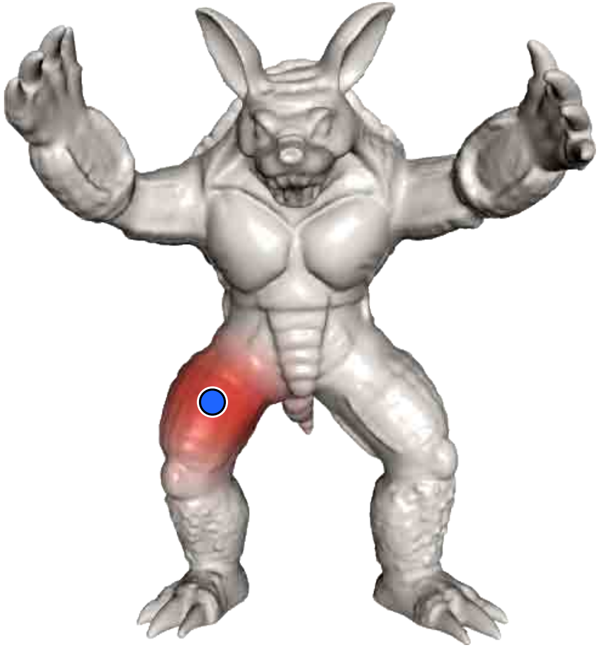
weight space

$$\mathbf{x}_j = \begin{bmatrix} w_1(\mathbf{v}_j) \\ w_2(\mathbf{v}_j) \\ \vdots \\ w_m(\mathbf{v}_j) \end{bmatrix}$$

Overlapping b-spline “bumps” in weight space

in weight space





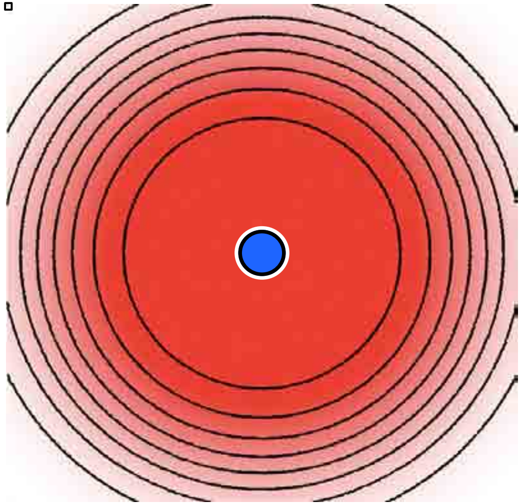
weight space

$$\mathbf{x}_j = \begin{bmatrix} w_1(\mathbf{v}_j) \\ w_2(\mathbf{v}_j) \\ \vdots \\ w_m(\mathbf{v}_j) \end{bmatrix}$$

b-spline basis parameterized by distance in weight space

Overlapping b-spline "bumps" in weight space

in weight space



weight space

$$\mathbf{x}_j = \begin{bmatrix} w_1(\mathbf{v}_j) \\ w_2(\mathbf{v}_j) \\ \vdots \\ w_m(\mathbf{v}_j) \end{bmatrix}$$

b-spline basis parameterized by distance in weight space

Final algorithm is simple and FAST

Precomputation per shape+rig *For a 50K triangle mesh:*

- Compute any additional weights *12 seconds*
- Construct, prefactor system matrices *2.7 seconds*

Final algorithm is simple and FAST

Precomputation per shape+rig	*For a 50K triangle mesh:*
- Compute any additional weights	*12 seconds*
- Construct, prefactor system matrices	*2.7 seconds*
Precomputation when switching constraint type	
- *Re*-factor global step system	*6 milliseconds*

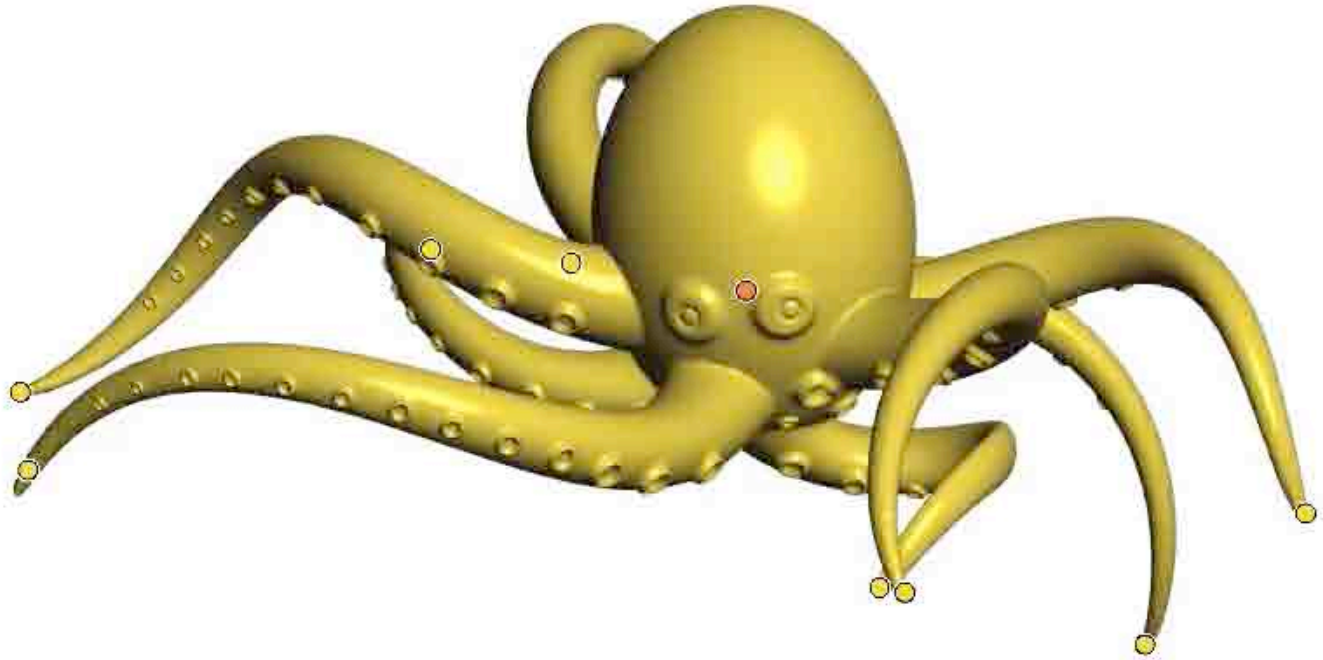
Final algorithm is simple and FAST

Precomputation per shape+rig	*For a 50K triangle mesh:*
- Compute any additional weights	*12 seconds*
- Construct, prefactor system matrices	*2.7 seconds*
Precomputation when switching constraint type	
- *Re*-factor global step system	*6 milliseconds*
~30 iterations	*22 microseconds*

global: #weights by #weights linear solve

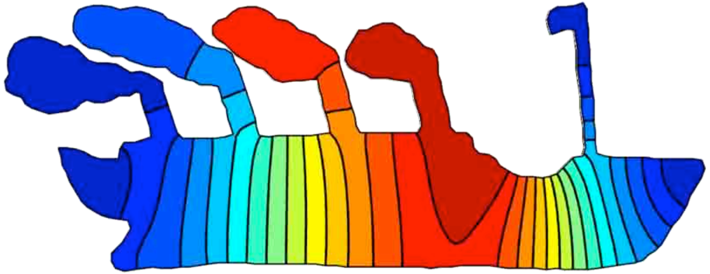
local: #rotations SVDs [McAdams et al. 2011]

