Pose Space Deformation

A unified Approach to Shape Interpolation and Skeleton-Driven Deformation

J.P. Lewis

Matt Cordner

Nickson Fong

Presented by Simone Croci

Talk Outline

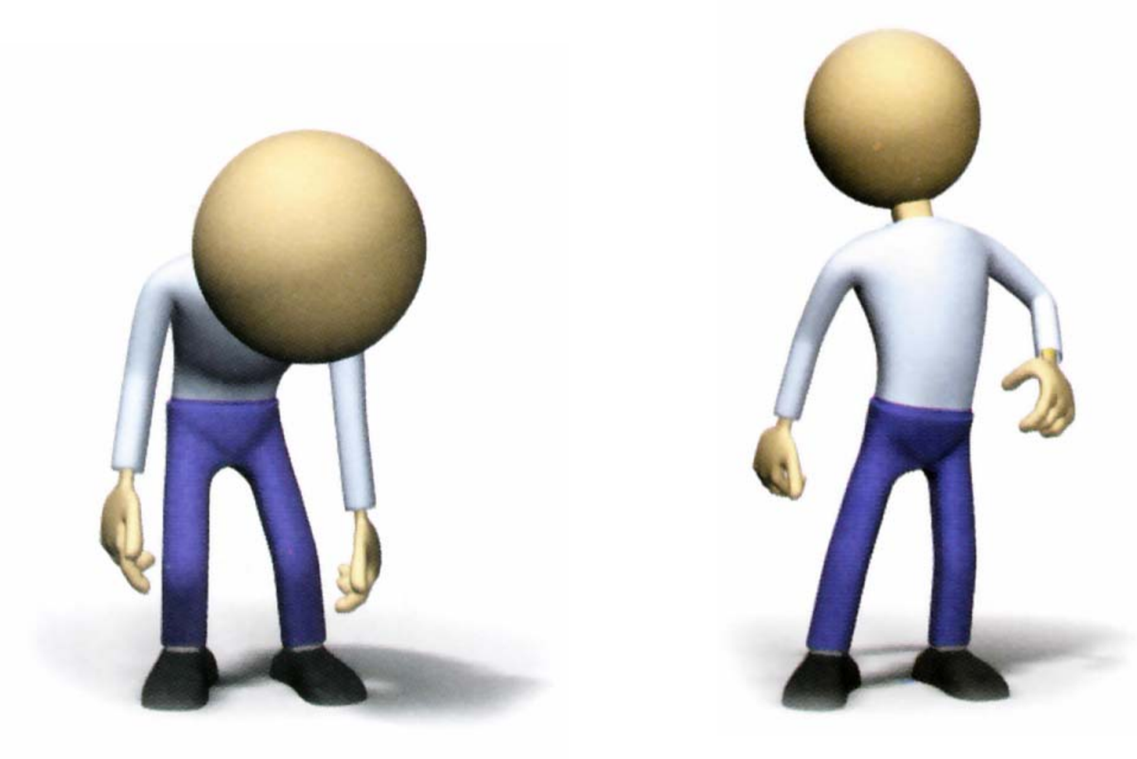
- Character Animation
  - Overview
  - Problem Statement
- Background
  - Skeleton-Subspace Deformation
  - Shape Interpolation
- Pose Space Deformation
  - Deformation Model
  - Evaluation
  - Related Works

Character Animation

Overview

Animation:

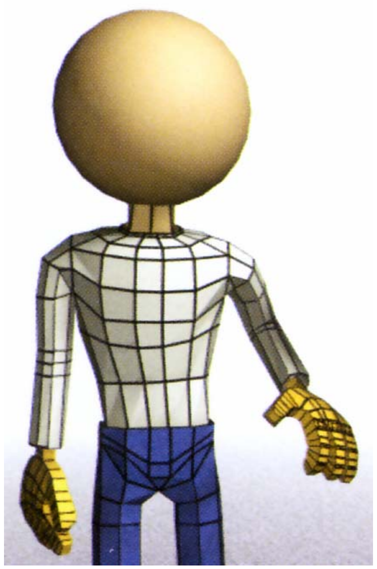
“To give a soul to a lifeless character”



Character Animation

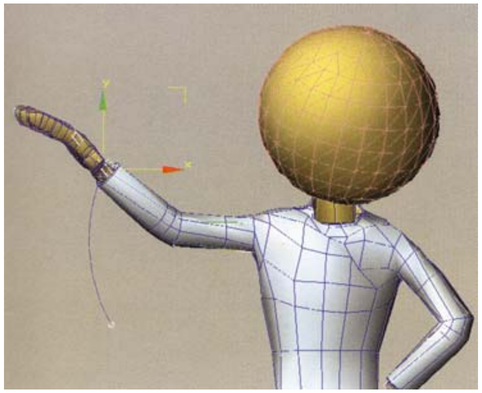
Problem Statement

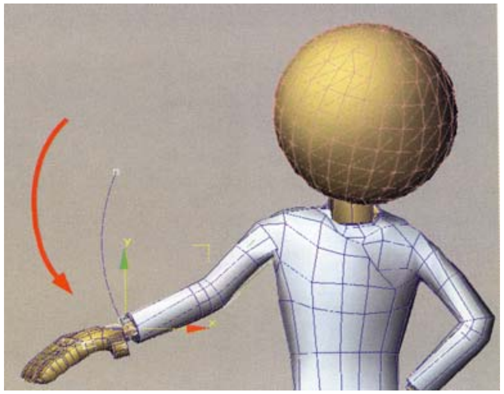
Main Components:



1. **Character Model**

2. **Deformation model**

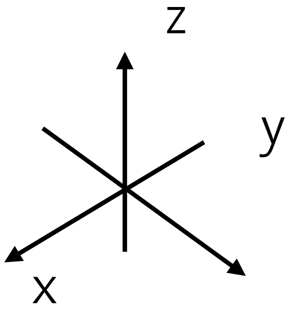




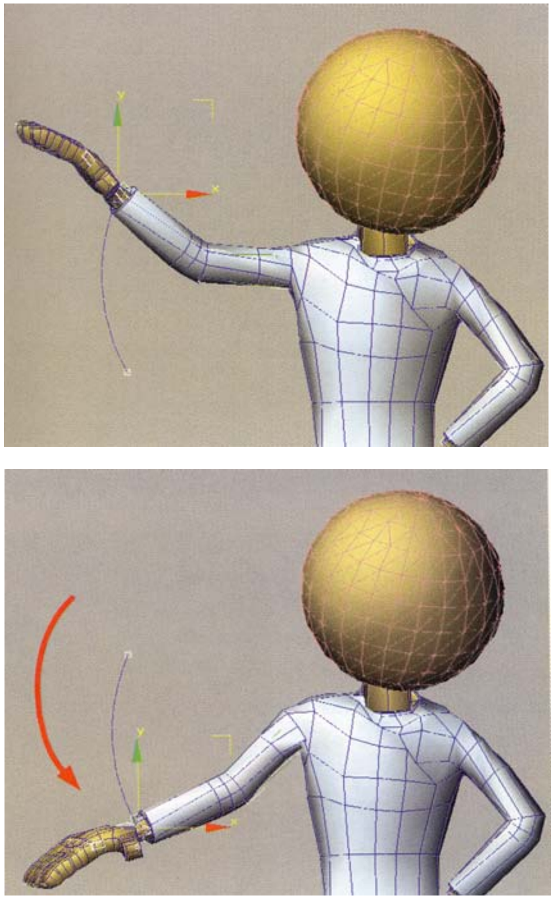
Deformation Model

Complexity:

∀ vertex: 3 DoFs



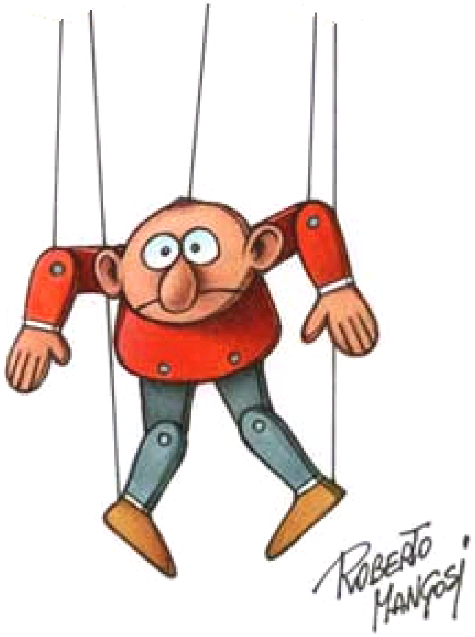
=> Reduction of DoFs



Deformation Model

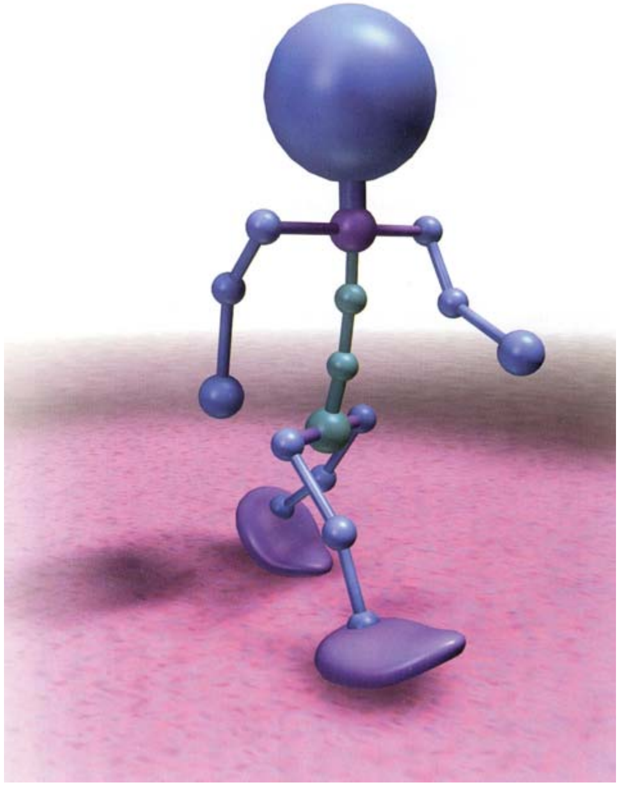
Mapping:

