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”
Problem Statement

Main Components:

1. Character Model
2. Deformation model
Character Animation

Deformation Model

Complexity:
∀ vertex: 3 DoFs

=> Reduction of DoFs
Character Animation

Deformation Model

Mapping:

Parameters $\Rightarrow$ Deformation
Character Animation

Deformation Model

Joint rotations of a skeleton
Character Animation

Deformation Model

Emotional Axes

X=Sad/Happy
Y=Relaxed/Excited

alarmed
afraid
angry
annoyed
frustrated

excited
astonished
delighted
happy
pleased
content

serene,
calm
relaxed

miserable
depressed
bored
tired

sleepy
Character Animation

**Deformation Model**

Reduction of DoFs

→ Reduced Deformation Freedom

- Pathological Defects
- Movement Deficiencies
- Lack of realism
- Difficulty to control the deformations

Simone Croci
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

\[ \mathbf{v} = \mathbf{w}' \mathbf{T}'(\mathbf{v}') + \mathbf{w}'' \mathbf{T}''(\mathbf{v}'') \]

\( \mathbf{T} \) from local to world coordinates
Prospectives

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

Frame 1

Frame 2
Skeleton-Subspace Deformation

New pose

Frame 1

Frame 2
Cons

- Indirect control on deformations through weights
  PSD: *direct sculpting*

- Deformation Subspace is limited
  \[\Rightarrow\text{Pathological Defects (complex Joints, “Collapsing Elbow”)}\]
  PSD: *interpolation of a vast range of deformations*
Shape Interpolation

also called shape blending, multi-target morphing

Algorithm
Input: key shapes $S_k$
(same topology)

Superposition

$$S = \sum_k w_k S_k$$
Shape Interpolation

neutral + smirk => half smirk
Simone Croci

Shape Interpolation

\[
\frac{1}{2} \left( \begin{array}{c}
\text{neutral} \\
\text{smirk} \\
\text{half smirk}
\end{array} \right)
\]
Shape Interpolation

\[
\frac{1}{2} \left( \text{neutral} + \text{smirk} \right) = \text{half smirk}
\]
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
Shape Interpolation

Conflicting Key Shapes

Key Shapes are not independent

Superposition

\[ a \, KS_{\text{Happy}} + b \, KS_{\text{Surprise}} \]
**Shape Interpolation**

**Conflicting Key Shapes**

Key Shapes are not independent

Superposition

\[ a \text{ KS}_{\text{Happy}} + b \text{ KS}_{\text{Surprise}} + c \text{ KS}_{\text{Fear}} \]
Shape Interpolation

Conflicting Key Shapes

Key Shapes are not independent

Superposition

\[ a' \text{ KS}_{\text{Happy}} + b' \text{ KS}_{\text{Surprise}} + c \text{ KS}_{\text{Fear}} \]
Shape Interpolation

Conflicting Key Shapes

Key Shapes are not independent

Superposition

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

PSD: *interpolation between key Shapes*
Shape Interpolation

Cons

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

Key Shape ⇔ Anim. Parameter

If new expression is needed

$S = \sum_{k} w_k S_k$

$\Rightarrow$ New Key shape & New Parameter
Shape Interpolation

Key Shape ⇔ Anim. Parameter

If new expression is needed

$\Rightarrow$ New Key shape & New Parameter

$S = \sum_k w_k S_k$

PSD: no additional parameter

freedom in the design of parameter space
# Comparison Table

<table>
<thead>
<tr>
<th></th>
<th>Shape Interpolation</th>
<th>Skeleton-subspace Deformation</th>
<th>Pose Space Deformation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Performance</strong></td>
<td><em>Real-time</em></td>
<td><em>Real-time</em></td>
<td></td>
</tr>
<tr>
<td><strong>Memory</strong></td>
<td><em>High requirements</em></td>
<td><em>Low requirements</em></td>
<td></td>
</tr>
<tr>
<td><strong>Smoothness of Interpolation</strong></td>
<td><em>Not always</em></td>
<td><em>Yes</em></td>
<td></td>
</tr>
<tr>
<td><strong>Direct manipulation</strong></td>
<td><em>Yes</em></td>
<td><em>No</em></td>
<td></td>
</tr>
<tr>
<td><strong>Field of application</strong></td>
<td><em>Facial Deformation</em></td>
<td><em>Body (skeleton-influenced) Deformation</em></td>
<td></td>
</tr>
</tbody>
</table>
# Comparison Table

<table>
<thead>
<tr>
<th></th>
<th>Shape Interpolation</th>
<th>Skeleton-subspace Deformation</th>
<th>Pose Space Deformation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Performance</strong></td>
<td>Real-time</td>
<td>Real-time</td>
<td>Real-time</td>
</tr>
<tr>
<td><strong>Memory</strong></td>
<td>High requirements</td>
<td>Low requirements</td>
<td></td>
</tr>
<tr>
<td><strong>Smoothness of Interpolation</strong></td>
<td>Not always</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><strong>Direct manipulation</strong></td>
<td>Yes</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td><strong>Field of application</strong></td>
<td>Facial Deformation</td>
<td>Body (skeleton-influenced) Deformation</td>
<td></td>
</tr>
</tbody>
</table>
## Comparison Table