Simple drag-only interface for point handles

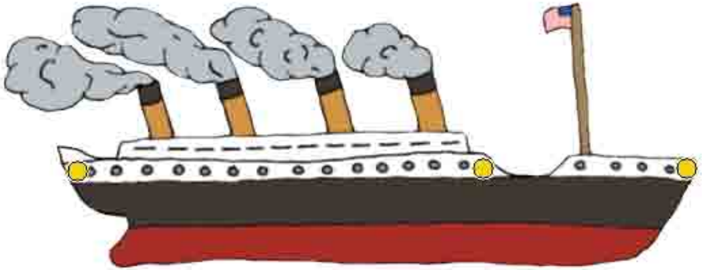


Extrema distort small features

Bounded Δ^2 [Jacobson et al. 2011]



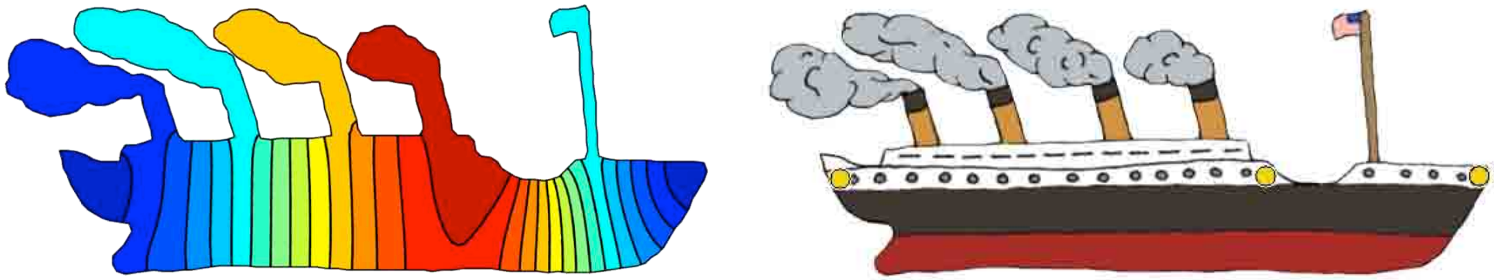
weight of middle point



“Monotonicity” helps preserve small features

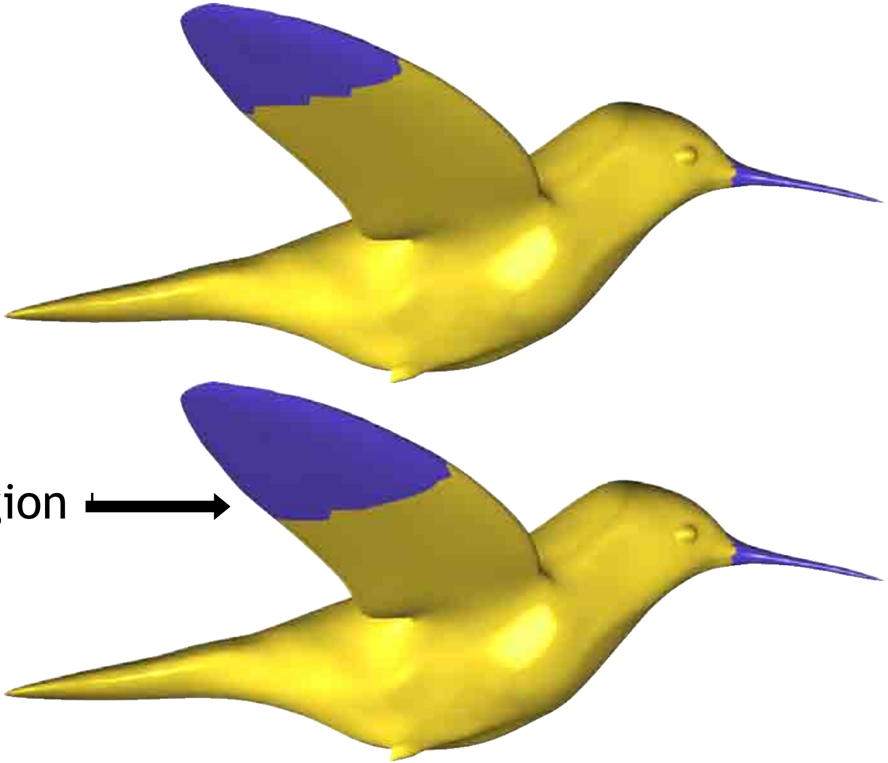
Bounded Δ^2 [Jacobson et al. 2011]

Our Δ^2

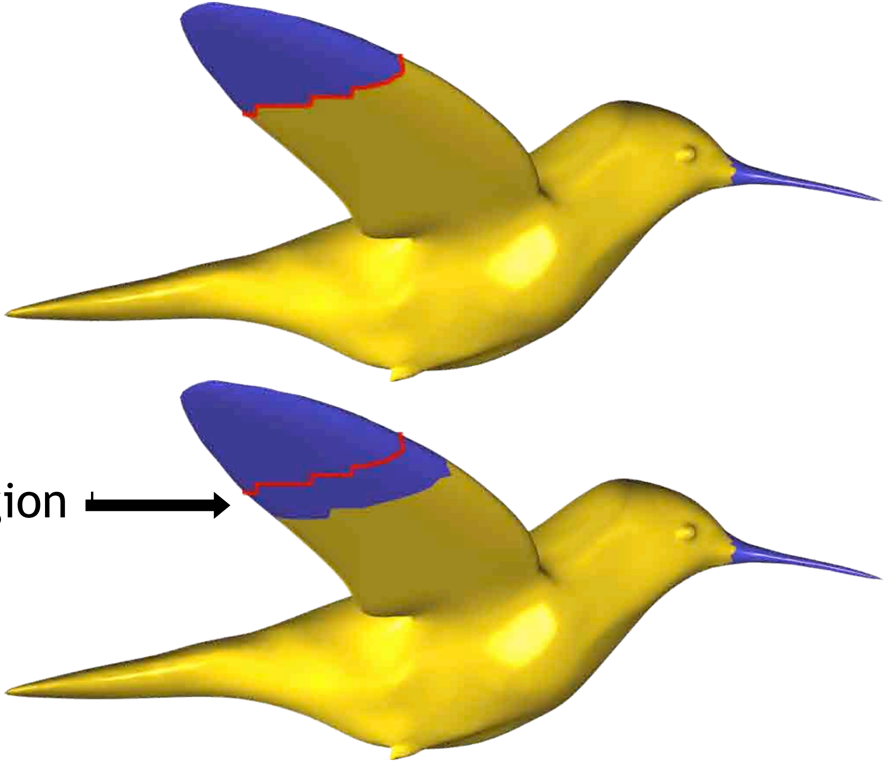


Spurious extrema are unstable, may "flip"