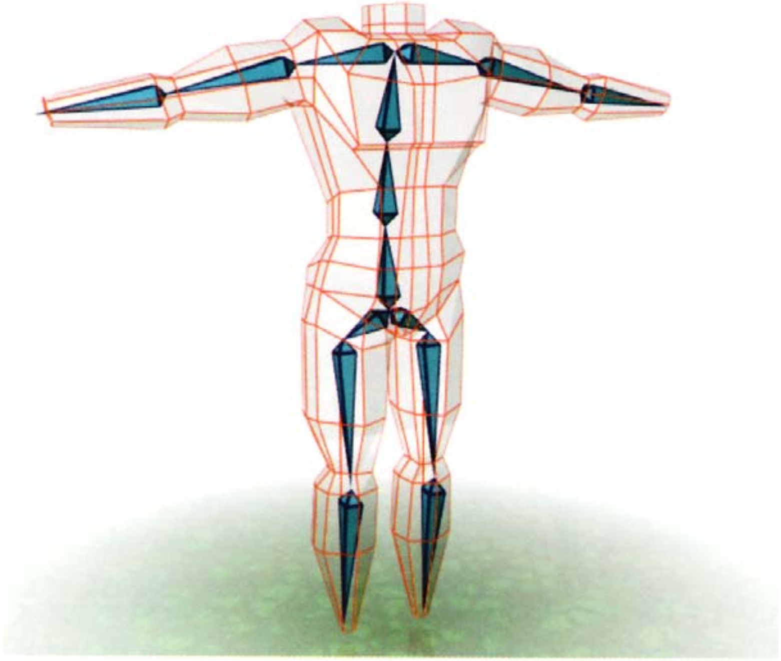
Parameters => Deformation



Deformation Model

Joint rotations of a skeleton



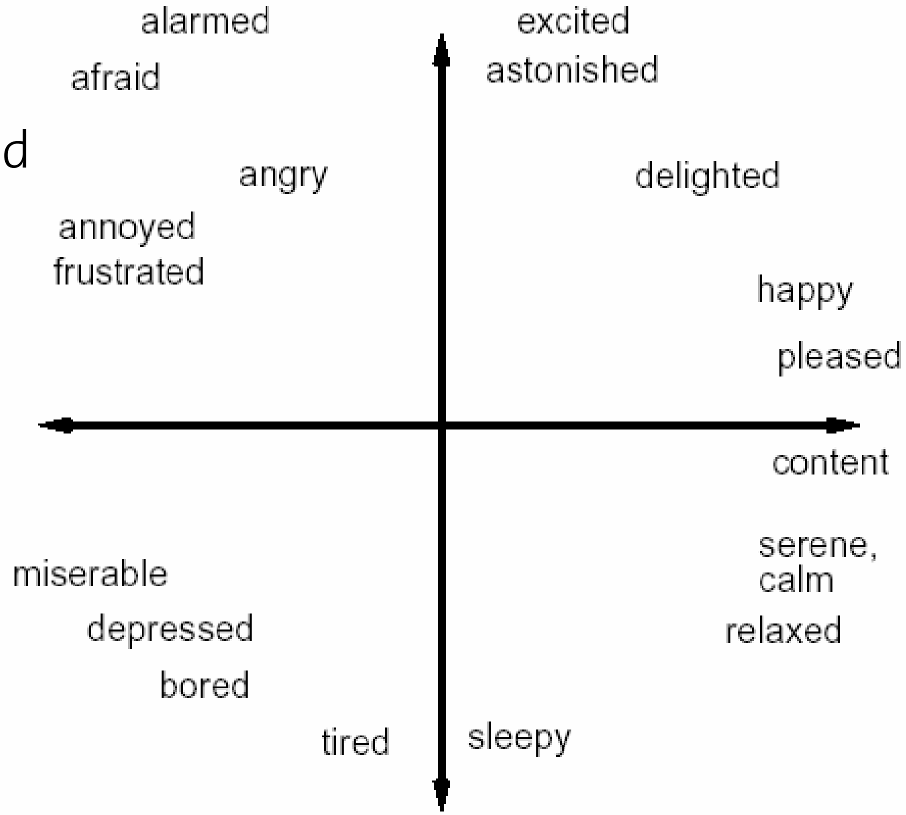


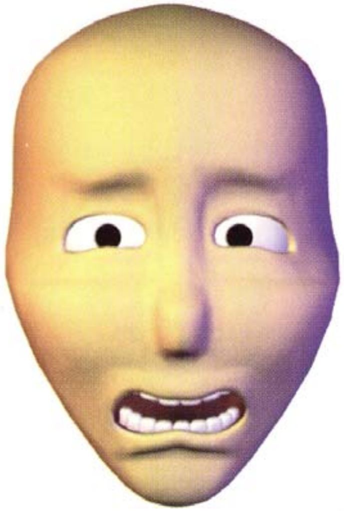
Character Animation

Deformation Model

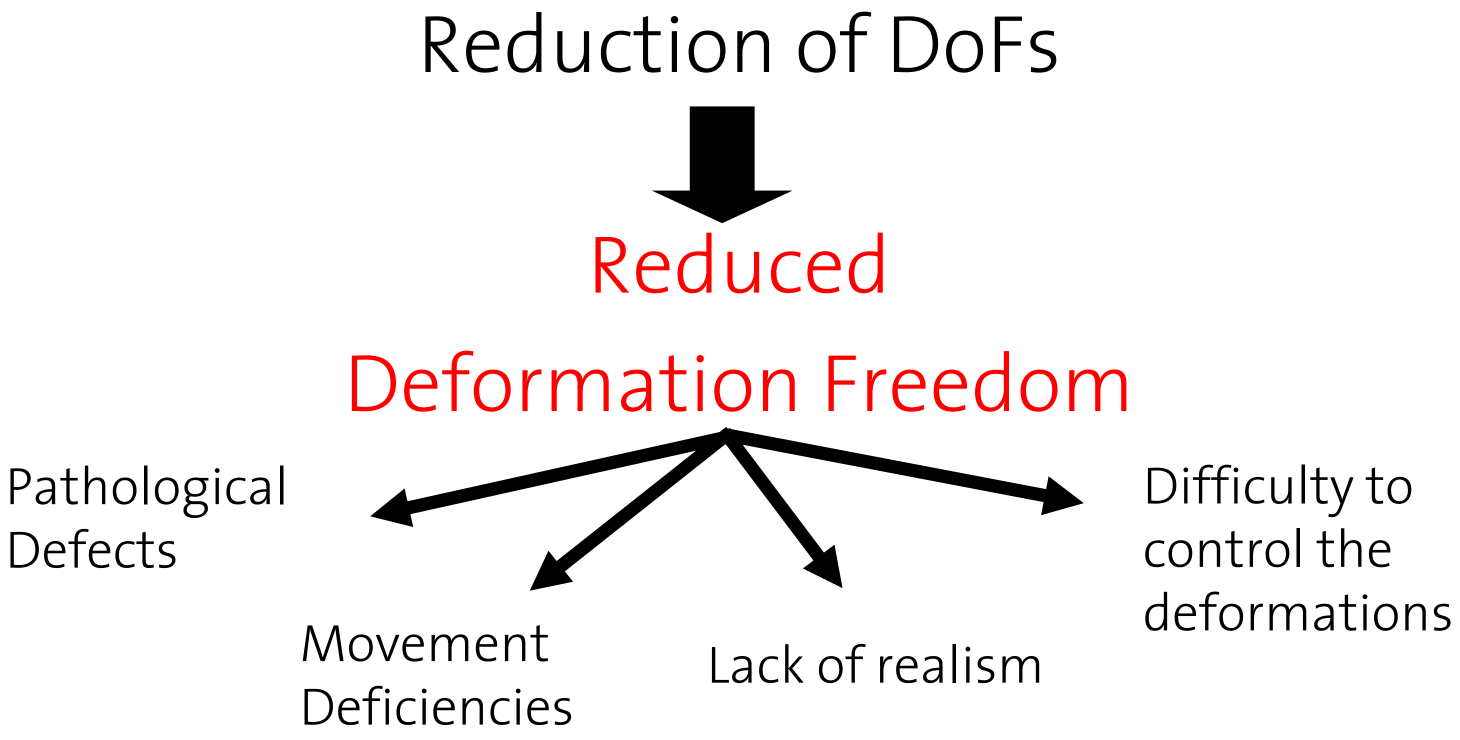
Emotional Axes

X=Sad/Happy
Y=Relaxed/ Excited





Deformation Model



Reduction of DoFs

Reduced

Deformation Freedom

Pathological Defects

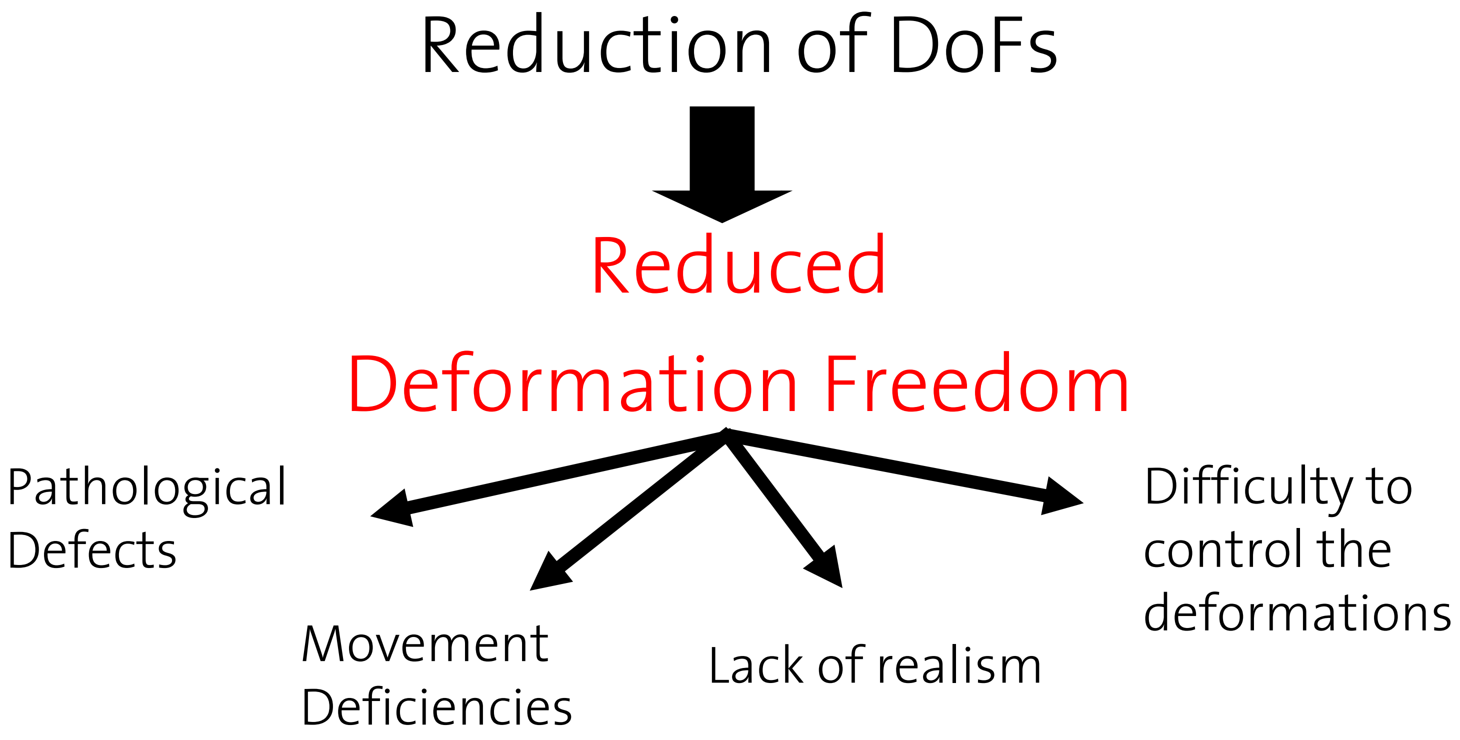
Movement Deficiencies

Lack of realism

Difficulty to control the deformations

Character Animation

Deformation Model



Reduction of DoFs

Reduced

Deformation Freedom

Pathological Defects

Movement Deficiencies

Lack of realism

Difficulty to control the deformations

Pose Space Deformation **solves these problems**

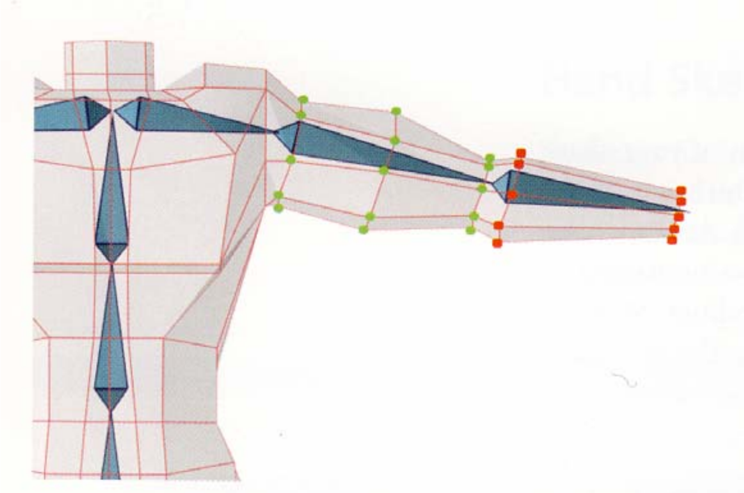
retaining few DoFs

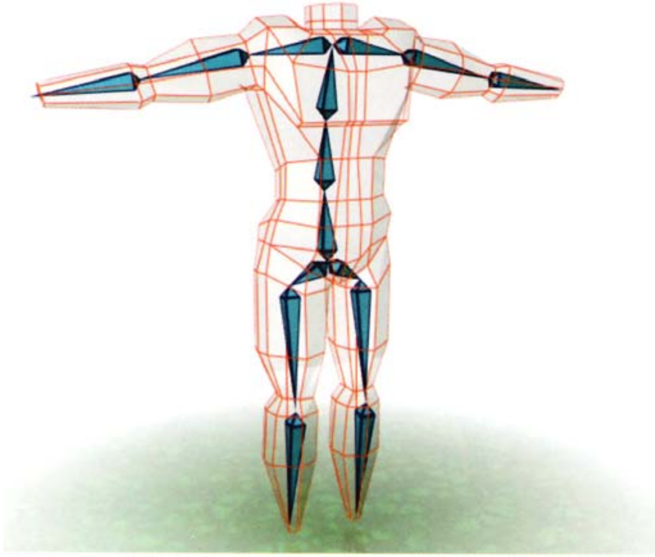
Skeleton-Subspace Deformation

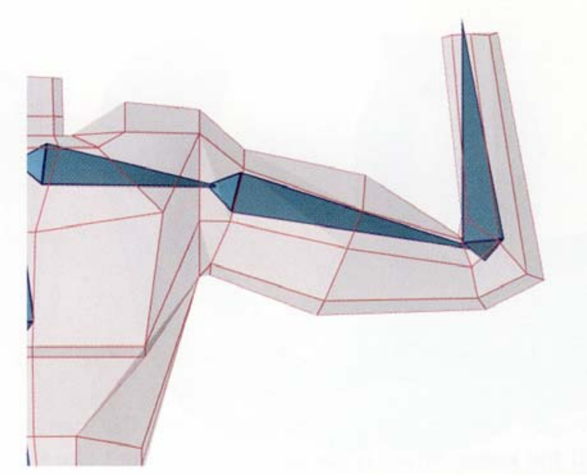
...also called skinning, enveloping

Deformation Model

- Skeleton => Vertex Position



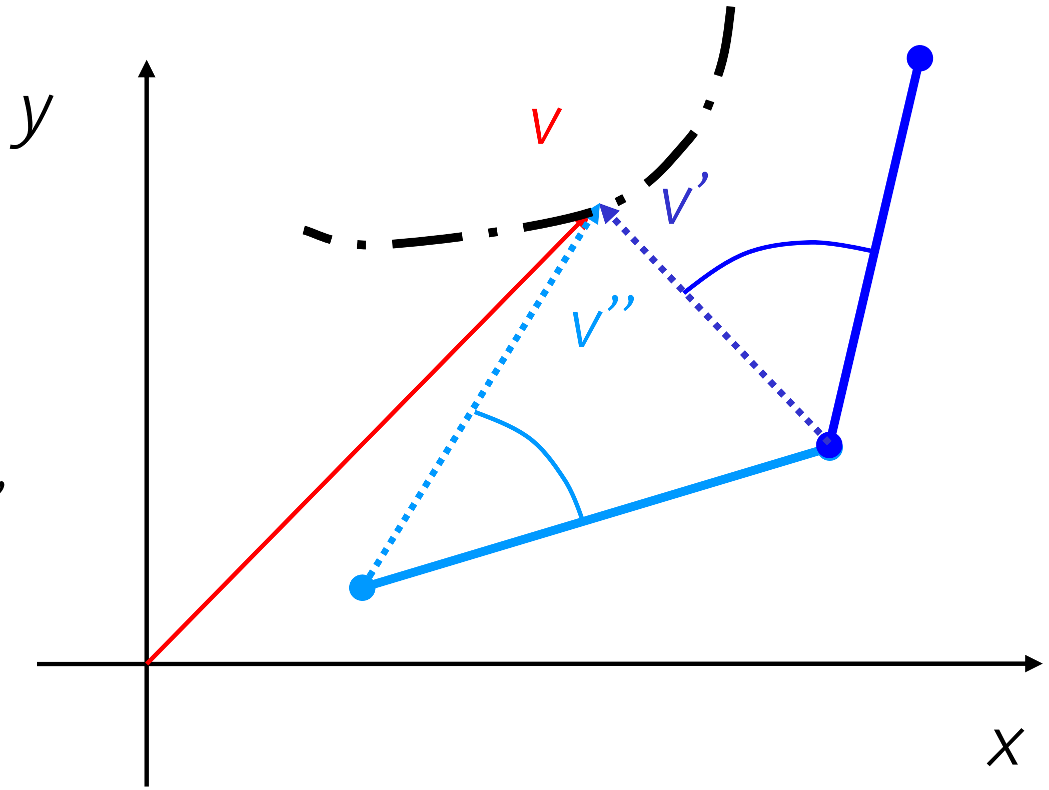




Skeleton-Subspace Deformation

Initial pose

1. Local coordinates v’,v’’
2. Vertex-joint weights: w`, w``