<table>
<thead>
<tr>
<th></th>
<th>Shape Interpolation</th>
<th>Skeleton-subspace Deformation</th>
<th>Pose Space Deformation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Performance</strong></td>
<td><em>Real-time</em></td>
<td><em>Real-time</em></td>
<td><em>Real-time</em></td>
</tr>
<tr>
<td><strong>Memory</strong></td>
<td><em>High requirements</em></td>
<td><em>Low requirements</em></td>
<td><em>Low requirements</em></td>
</tr>
<tr>
<td><strong>Smoothness of Interpolation</strong></td>
<td><em>Not always</em></td>
<td><em>Yes</em></td>
<td></td>
</tr>
<tr>
<td><strong>Direct manipulation</strong></td>
<td><em>Yes</em></td>
<td><em>No</em></td>
<td></td>
</tr>
<tr>
<td><strong>Field of application</strong></td>
<td><em>Facial Deformation</em></td>
<td><em>Body (skeleton-influenced) Deformation</em></td>
<td></td>
</tr>
</tbody>
</table>
## Comparison Table

<table>
<thead>
<tr>
<th></th>
<th>Shape Interpolation</th>
<th>Skeleton-subspace Deformation</th>
<th>Pose Space Deformation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Performance</strong></td>
<td>Real-time</td>
<td>Real-time</td>
<td>Real-time</td>
</tr>
<tr>
<td><strong>Memory</strong></td>
<td>High requirements</td>
<td>Low requirements</td>
<td>Low requirements</td>
</tr>
<tr>
<td><strong>Smoothness of Interpolation</strong></td>
<td>Not always</td>
<td>Yes</td>
<td>If needed</td>
</tr>
<tr>
<td><strong>Direct manipulation</strong></td>
<td>Yes</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td><strong>Field of application</strong></td>
<td>Facial Deformation</td>
<td>Body (skeleton-influenced) Deformation</td>
<td></td>
</tr>
</tbody>
</table>
## Comparison Table

<table>
<thead>
<tr>
<th></th>
<th>Shape Interpolation</th>
<th>Skeleton-subspace Deformation</th>
<th>Pose Space Deformation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Performance</strong></td>
<td><em>Real-time</em></td>
<td><em>Real-time</em></td>
<td><em>Real-time</em></td>
</tr>
<tr>
<td><strong>Memory</strong></td>
<td><em>High requirements</em></td>
<td><em>Low requirements</em></td>
<td><em>Low requirements</em></td>
</tr>
<tr>
<td><strong>Smoothness of Interpolation</strong></td>
<td><em>Not always</em></td>
<td><em>Yes</em></td>
<td><em>If needed</em></td>
</tr>
<tr>
<td><strong>Direct manipulation</strong></td>
<td><em>Yes</em></td>
<td><em>No</em></td>
<td><em>Yes</em></td>
</tr>
<tr>
<td><strong>Field of application</strong></td>
<td><em>Facial Deformation</em></td>
<td><em>Body (skeleton-influenced) Deformation</em></td>
<td></td>
</tr>
</tbody>
</table>
## Comparison Table

<table>
<thead>
<tr>
<th></th>
<th>Shape Interpolation</th>
<th>Skeleton-subspace Deformation</th>
<th>Pose Space Deformation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Performance</strong></td>
<td><em>Real-time</em></td>
<td><em>Real-time</em></td>
<td><em>Real-time</em></td>
</tr>
<tr>
<td><strong>Memory</strong></td>
<td><em>High requirements</em></td>
<td><em>Low requirements</em></td>
<td><em>Low requirements</em></td>
</tr>
<tr>
<td><strong>Smoothness of Interpolation</strong></td>
<td><em>Not always</em></td>
<td><em>Yes</em></td>
<td><em>If needed</em></td>
</tr>
<tr>
<td><strong>Direct manipulation</strong></td>
<td><em>Yes</em></td>
<td><em>No</em></td>
<td><em>Yes</em></td>
</tr>
<tr>
<td><strong>Field of application</strong></td>
<td><em>Facial Deformation</em></td>
<td><em>Body (skeleton-influenced) Deformation</em></td>
<td><em>Facial and Body Deformations</em></td>
</tr>
</tbody>
</table>
Character Animation

Deformation Model

Mapping:

Parameters $\Rightarrow$ Deformation
Character Animation

Deformation Model

Mapping:

Parameters $\Rightarrow$ Deformation

Pose Space
Deformation

Interpolat from deformation examples
Character Animation

Deformation Model

Mapping:

Parameters $\Rightarrow$ Deformation

Pose Space $\Rightarrow$ Displacements $\Delta x$

Pose Space

Deformation

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

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
   \[ \Delta x: \text{between SSD and examples} \]

4. Interpolant

---

SSD

PSD
Pose Space Deformation

Body Animation
Interpolat from Examples

Interpolant:

\[ \text{Pose Space} \Rightarrow \text{Displacements } \Delta \mathbf{x} \]
from \( \{(\text{PSCoord}_1, \Delta \mathbf{x}_1), ..., (\text{PSCoord}_N, \Delta \mathbf{x}_N)\} \)