slightly larger region

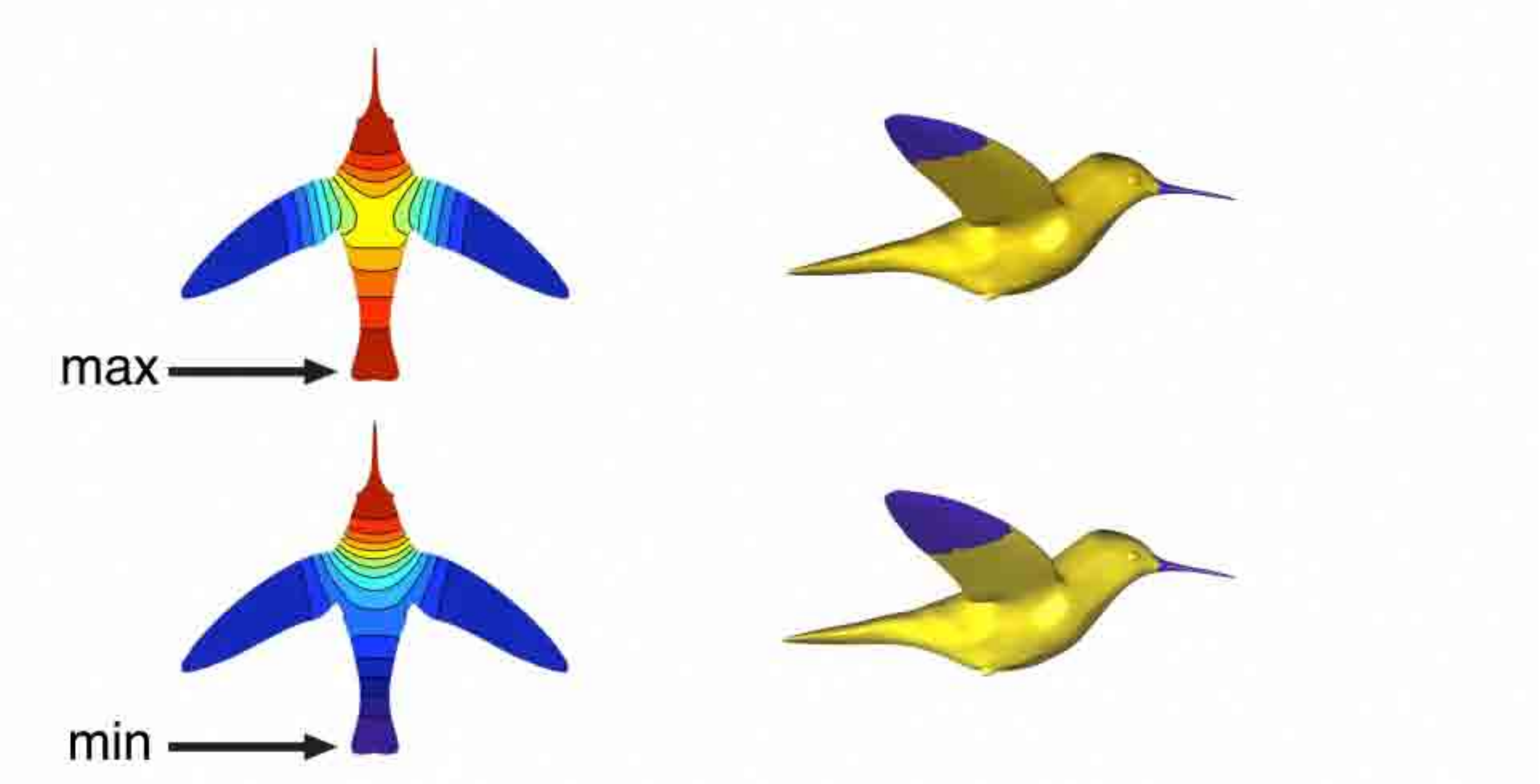


Spurious extrema are unstable, may "flip"