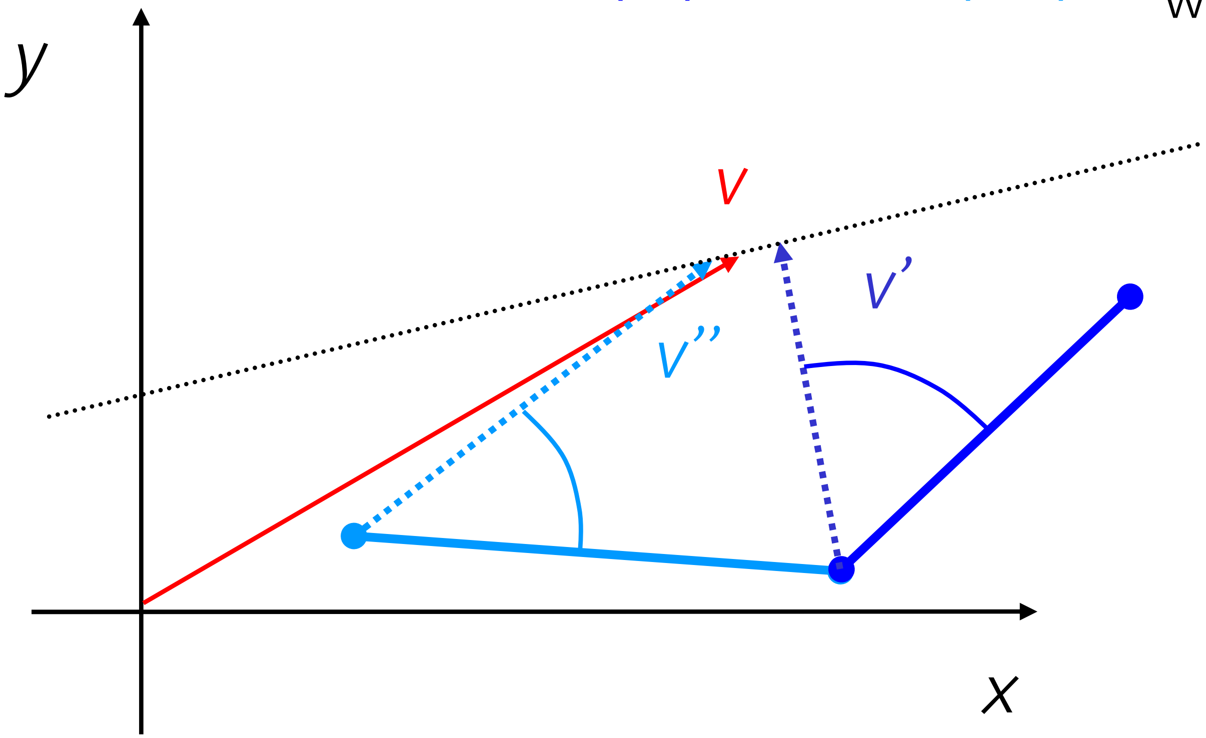


Skeleton-Subspace Deformation

New pose

v = w` T’(v`)+ w`` T’’(v``)

T from local to world coordinates



Skeleton-Subspace Deformation

Pros

- Simple
- Smooth deformations
- Low memory requirements
- Real-time deformations

Skeleton-Subspace Deformation

Cons

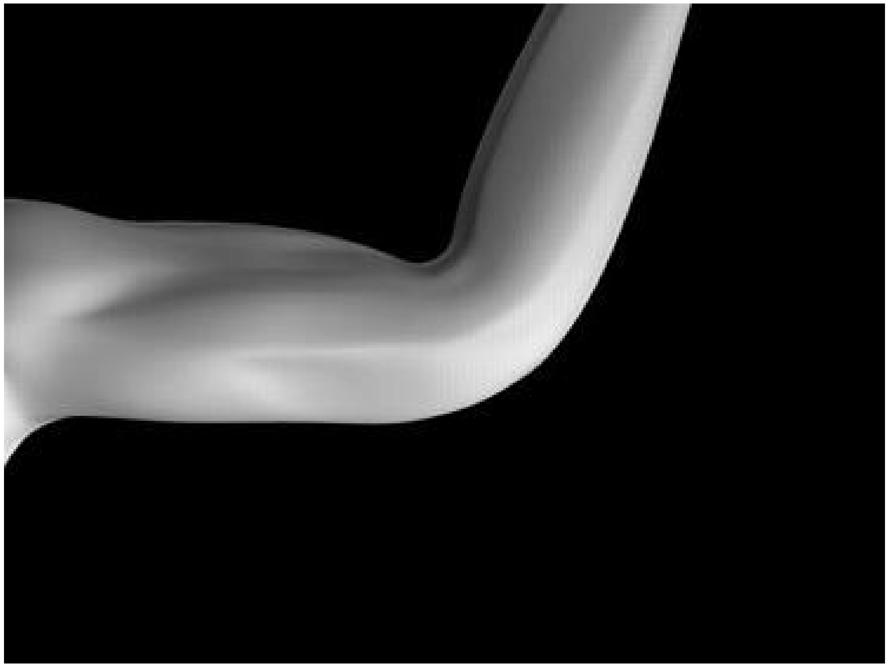
- Indirect control on deformations through weights

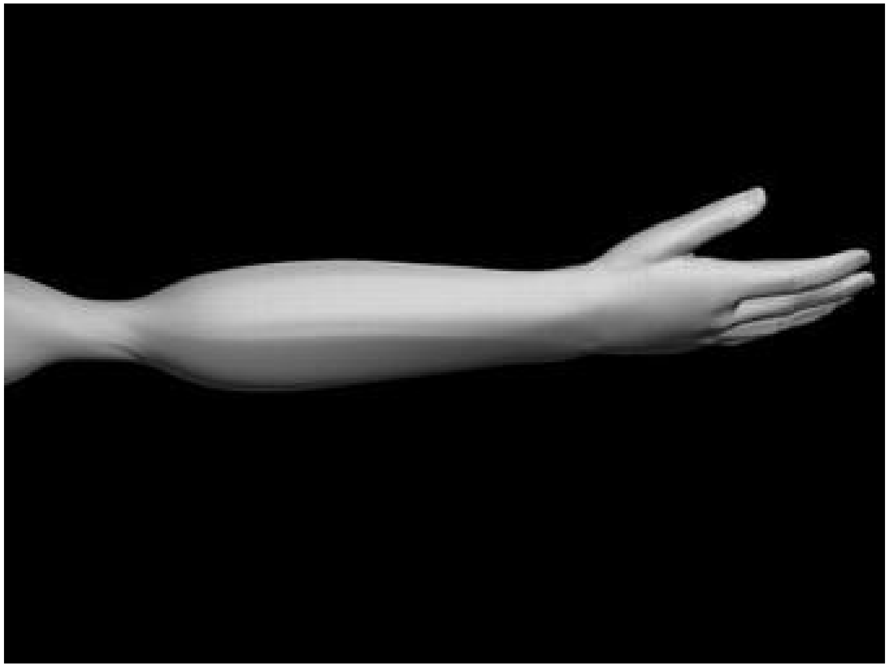
- Deformation Subspace is limited

  => Pathological Defects (complex Joint configurations, "Collapsing Elbow")

Skeleton-Subspace Deformation

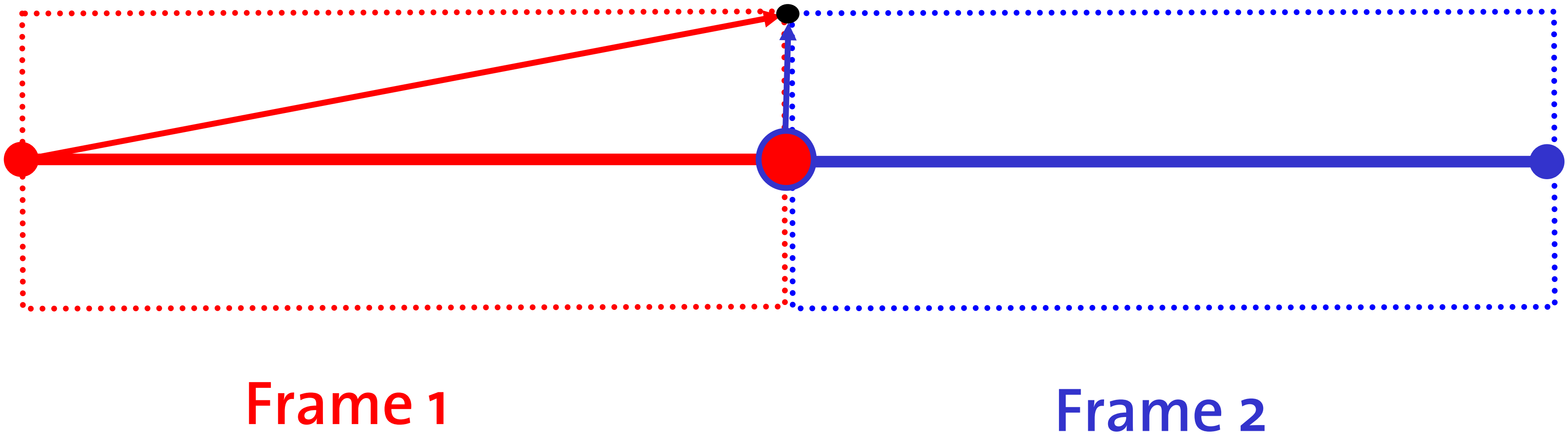
Collapsing Elbow



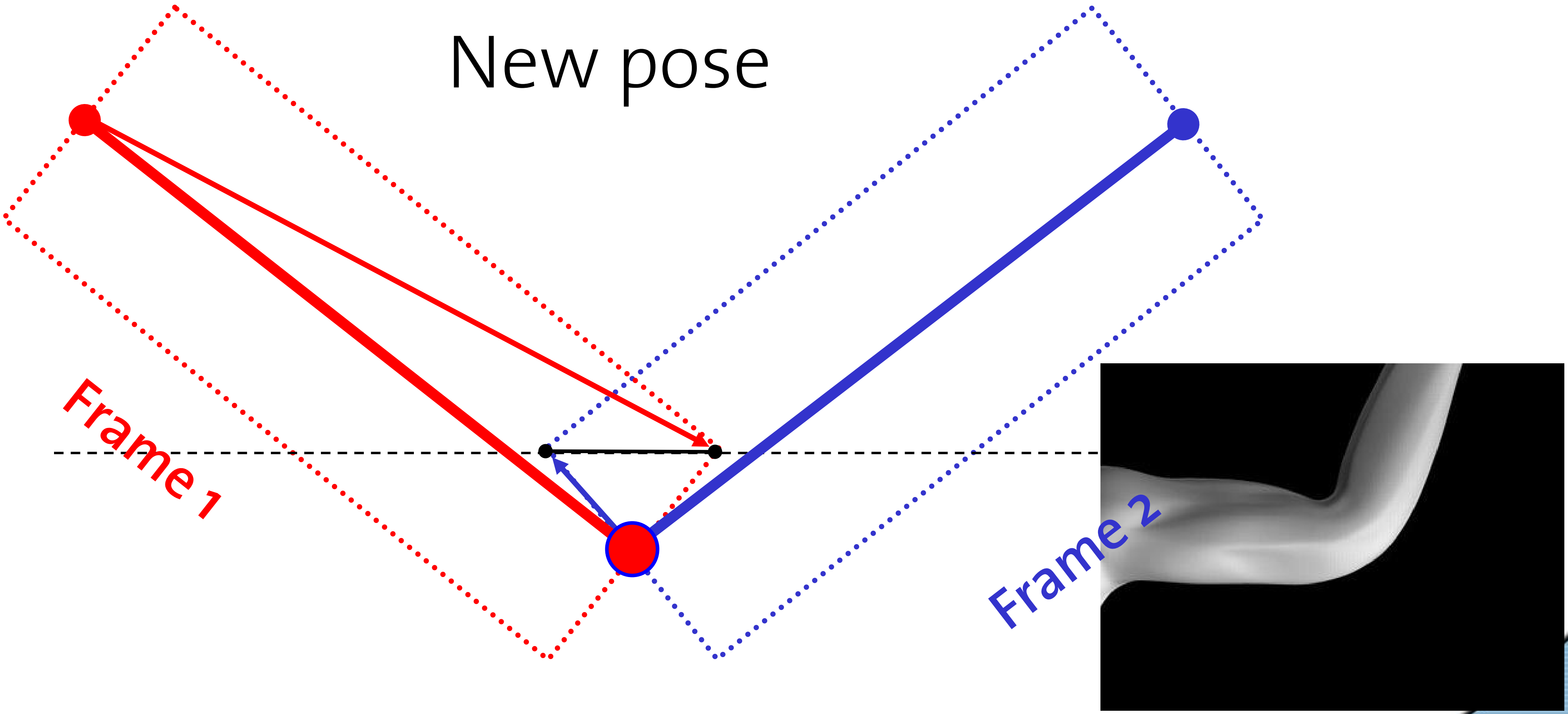


Skeleton-Subspace Deformation

Initial pose



Skeleton-Subspace Deformation



New pose

Skeleton-Subspace Deformation

Cons

- Indirect control on deformations through weights

  PSD: *direct sculpting*

- Deformation Subspace is limited

  => *Pathological Defects (complex Joints, "Collapsing Elbow")*

  PSD: *interpolation of a vast range of deformations*

Shape Interpolation

also called shape blending, multi-target morphing

Algorithm

Input: key shapes $\boldsymbol{S}_k$

(same topology)