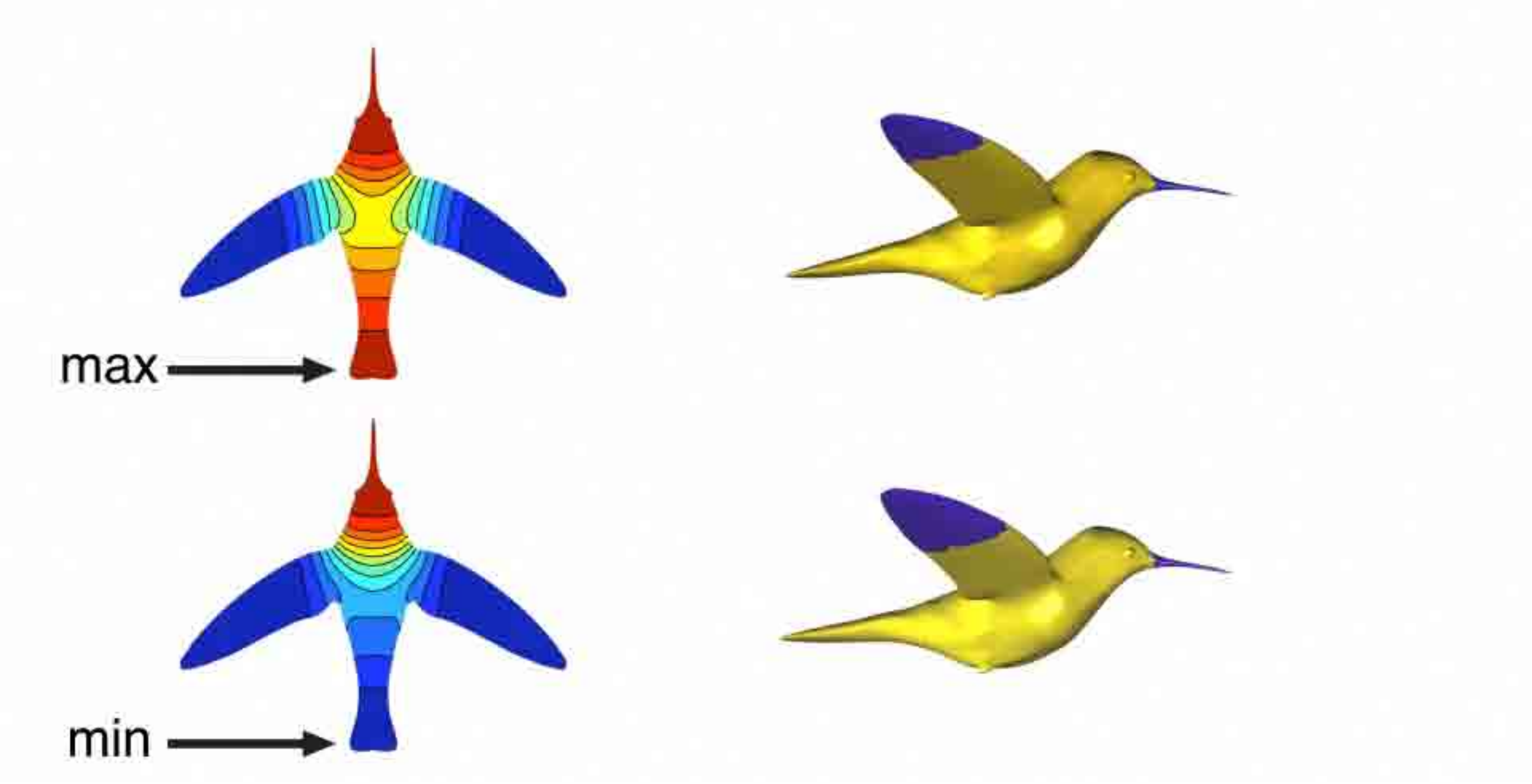
slightly larger region

Spurious extrema are unstable, may “flip”



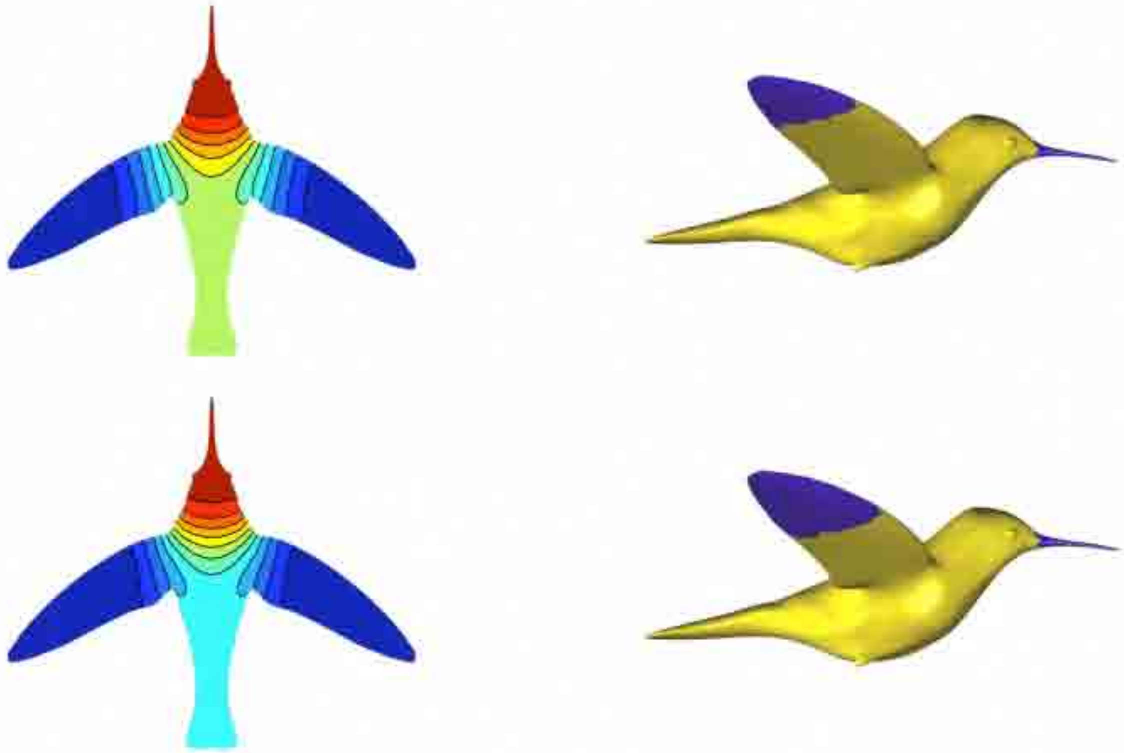
Unconstrained Δ^3 [Botsch & Kobbelt, 2004]

Spurious extrema are unstable, may “flip”



Bounded Δ^3

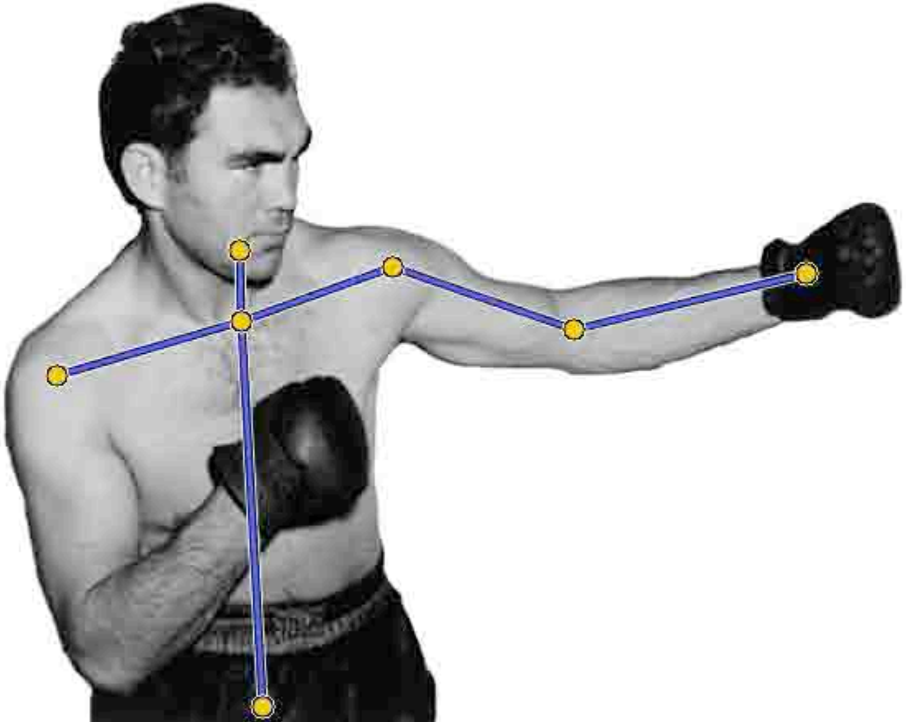
Lack of extrema leads to more stability



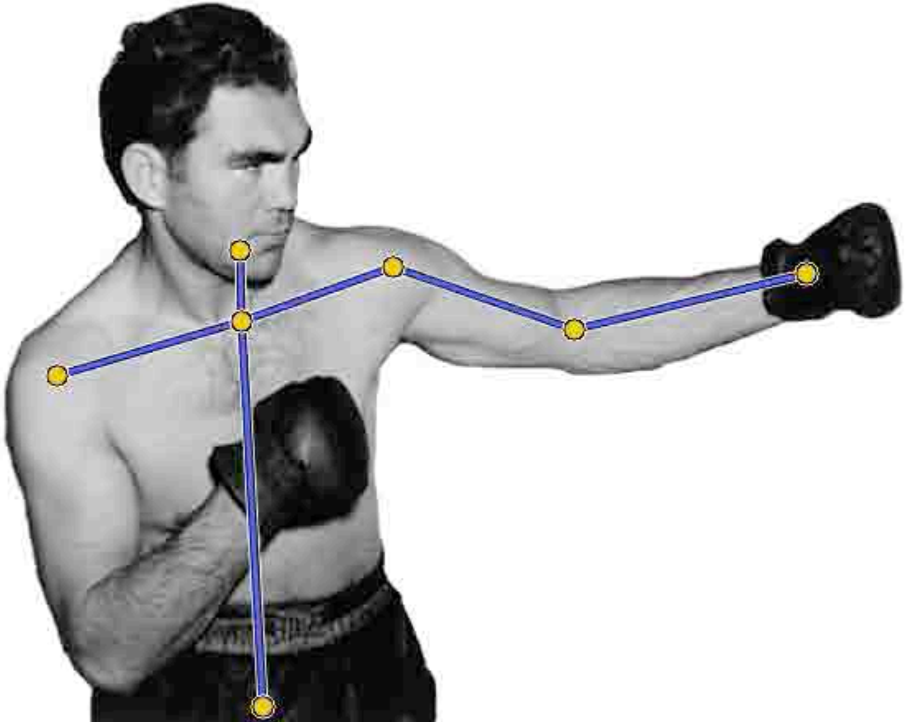
Our Δ^3

In 2D, stretching manipulates foreshortening

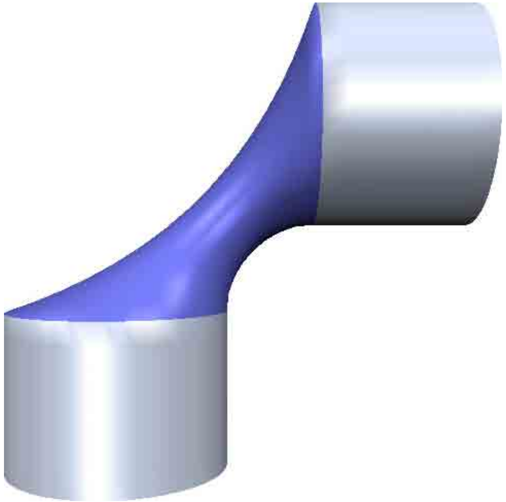
LBS with rigid bones



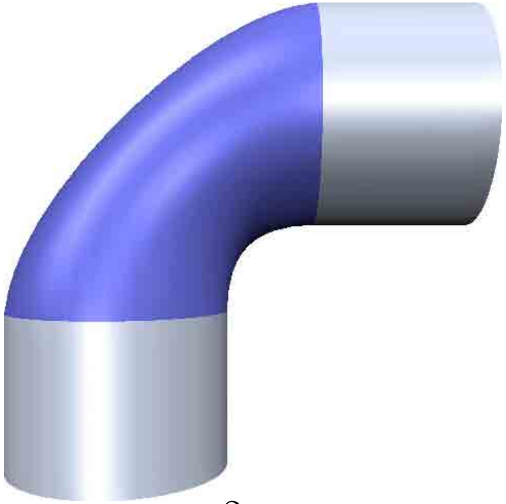
STBS with *stretchable* bones



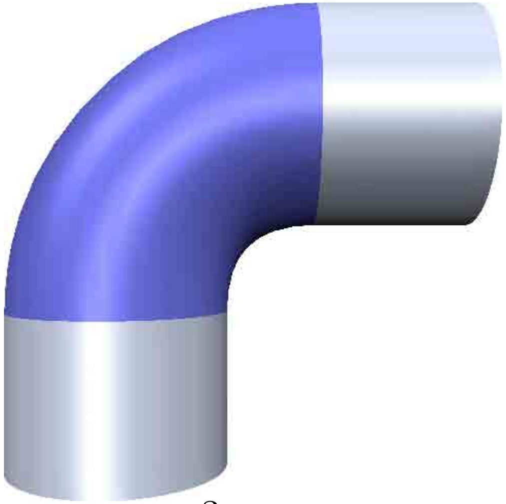
Behavior depends heavily on parameterization and boundary conditions