Superposition

$$S = \sum_k w_k S_k$$

Anger

Disgust

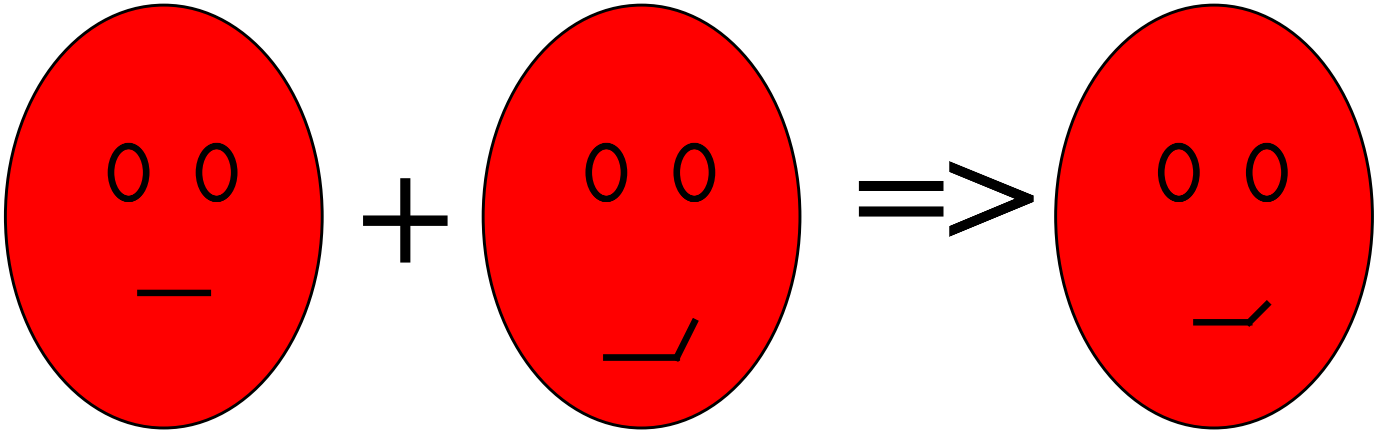
Fear

Joy

Sadness

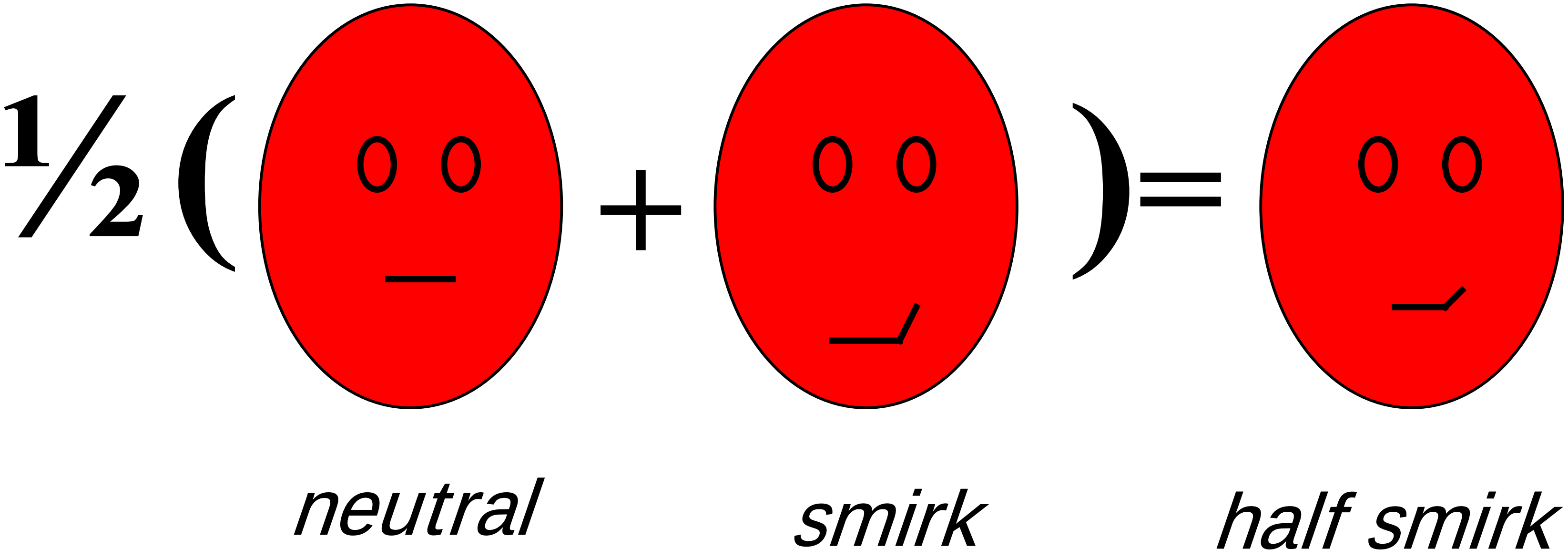
Surprise

Shape Interpolation



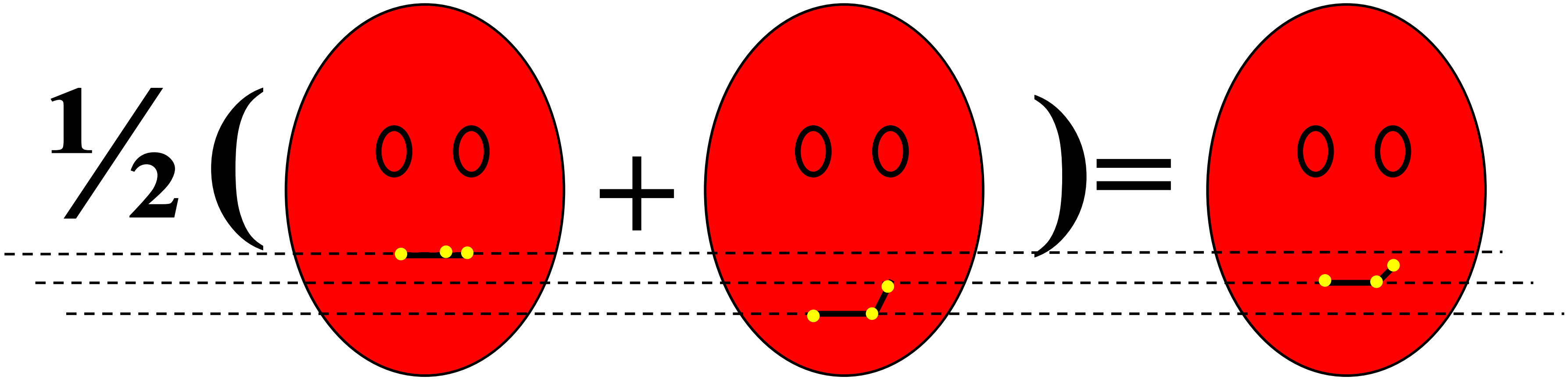
neutral *smirk* *half smirk*

Shape Interpolation



neutral smirk half smirk

Shape Interpolation



neutral *smirk* *half smirk*

Tony de Peltrie

1985

Shape Interpolation

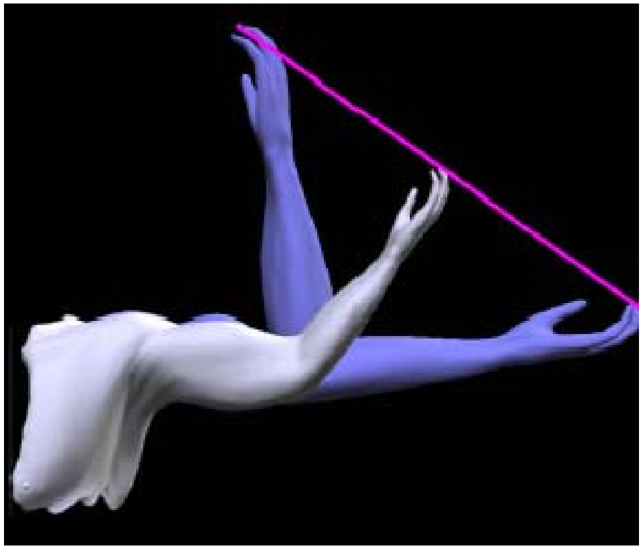
Pros

- Direct manipulation of desired expressions
- Skin deformation for facial animation
  (Used in the movies)

Shape Interpolation

Cons

- Not suited for skeleton-based deformations



Shape Interpolation

Cons

- Not suited for skeleton-based deformations
- Storage expensive

Shape Interpolation

Cons

- Not suited for skeleton-based deformations
- Storage expensive
- Conflicting Key Shapes

Conflicting Key Shapes

Key Shapes are not independent

Superposition

$a\,\mathrm{KS}_{\mathrm{Happy}} + b\,\mathrm{KS}_{\mathrm{Surprise}}$

Conflicting Key Shapes

Key Shapes are not independent

Superposition

$$a\ \mathrm{KS}_{\mathrm{Happy}} + b\ \mathrm{KS}_{\mathrm{Surprise}} + c\ \mathrm{KS}_{\mathrm{Fear}}$$

Conflicting Key Shapes

Key Shapes are not independent

Superposition

$a' \, \mathrm{KS}_{\mathrm{Happy}} + b' \, \mathrm{KS}_{\mathrm{Surprise}} + c \, \mathrm{KS}_{\mathrm{Fear}}$

Shape Interpolation

Conflicting Key Shapes

Key Shapes are not independent

Superposition

$$a'\,\mathrm{KS}_{\text{Happy}} + b'\,\mathrm{KS}_{\text{Surprise}} + c\,\mathrm{KS}_{\text{Fear}}$$

PSD: *interpolation between key Shapes*

Shape Interpolation

Cons

- Not suited for skeleton-based deformations
- Storage expensive
- Conflicting Key Shapes
- Key Shape <=> Animation Parameter

Key Shape ⬄ Anim. Parameter

If new expression is needed

$$S = \sum_k w_k S_k$$

=> New Key shape & New Parameter

Key Shape ⇔ Anim. Parameter

If new expression is needed

$$S = \sum_k w_k S_k$$

=> New Key shape & New Parameter

PSD: *no additional parameter*

freedom in the design of parameter space

Comparison Table

	Shape Interpolation	Skeleton-subspace Deformation	Pose Space Deformation
Performance	*Real-time*	*Real-time*	
Memory	*High requirements*	*Low requirements*	
Smoothness of Interpolation	*Not always*	*Yes*	
Direct manipulation	*Yes*	*No*	
Field of application	*Facial Deformation*	*Body (skeleton-influenced) Deformation*	

Comparison Table

	Shape Interpolation	**Skeleton-subspace Deformation**	**Pose Space Deformation**
Performance	*Real-time*	*Real-time*	*Real-time*
Memory	*High requirements*	*Low requirements*	
Smoothness of Interpolation	*Not always*	*Yes*	
Direct manipulation	*Yes*	*No*	
Field of application	*Facial Deformation*	*Body (skeleton-influenced) Deformation*	

Comparison Table

	Shape Interpolation	Skeleton-subspace Deformation	Pose Space Deformation
Performance	*Real-time*	*Real-time*	*Real-time*
Memory	*High requirements*	*Low requirements*	*Low requirements*
Smoothness of Interpolation	*Not always*	*Yes*	
Direct manipulation	*Yes*	*No*	
Field of application	*Facial Deformation*	*Body (skeleton-influenced) Deformation*	

Comparison Table

	Shape Interpolation	Skeleton-subspace Deformation	Pose Space Deformation
Performance	*Real-time*	*Real-time*	*Real-time*
Memory	*High requirements*	*Low requirements*	*Low requirements*
Smoothness of Interpolation	*Not always*	*Yes*	*If needed*
Direct manipulation	*Yes*	*No*	
Field of application	*Facial Deformation*	*Body (skeleton-influenced) Deformation*	

Comparison Table

	Shape Interpolation	**Skeleton-subspace Deformation**	**Pose Space Deformation**
Performance	*Real-time*	*Real-time*	*Real-time*
Memory	*High requirements*	*Low requirements*	*Low requirements*
Smoothness of Interpolation	*Not always*	*Yes*	*If needed*
Direct manipulation	*Yes*	*No*	*Yes*
Field of application	*Facial Deformation*	*Body (skeleton-influenced) Deformation*	

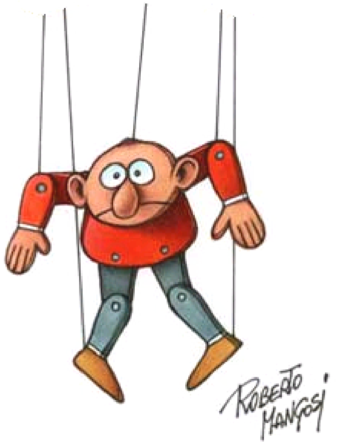
Comparison Table

	Shape Interpolation	Skeleton-subspace Deformation	Pose Space Deformation
Performance	*Real-time*	*Real-time*	*Real-time*
Memory	*High requirements*	*Low requirements*	*Low requirements*
Smoothness of Interpolation	*Not always*	*Yes*	*If needed*
Direct manipulation	*Yes*	*No*	*Yes*
Field of application	*Facial Deformation*	*Body (skeleton-influenced) Deformation*	*Facial and Body Deformations*