$\Delta \mathbf{x} = 0$

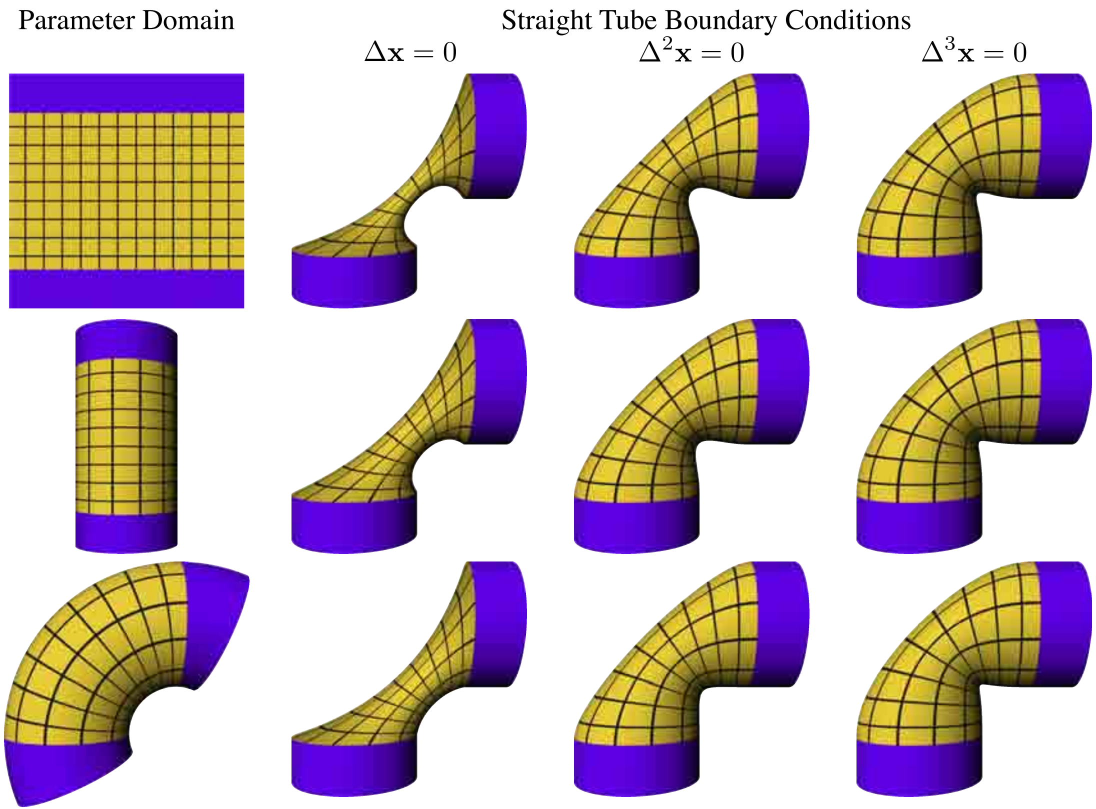


$\Delta^2 \mathbf{x} = 0$

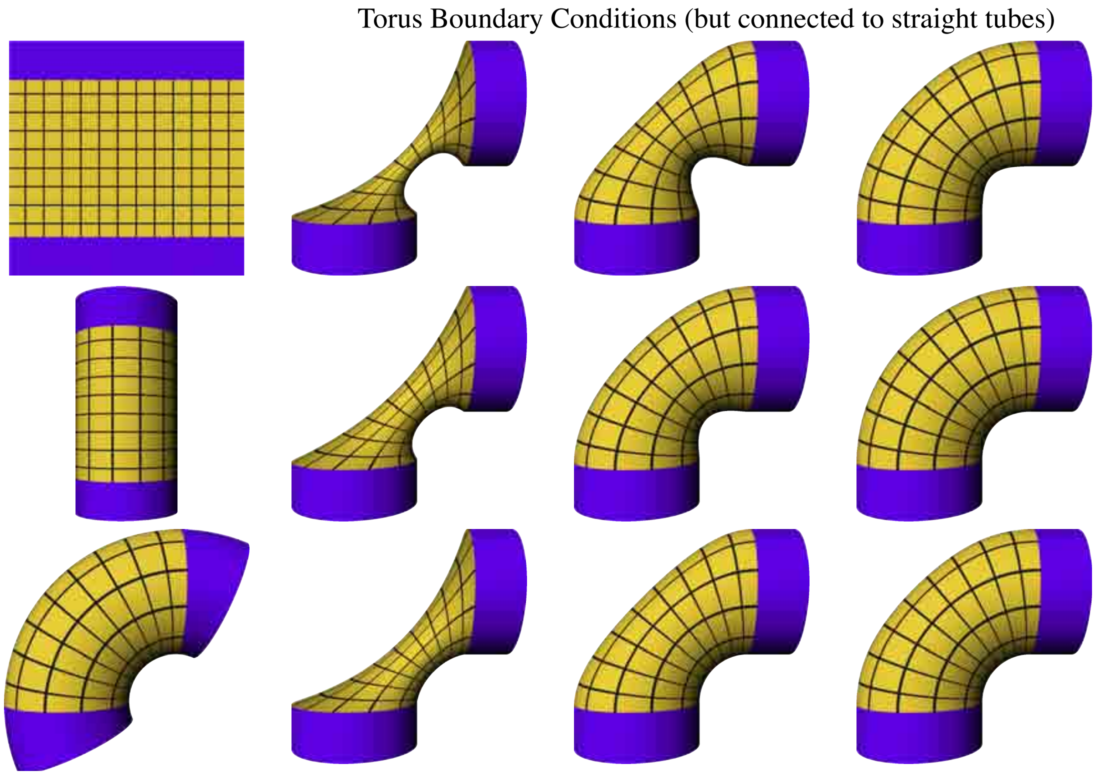


$\Delta^3 \mathbf{x} = 0$

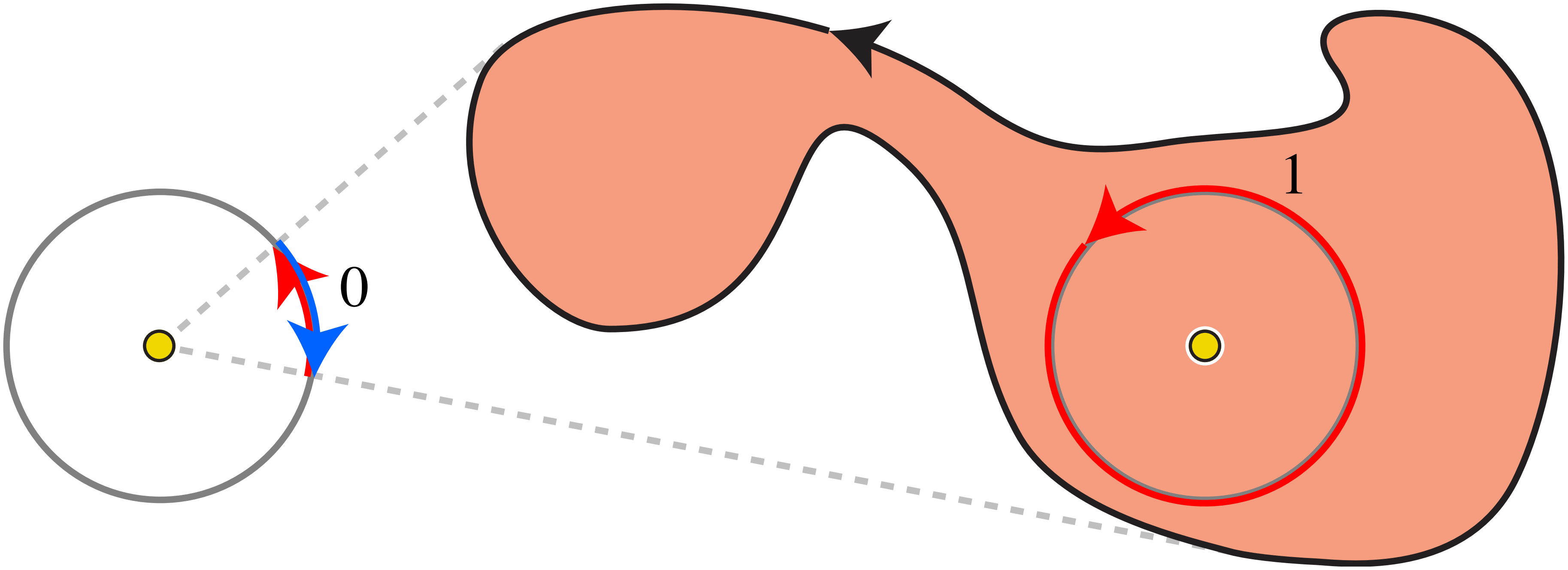
Behavior depends heavily on parameterization and boundary conditions



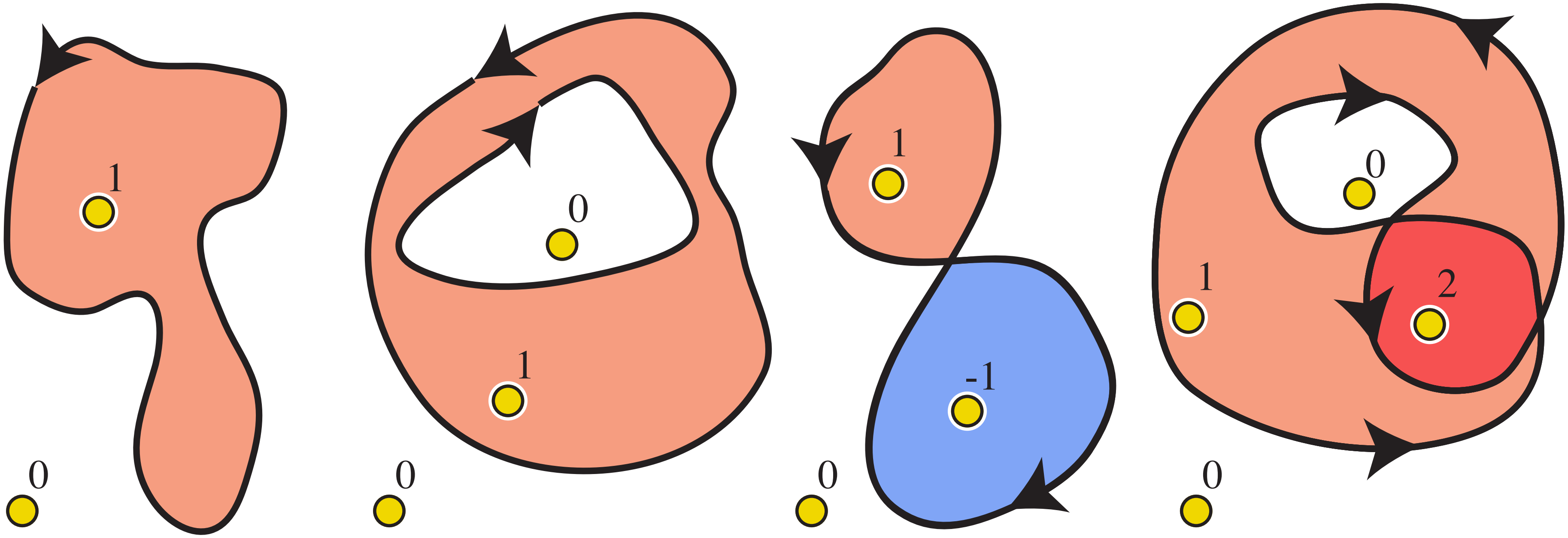
Behavior depends heavily on parameterization and boundary conditions



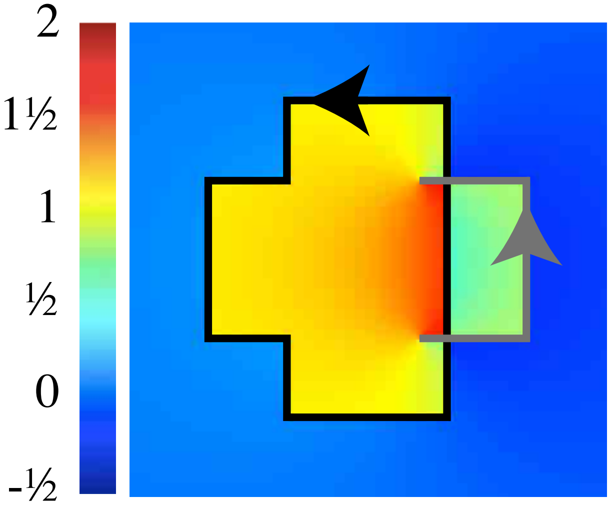
Traditional winding number determines *amount of insideness*