Deformation Model

Mapping:

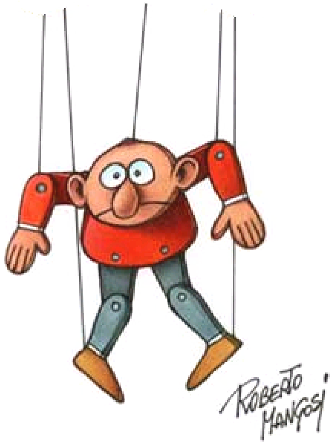
Parameters => Deformation



Deformation Model

Mapping:

Parameters => Deformation



Pose Space Deformation

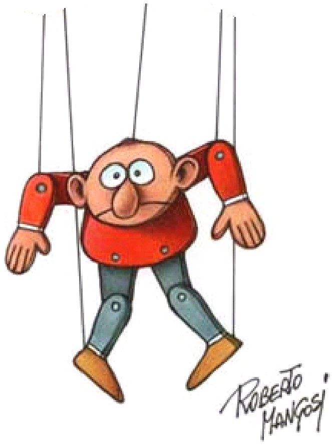
Interpolat from deformation examples

Deformation Model

Mapping:

Parameters => Deformation

Pose Space => Displacements $\Delta\mathbf{x}$



Pose Space Deformation

Interpolat from deformation examples

Pose Space Deformation

Face Animation

1. Pose Space

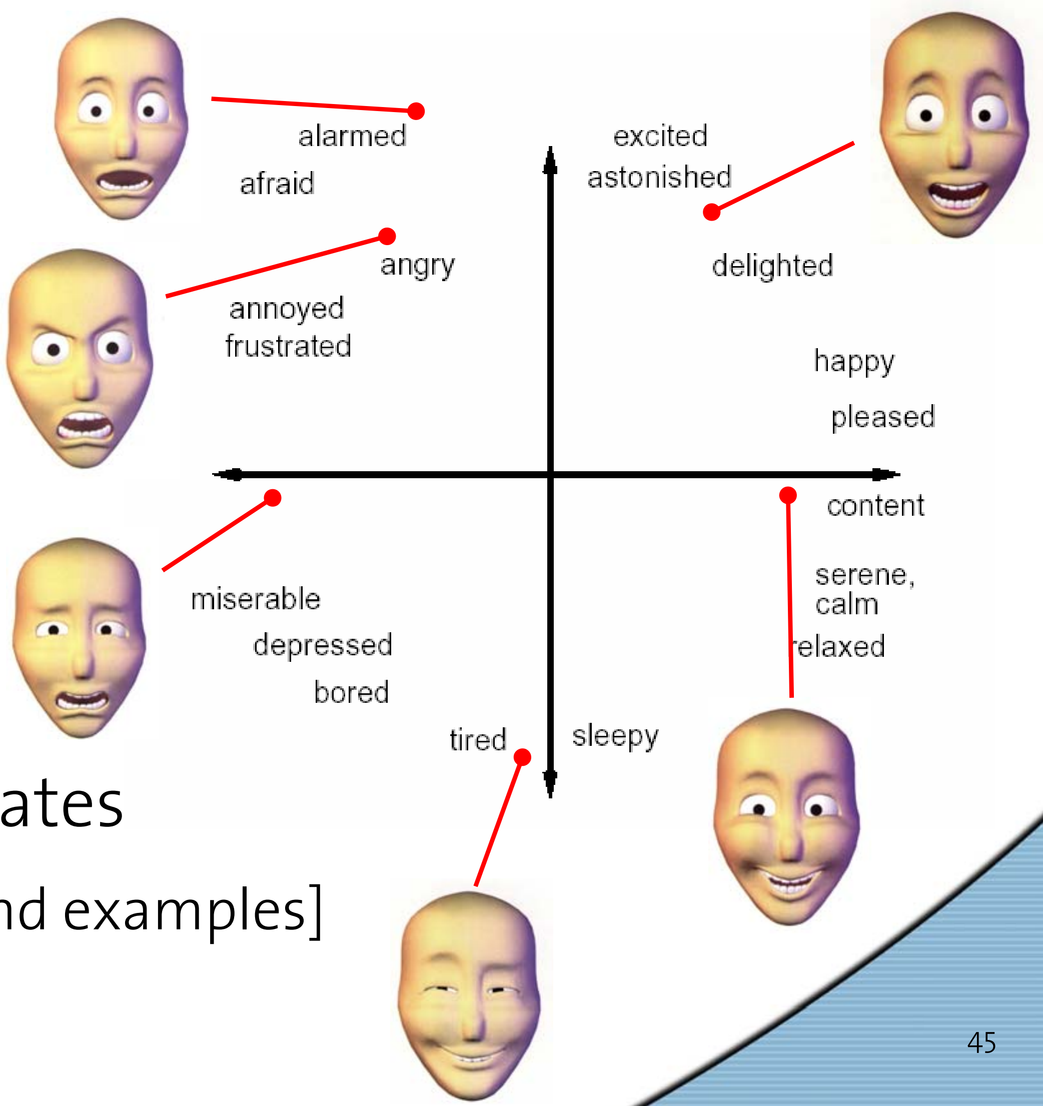
   Ex: Emotion Space

2. Initial Face Model

3. Sculpting of examples with Pose Space Coordinates

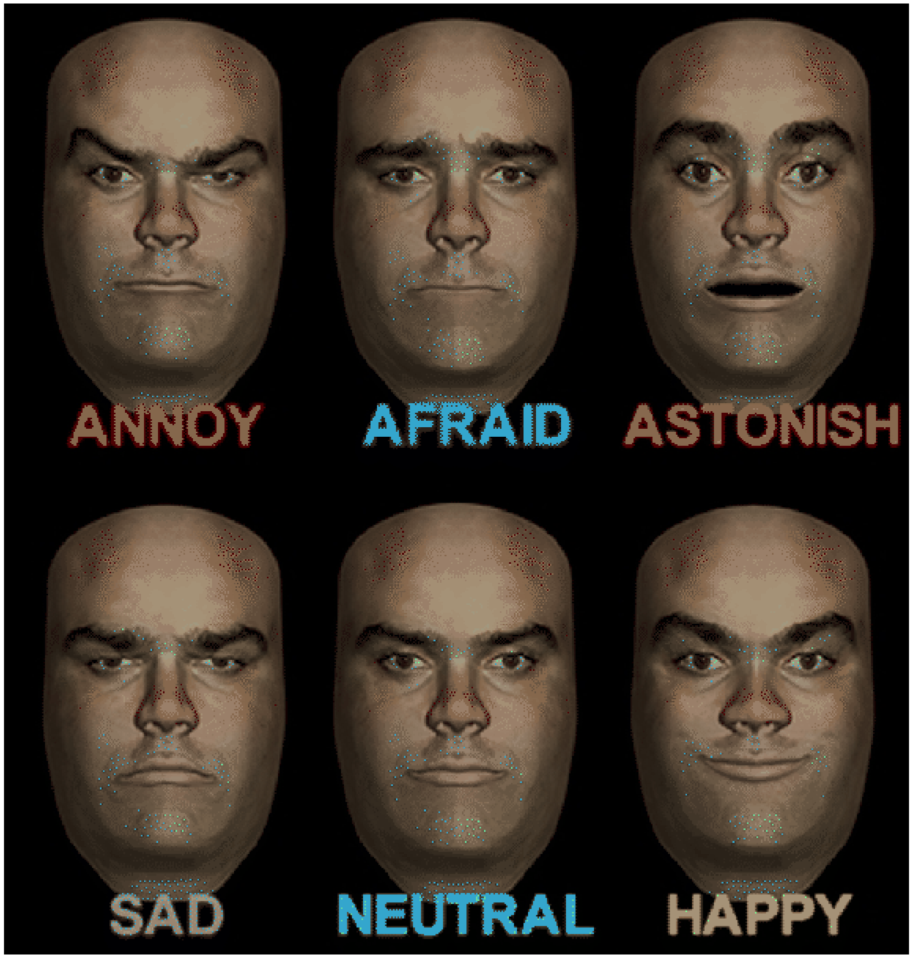
   [$\Delta\mathbf{x}$: between initial model and examples]

4. Interpolant



Pose Space Deformation

Face Animation



Pose Space Deformation

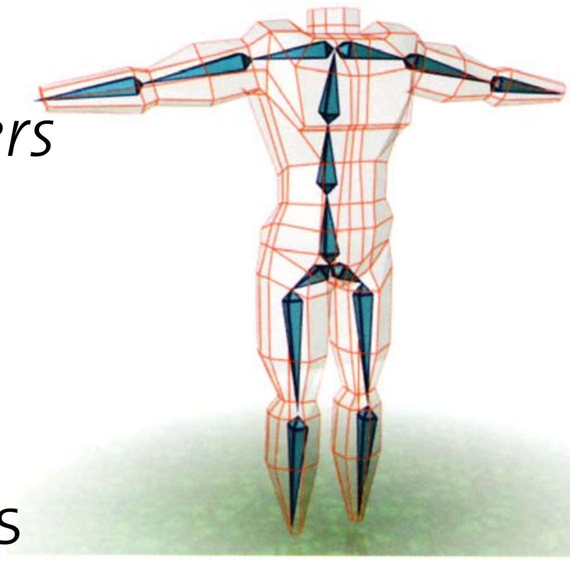
Body Animation

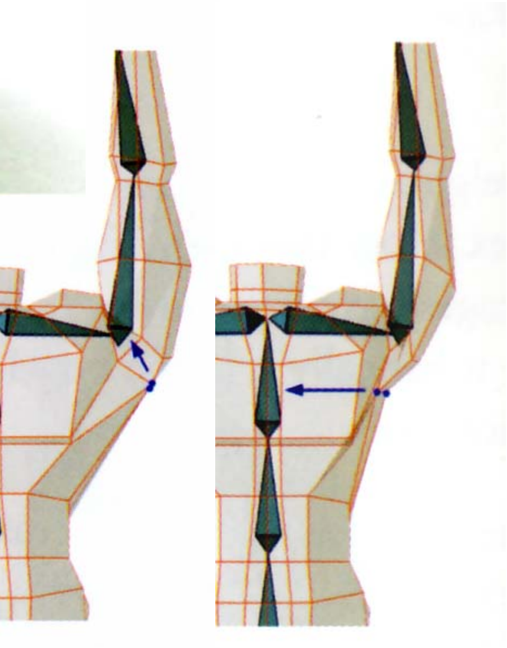
1. Pose Space
   Ex: Skeleton Parameters

2. Skeleton & Model

3. Sculpting of examples
   with Pose Space Coordinates
   [$\Delta\mathbf{x}$: between SSD and examples]

4. Interpolant





SSD PSD

Pose Space Deformation

Body Animation

Interpolat from Examples

Interpolant:

Pose Space => Displacements $\Delta\mathbf{x}$

from $\{(\text{PSCoord}_1, \Delta\mathbf{x}_1), \ldots, (\text{PSCoord}_N, \Delta\mathbf{x}_N)\}$

Scattered Data Interpolation:

Interpolation of a set of irregularly located data points

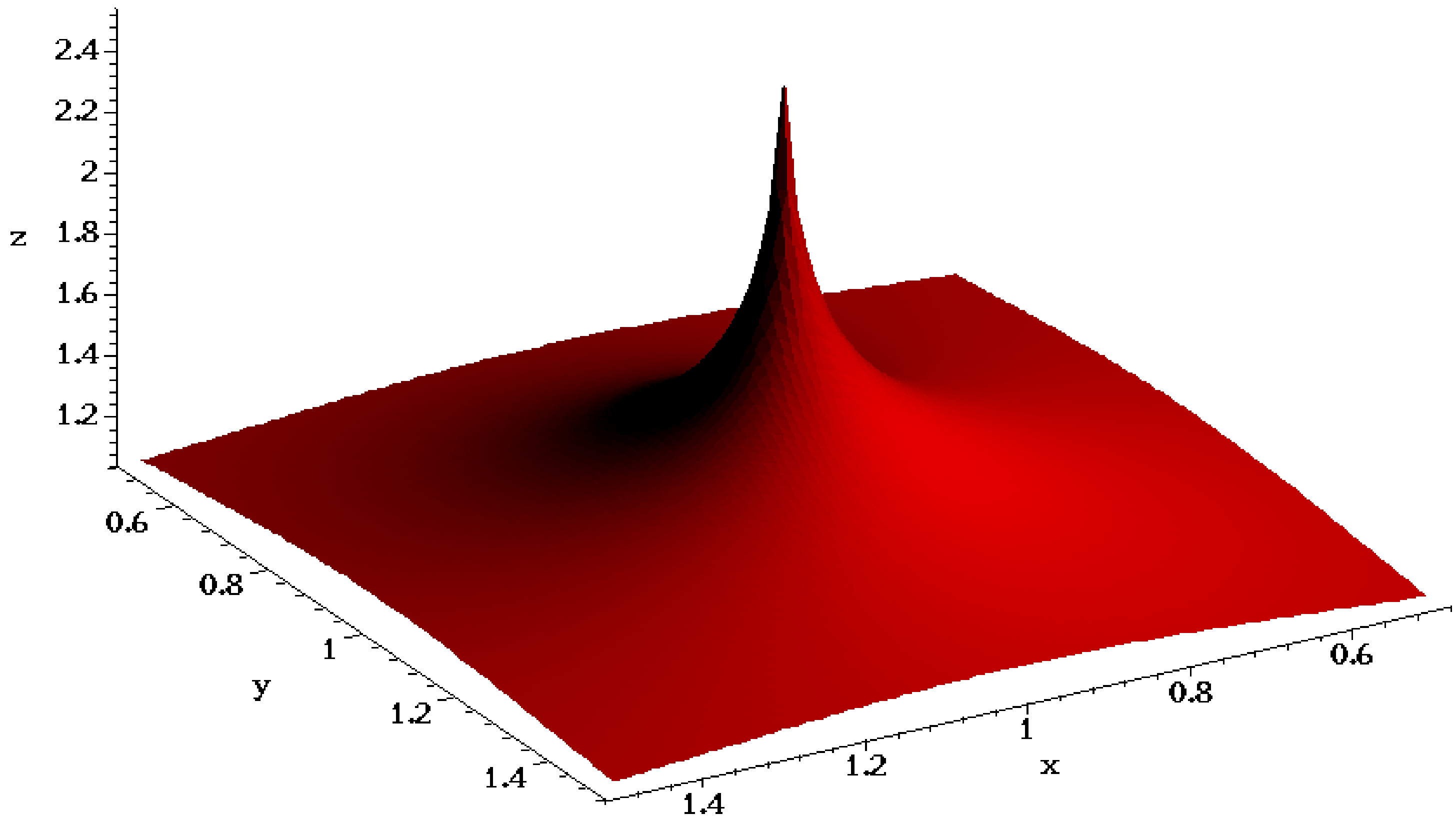
Approaches:
- Shepard’s Method
- Radial Basis Function Interpolation