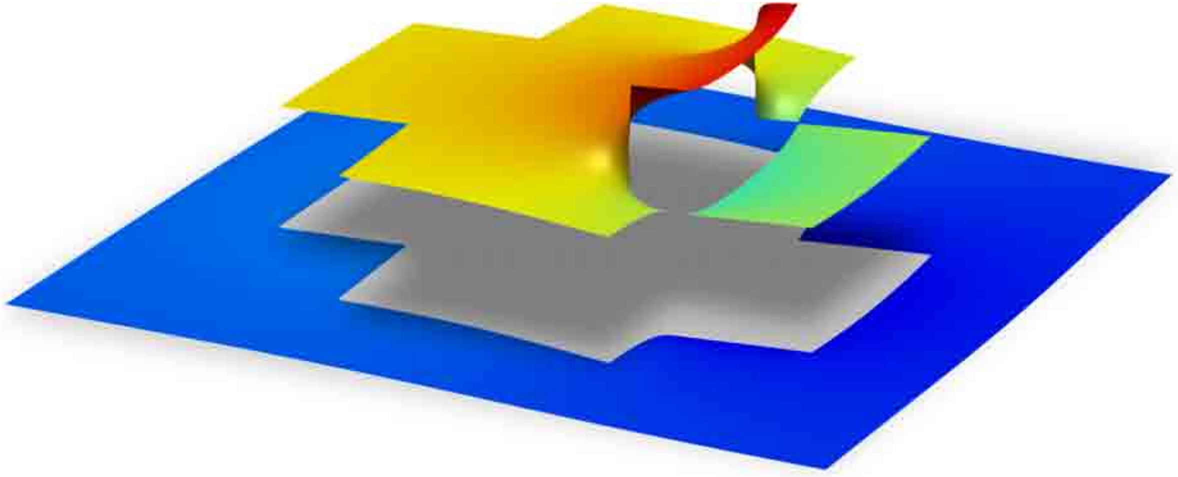


Traditional winding number determines *amount of insideness*

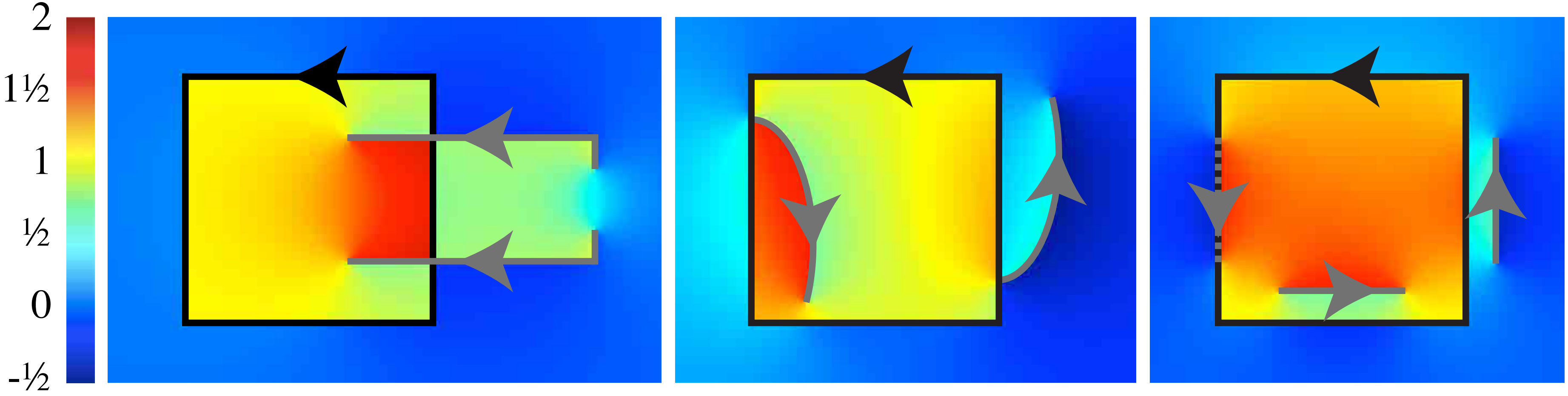


Winding number jumps across boundaries, harmonic otherwise

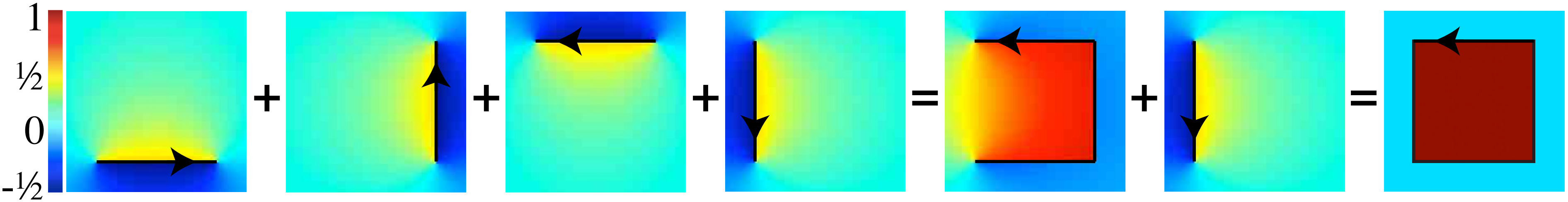




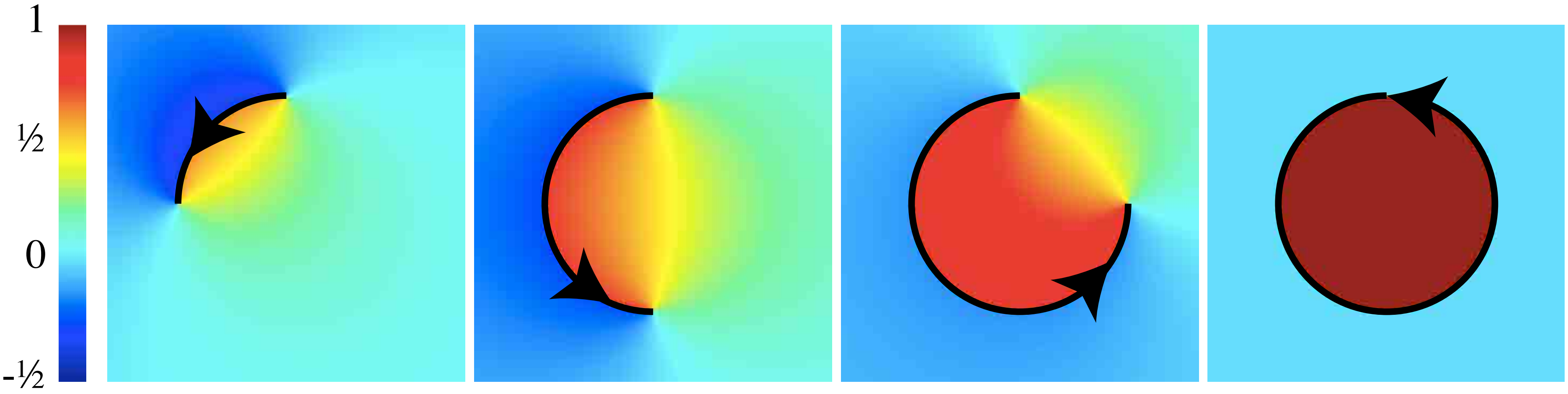
Winding number jumps across boundaries, harmonic otherwise



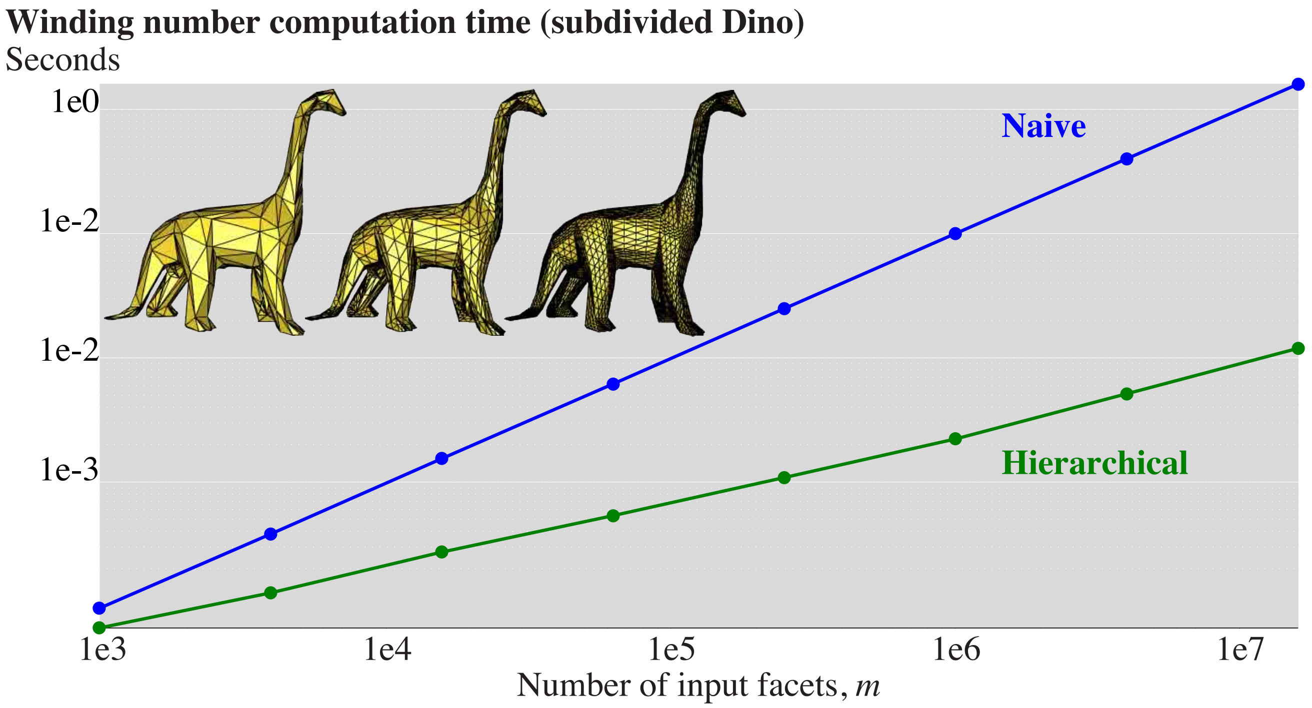
Sum of harmonic functions is harmonic



Winding number gracefully tends toward indicator function



Hierarchical evaluation performs asymptotically better



Hierarchical evaluation performs asymptotically better