Shepard's Method

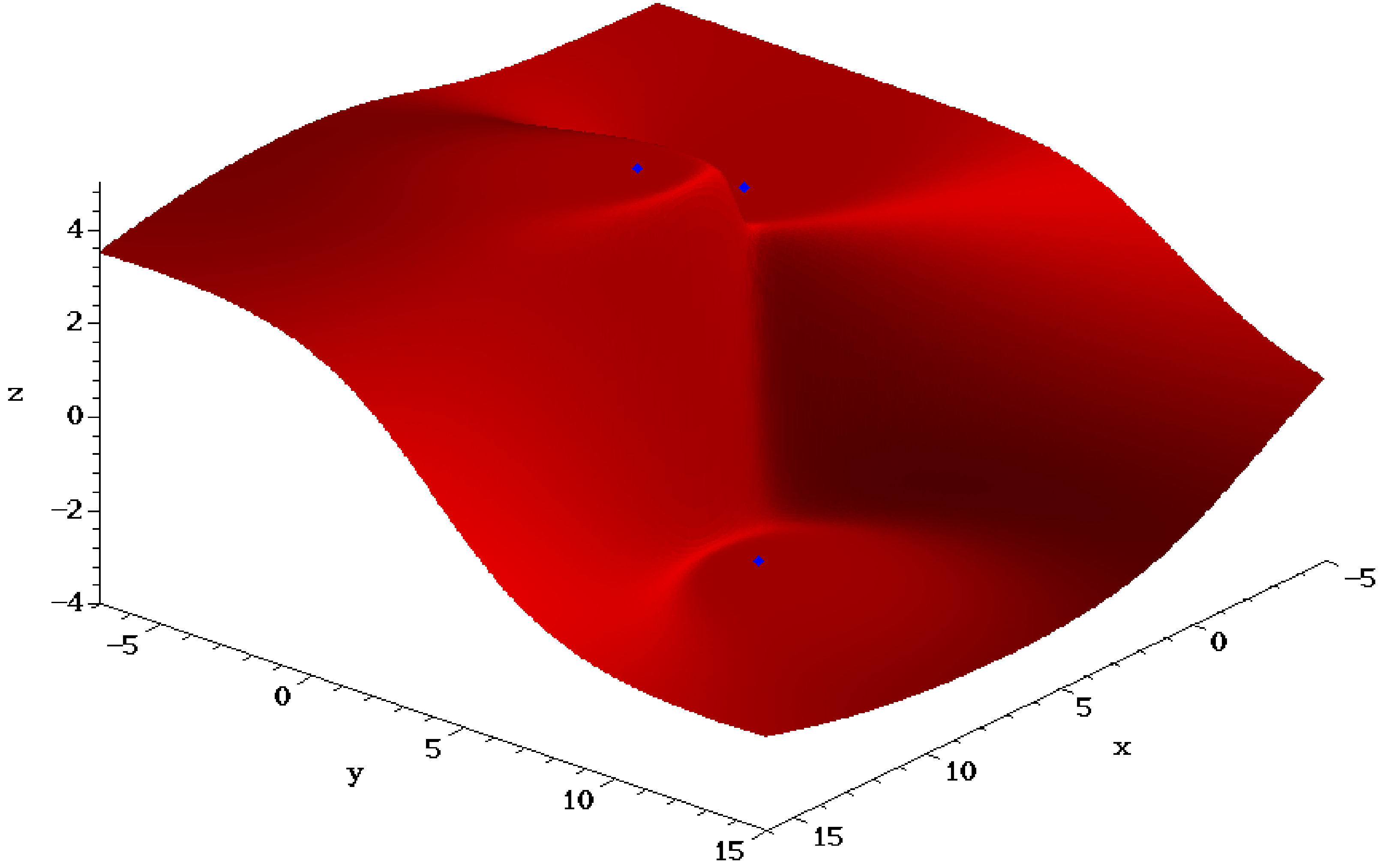
$$\hat{d}(\mathbf{x}) = \frac{\sum_{k=1}^{N} w_k(\mathbf{x}) d_k}{\sum_{k=1}^{N} w_k(\mathbf{x})} \qquad w_k(\mathbf{x}) = \left\| \mathbf{x} - \mathbf{x}_k \right\|^{-p} \ , p > 0$$

$d_1, \ldots, d_N$ are the given data points at $\mathbf{x}_1, \ldots, \mathbf{x}_N$

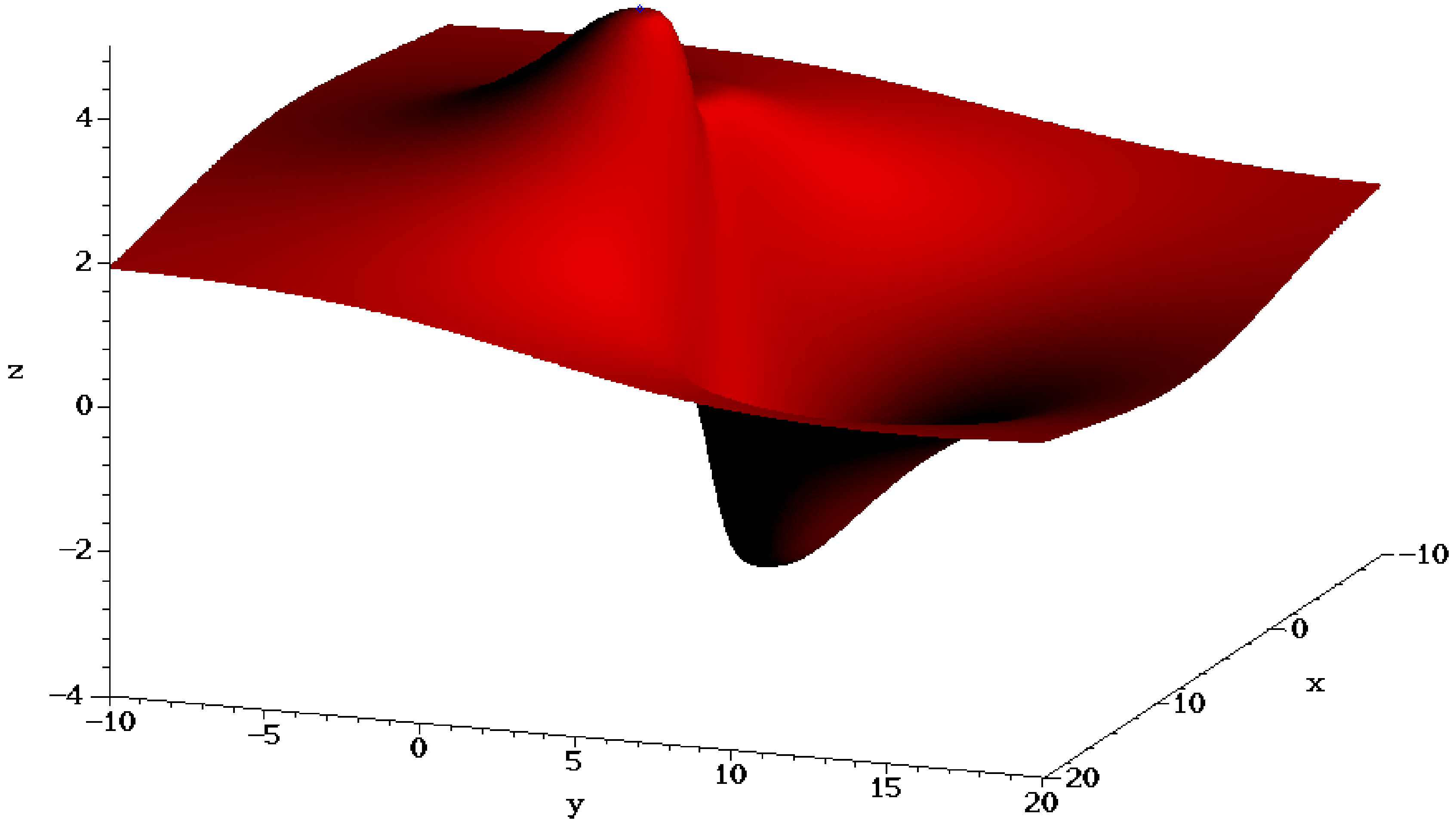
Weight function



Flat zones



Average at far points



Shepard's Method

Properties:

- Singular at data points $\mathbf{x}_k$($w_k(\mathbf{x}) = \|\mathbf{x} - \mathbf{x}_k\|^{-p}$ with $p > 0$)
- Infinitely differentiable except at data points
- Partial derivatives are zero at data points

  => flat zones around data points $\mathbf{x}_k$
- Far from data points converges to the average value

$$\hat{d}(\infty) = \frac{\sum_{k=1}^{N} w_k(\infty) d_k}{\sum_{k=1}^{N} w_k(\infty)} = \frac{\sum_{k=1}^{N} d_k}{N}$$

Radial Basis Functions

…also called Hardy's multiquadratics

$$\hat{d}(\mathbf{x}) = \sum_{k=1}^{N} w_k \, \psi\left(\left\|\mathbf{x} - \mathbf{x}_k\right\|\right)$$

$w_1,\ldots,w_N$ from $d_1,\ldots,d_N$ at $\mathbf{x}_1,\ldots,\mathbf{x}_N$ solving a system of linear equations:

$$\begin{pmatrix} d_1 \\ \vdots \\ d_N \end{pmatrix} = \begin{pmatrix} \psi\left(\left\|\mathbf{x}_1 - \mathbf{x}_k\right\|\right) & \cdots & \psi\left(\left\|\mathbf{x}_N - \mathbf{x}_1\right\|\right) \\ \vdots & \ddots & \vdots \\ \psi\left(\left\|\mathbf{x}_1 - \mathbf{x}_N\right\|\right) & \cdots & \psi\left(\left\|\mathbf{x}_N - \mathbf{x}_N\right\|\right) \end{pmatrix} \begin{pmatrix} w_1 \\ \vdots \\ w_N \end{pmatrix}$$

$$\mathbf{d} = \Psi \, \mathbf{w}$$

Choice of Radial Basis Function

$\psi(r) = r$ linear

$\psi(r) = r^3$ cubic $r = \|\mathbf{x} - \mathbf{x}_k\|$

$\psi(r) = (r^2 + \sigma^2)^{-\alpha}\ , \alpha > 0$ localized

$\psi(r) = r^2 \ln(r)$ Thin-plate spline function

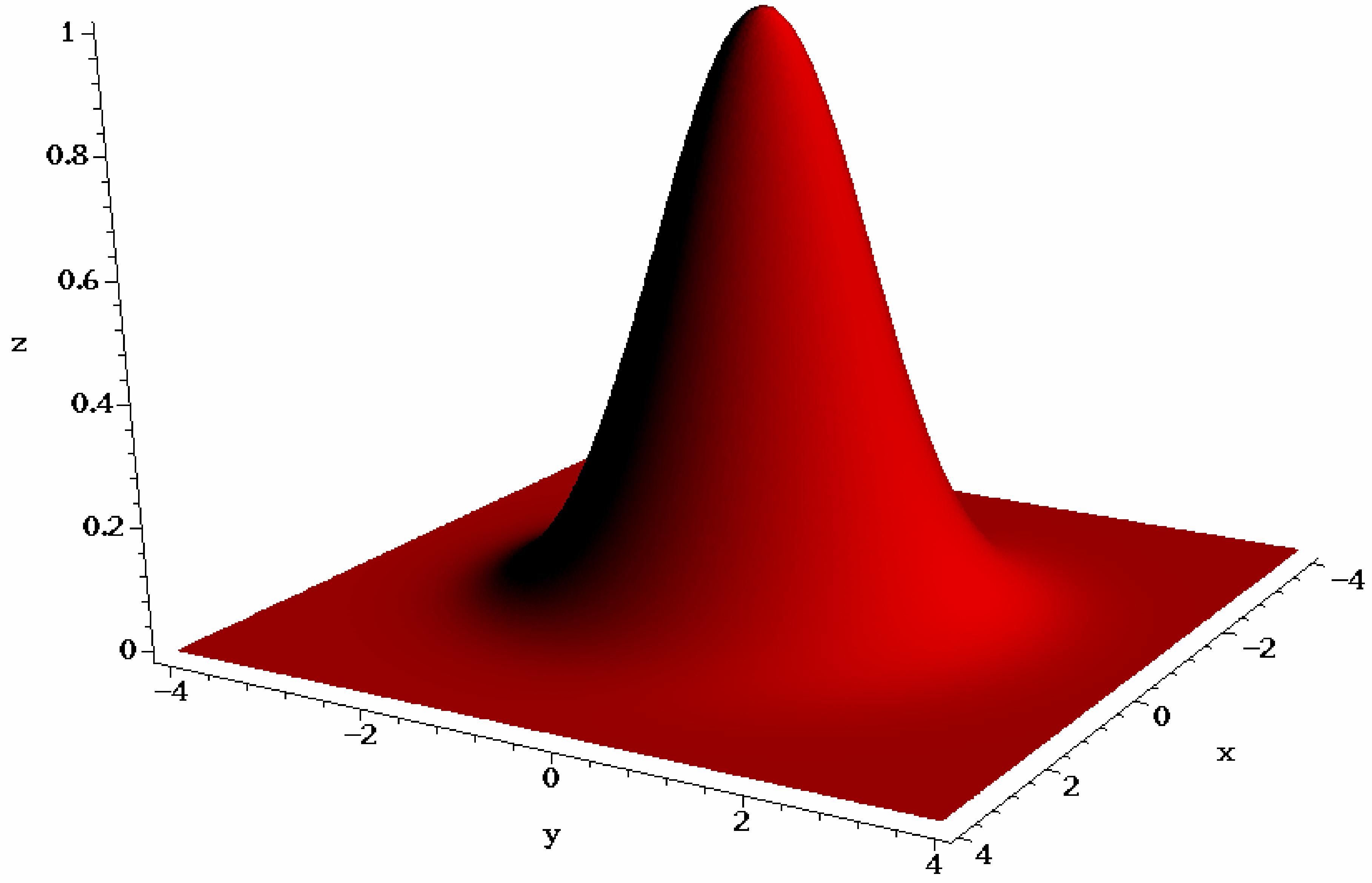
$\psi(r) = (r^2 + \sigma^2)^{\beta}\ , 0 < \beta < 1$

Gaussian Radial Basis

$$\psi(r) = \exp\left(\frac{-r^2}{2\sigma^2}\right) \qquad r = \left\|\mathbf{x} - \mathbf{x}_k\right\|$$

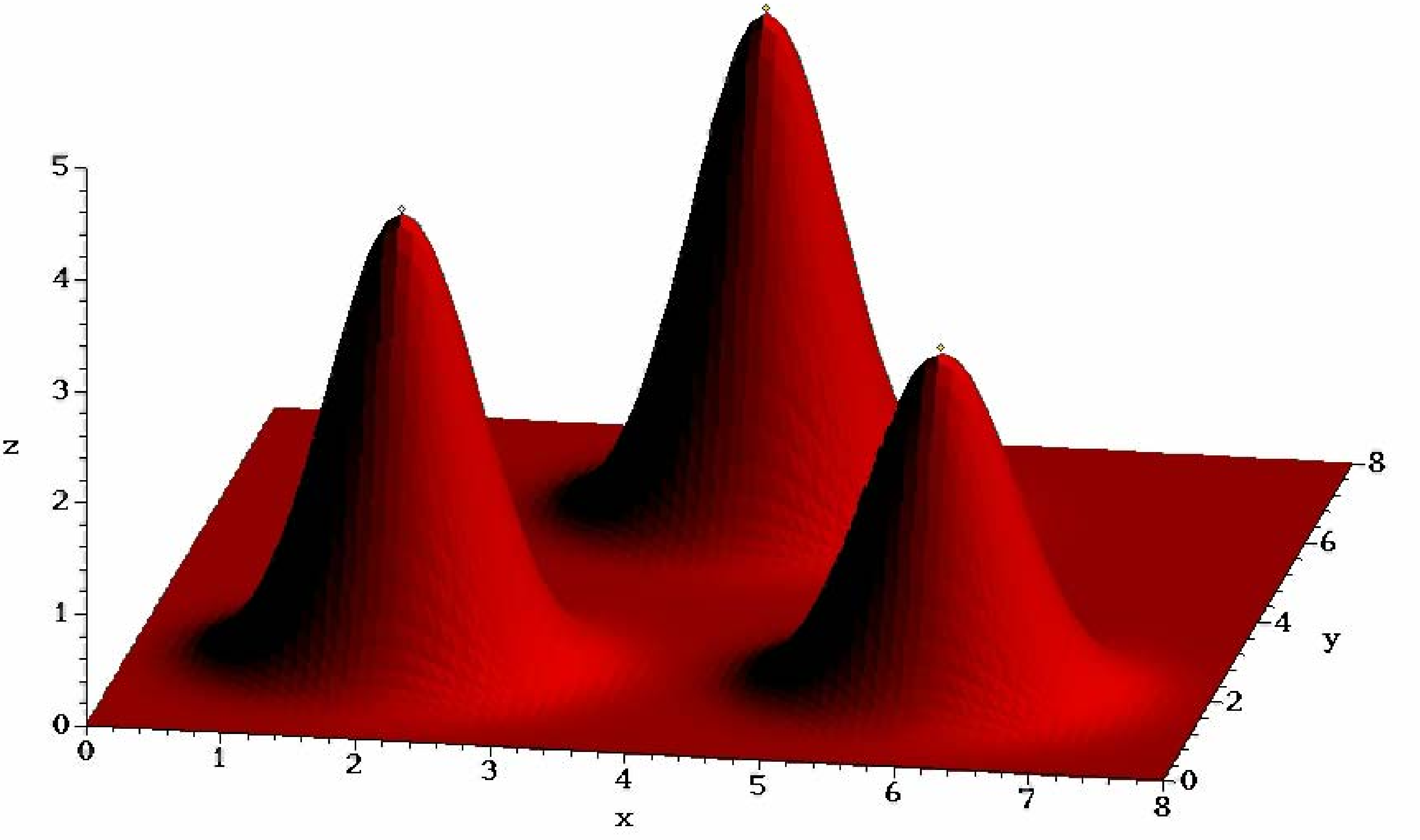
=> $$\hat{d}(\mathbf{x}) = \sum_{k=1}^{N} w_k \exp\left(\frac{-\left\|\mathbf{x} - \mathbf{x}_k\right\|^2}{2\sigma^2}\right)$$

Gaussian Basis Function



sigma = .55

Gaussian Radial Basis



Gaussian Radial Basis

$$\psi(r) = \exp\left(\frac{-r^2}{2\sigma^2}\right)$$

Properties:

- smooth
- localized: $\psi(r) = 0 \text{ for } |r| \to \infty$

  => σ width of falloff is adjustable
- relatively fast to compute (table)

Pose Space Deformation [Summary]

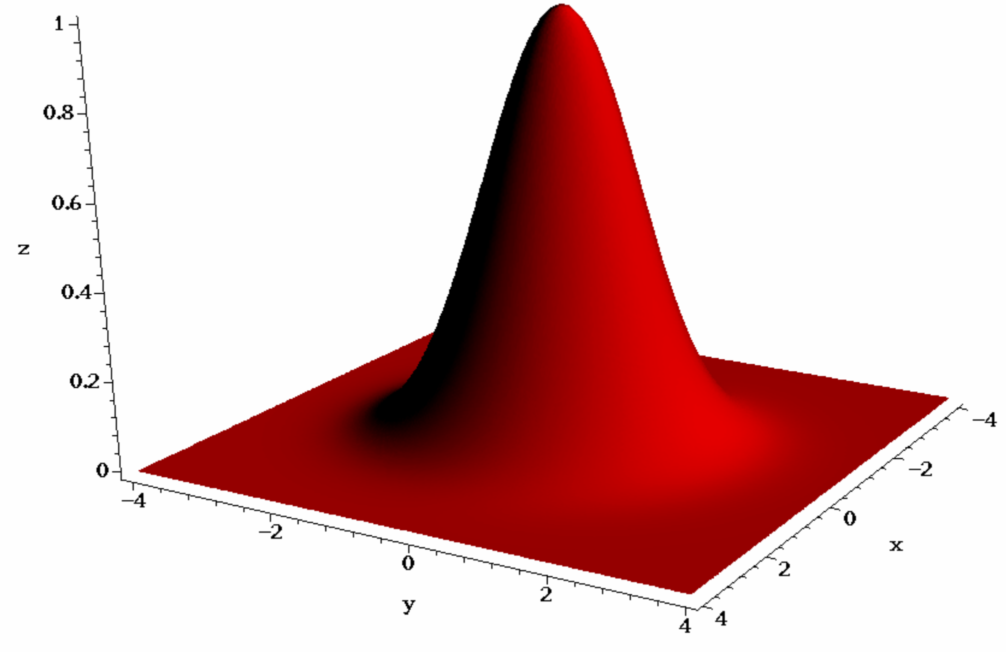
Deformation model:

Interpolant:

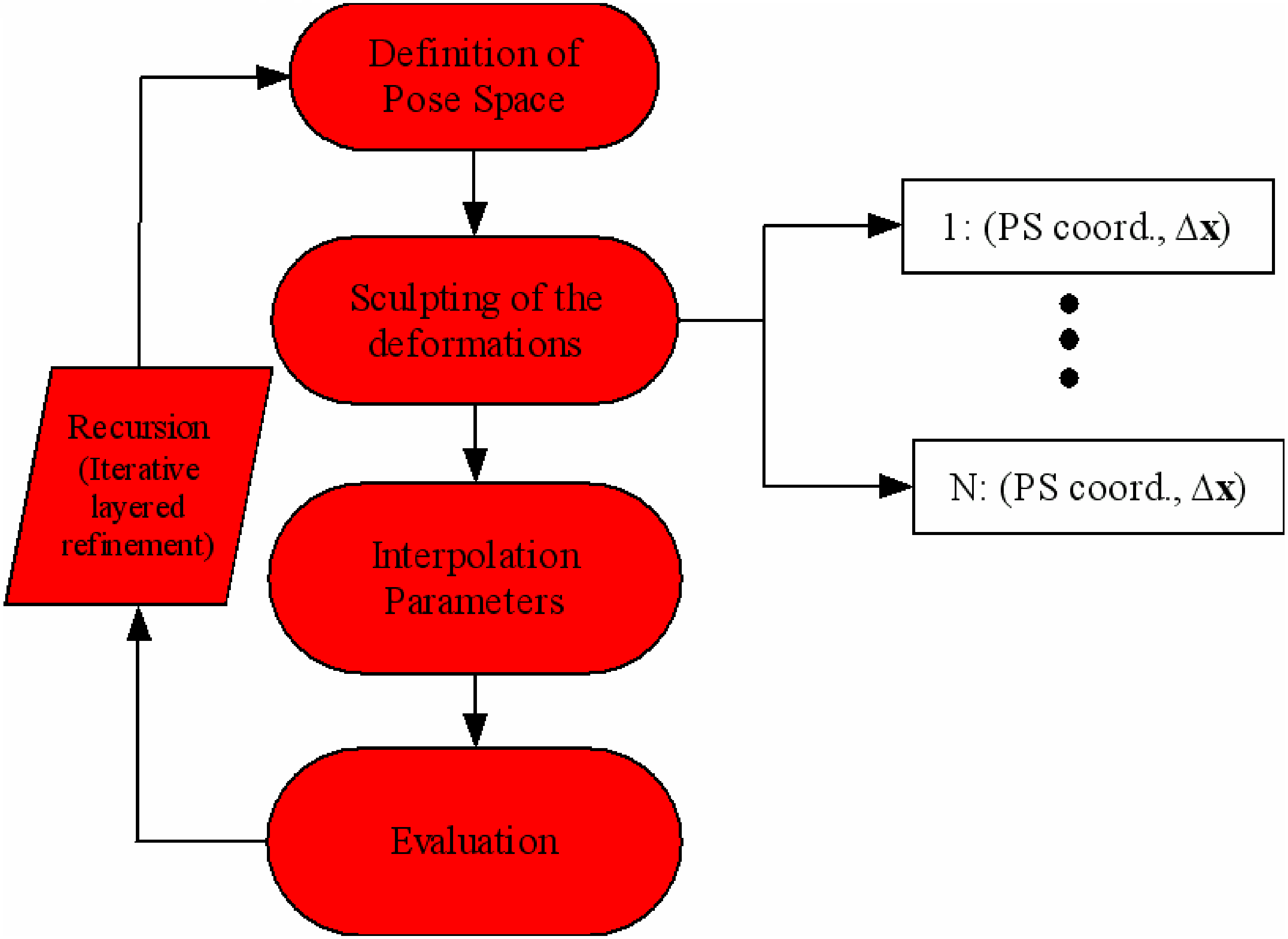
Pose Space => Displacements $\Delta\mathbf{x}$

from $\{(\text{PSCoord}_1, \Delta\mathbf{x}_1), \ldots, (\text{PSCoord}_N, \Delta\mathbf{x}_N)\}$

$$\hat{d}(\mathbf{x}) = \sum_{k=1}^{N} w_k \, \psi(\mathbf{x})$$



Workflow


Definition of Pose Space
Sculpting of the deformations
Interpolation Parameters
Evaluation
Recursion (Iterative layered refinement)
1: (PS coord., Δ**x**)
N: (PS coord., Δ**x**)

Performance

For N poses:

$$\Psi = \begin{pmatrix} \psi\left(\left\|\mathbf{x}_1 - \mathbf{x}_k\right\|\right) & \cdots & \psi\left(\left\|\mathbf{x}_N - \mathbf{x}_1\right\|\right) \\ \vdots & \ddots & \vdots \\ \psi\left(\left\|\mathbf{x}_1 - \mathbf{x}_N\right\|\right) & \cdots & \psi\left(\left\|\mathbf{x}_N - \mathbf{x}_N\right\|\right) \end{pmatrix}$$

- Preprocessing phase: $\mathbf{d} = \Psi\,\mathbf{w}$
  - NxN Matrix Ψ must be inverted
  - Matrix-vector multiplication $\mathbf{w} = \Psi^{-1}\,\mathbf{d}$
    (∀ component of ∀ displaced vertex)

- Animation phase: $\hat{d}(\mathbf{x}) = \sum_{k=1}^{N} w_k\,\psi\left(\left\|\mathbf{x} - \mathbf{x}_k\right\|\right)$
  - Interpolated table lookup

Memory Requirements

For N poses:

Every vertex stores

Nx3 weights

(N weights ∀ component of a vertex)

N pose space coordinates of ex.

$$\hat{d}(\mathbf{x}) = \sum_{k=1}^{N} w_k \, \psi\left(\left\|\mathbf{x} - \mathbf{x}_k\right\|\right)$$

Memory Requirements

For N poses:

Every vertex stores

Nx3 weights

(N weights ∀ component of a vertex)

N pose space coordinates of ex.

$$\hat{d}(\mathbf{x}) = \sum_{k=1}^{N} w_k \, \psi\left(\left\|\mathbf{x} - \mathbf{x}_k\right\|\right)$$

Memory Requirements

For N poses:

Every vertex stores

Nx3 weights

(N weights ∀ component of a vertex)

N pose space coordinates of ex.

$$\hat{d}(\mathbf{x}) = \sum_{k=1}^{N} w_k \, \psi\left(\left\|\mathbf{x} - \mathbf{x}_k\right\|\right)$$

Pros & Cons

Pros:

- Wide range of deformations
- Arbitrary Pose Space Axes
- Real-time synthesis
- Relatively simple to implement
- Control on interpolation (σ)
- Direct manipulation
  (Iterative layered refinement)

Pros & Cons

Cons:

- Accuracy is reliant on the modeler/animator
- σ is manually tuned
- Performance
- Memory requirements

Critiques

- Least Square Problem $\mathbf{w} = \left(\mathbf{\Psi}^T \mathbf{\Psi}\right)^{-1} \mathbf{\Psi}^T \mathbf{d}$?

  $\mathbf{w} = \mathbf{\Psi}^{-1}\,\mathbf{d}$ with $\mathbf{\Psi}$ regular
- If N poses then 3 NxN matrices must be inverted for each control vertex?

  Only one NxN matrix must be inverted $\mathbf{\Psi}$
- Close coordinates in PS results in a numerically unstable matrix $\mathbf{\Psi}$ (regularization, TSVD)

Related Papers

Shape from Examples

Sloan, Rose, Cohen (I3D conference 2001)

EigenSkin

Kry, James, Pai (SIGGRAPH 2002)

Skinning Mesh Animation

James, Twigg (SIGGRAPH 2005)

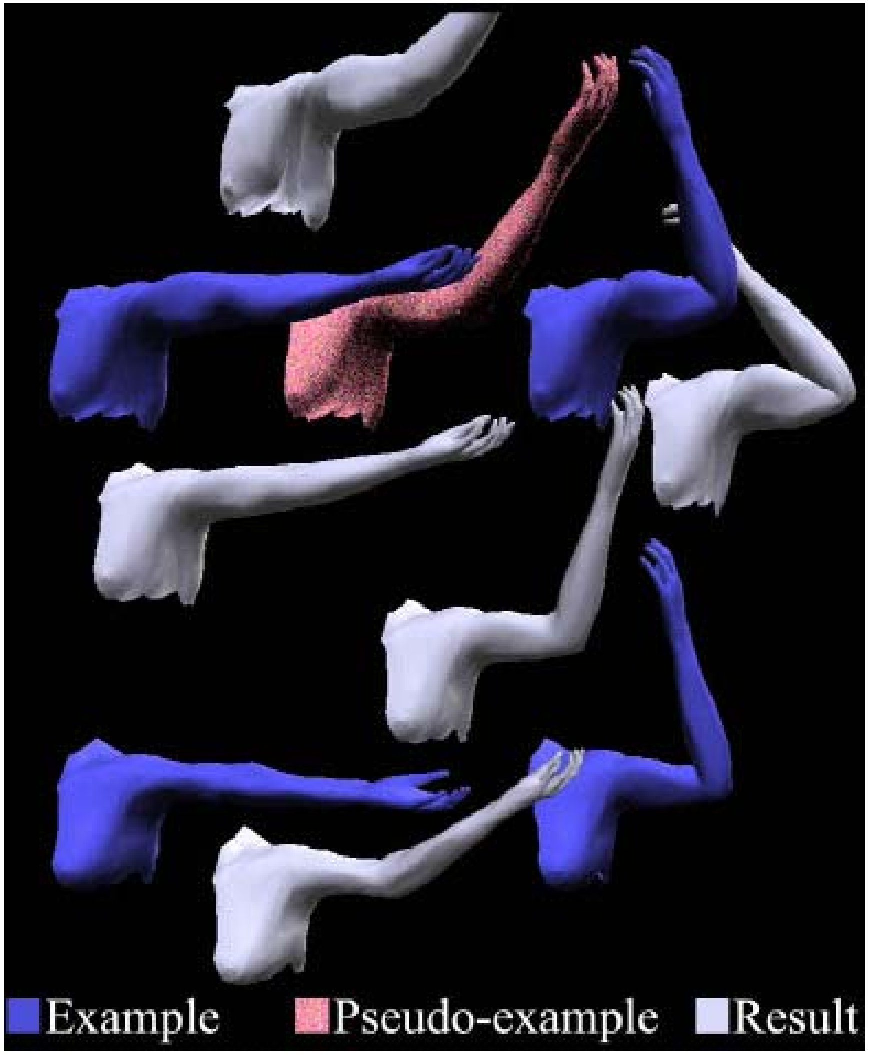
Shape by Examples

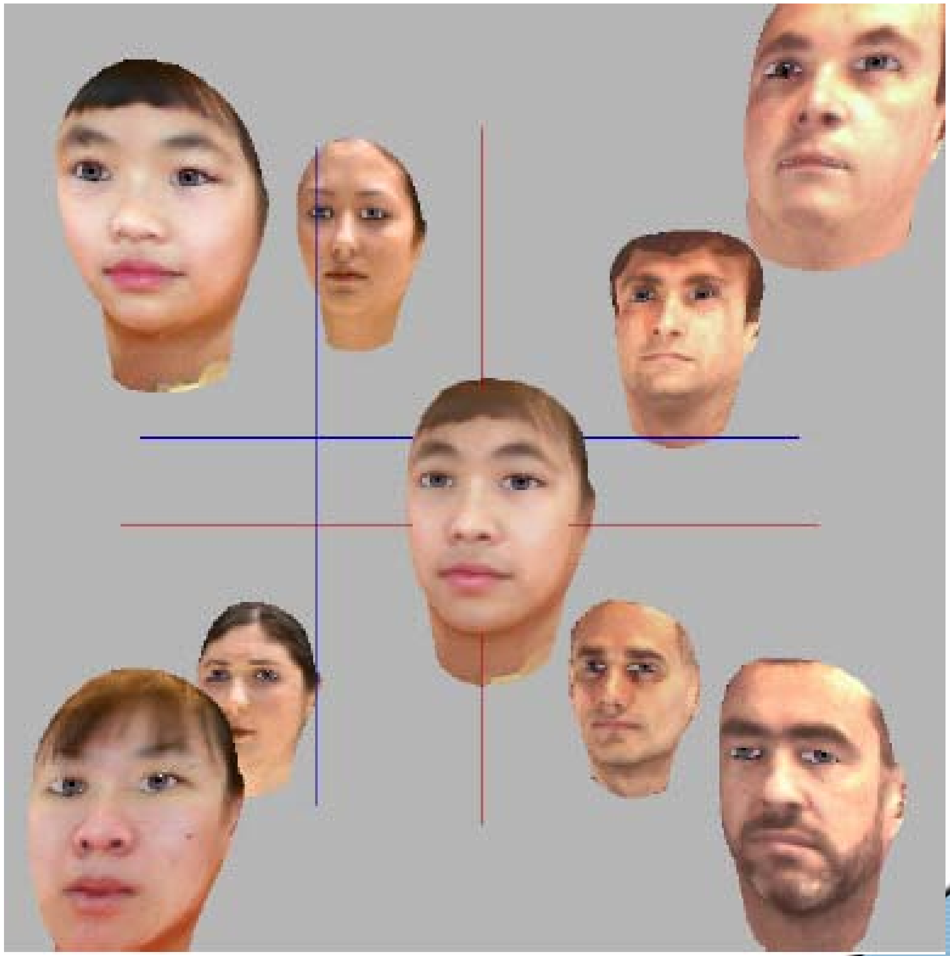
Sloan, Rose, Cohen

- Paradigm: Design by Examples
- Interpolation in Lagrangian form
- Radial Basis Functions (B-Splines) and Polynomials
- Interpolation & Extrapolation
- Reparameterization
- Applied also to Textures
- Preprocessing phase:
  - one linear system per example
- Animation phase: twice faster

Shape by Examples

Sloan, Rose, Cohen





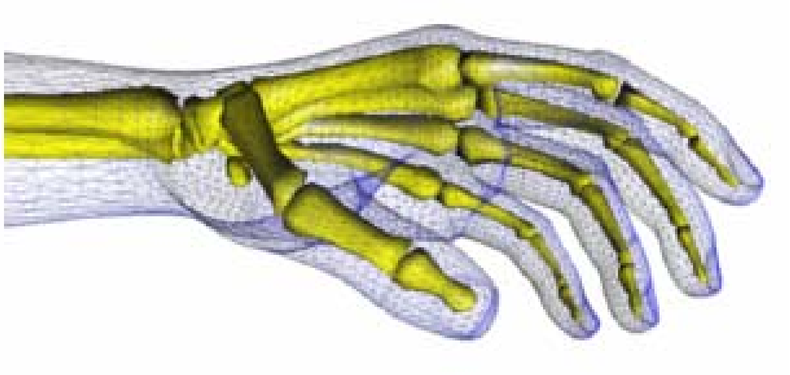
EigenSkin

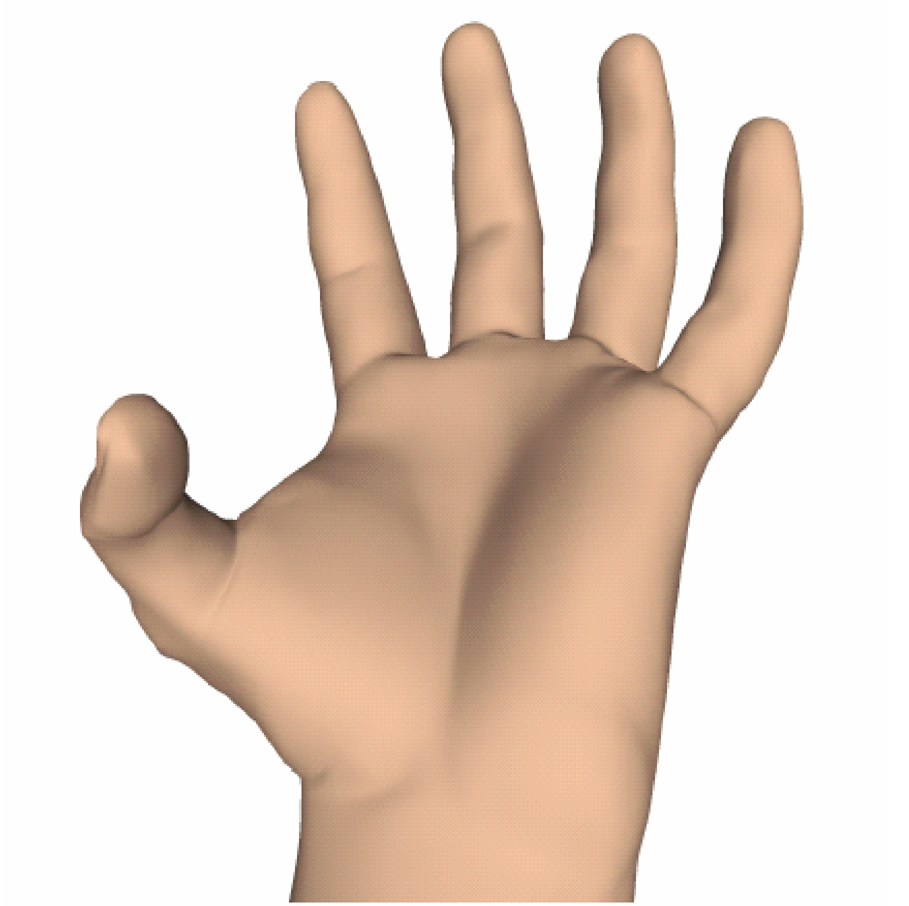
Kry, James, Pai

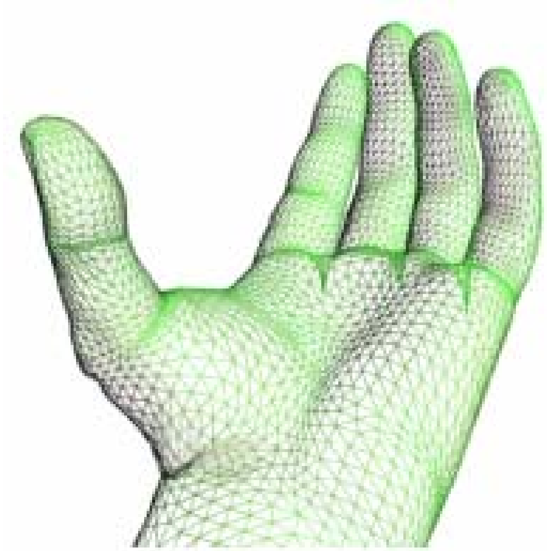
- Paradigm: Design by Examples
- Articulated characters
- Deformation field for each Joint support
  - EigenDeformations (Deformation basis)
- Mapping: Joint => Eigendeformation coordinates
  - 1-D interpolation with Radial Basis Functions
  - No non-linear joint-joint coupling effects
- Graphical Hardware Optimization
  - Reduction of Memory Requirements
  - Real-time rendering

EigenSkin

Kry, James, Pai







Skinning Mesh Animation

James, Twigg

- Approximation of sequence of input meshes
  - No Pose Space
- Skinning algorithm:
  - Proxy bone Transformations (+ weights) automatically approximated
  - Displacement field in rest pose (TSVD)
- Real-time rendering (Graphical HW support)

… more to come

Question Time