Scattered Data Interpolation:
Interpolation of a set of irregularly located data points

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

Shepard’s Method

\[ \hat{d}(x) = \sum_{k=1}^{N} \frac{w_k(x)d_k}{\sum_{k=1}^{N} w_k(x)} \]

\[ w_k(x) = \|x - x_k\|^{-p}, \quad p > 0 \]

\(d_1, \ldots, d_N\) are the given data points at \(x_1, \ldots, x_N\).
Weight function
Flat zones
Average at far points
Shepard’s Method

Properties:

• **Singular** at data points \( \mathbf{x}_k \) with

  \[
  w_k(x) = \|x - \mathbf{x}_k\|^{-p} \text{ with } p > 0
  \]

• 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

  \[
  \hat{d}(\infty) = \frac{\sum_{k=1}^{N} w_k(\infty) d_k}{\sum_{k=1}^{N} w_k(\infty)} \quad = \frac{\sum_{k=1}^{N} d_k}{N}
  \]
Radial Basis Functions

...also called Hardy’s multiquadratics

\[ \hat{d}(x) = \sum_{k=1}^{N} w_k \psi \left( \| x - x_k \| \right) \]

\( w_1, \ldots, w_N \) from \( d_1, \ldots, d_N \) at \( x_1, \ldots, x_N \), solving a system of linear equations:

\[
\begin{pmatrix}
  d_1 \\
  \vdots \\
  d_N
\end{pmatrix}
= \begin{pmatrix}
  \psi \left( \| x_1 - x_k \| \right) & \cdots & \psi \left( \| x_N - x_1 \| \right) \\
  \vdots & \ddots & \vdots \\
  \psi \left( \| x_1 - x_N \| \right) & \cdots & \psi \left( \| x_N - x_N \| \right)
\end{pmatrix}
\begin{pmatrix}
  w_1 \\
  \vdots \\
  w_N
\end{pmatrix}
\]

\[ d = \Psi w \]
Choice of Radial Basis Function

\[
\psi(r) = \begin{cases} 
  r & \text{linear} \\
  r^3 & \text{cubic} \\
  (r^2 + \sigma^2)^{-\alpha}, \alpha > 0 & \text{localized} \\
  r^2 \ln(r) & \text{Thin-plate spline function} \\
  (r^2 + \sigma^2)^{\beta}, 0 < \beta < 1 & \end{cases}
\]
Gaussian Radial Basis

\[ \psi(r) = \exp\left(\frac{-r^2}{2\sigma^2}\right) \quad r = \|x - x_k\| \]

\[ \Rightarrow \quad \hat{d}(x) = \sum_{k=1}^{N} w_k \exp\left(\frac{-\|x - x_k\|^2}{2\sigma^2}\right) \]
Gaussian Basis Function
Gaussian Radial Basis

$\text{sigma} = 0.55$
Gaussian Radial Basis

Properties:
- smooth
- localized: \( \psi(r) = 0 \) for \( |r| \to \infty \)
  \( \Rightarrow \) \( \sigma \) width of falloff is adjustable
- relatively fast to compute (table)

\[
\psi(r) = \exp\left(\frac{-r^2}{2\sigma^2}\right)
\]
Pose Space Deformation

[Summary]

Deformation model:

Interpolant:

Pose Space $\Rightarrow$ 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

Recursion
(Iterative layered refinement)

Interpolation Parameters

Evaluation

1: (PS coord., $\Delta x$)

N: (PS coord., $\Delta x$)
Performance

For N poses:

- Preprocessing phase: \( \mathbf{d} = \Psi \mathbf{w} \)
  - \( N \times N \) Matrix \( \Psi \) must be inverted
  - Matrix-vector multiplication \( \mathbf{w} = \Psi^{-1} \mathbf{d} \)
    \( (\forall \) component of \( \forall \) displaced vertex)

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

For N poses:

Every vertex stores

N×3 weights

(N weights ∀ component of a vertex)

N pose space coordinates of ex.

\[ \hat{d}(\mathbf{x}) = \sum_{k=1}^{N} w_k \psi\left(\|\mathbf{x} - \mathbf{x}_k\|\right) \]
Memory Requirements

For $N$ poses:

Every vertex stores $N \times 3$ weights

$(N$ weights $\forall$ component of a vertex$)$

$N$ pose space coordinates of ex.

$$\hat{d}(x) = \sum_{k=1}^{N} w_k \psi \left( \| x - x_k \| \right)$$
Memory Requirements

For N poses:

Every vertex stores

Nx3 weights

(N weights ∀ component of a vertex)

N pose space coordinates of ex.

\[ \hat{d}(x) = \sum_{k=1}^{N} w_k \psi\left(\|x - x_k\|\right) \]
Pros & Cons

Pros:
- Wide range of deformations
- Arbitrary Pose Space Axes
- Real-time synthesis
- Relatively simple to implement
- Control on interpolation ($\sigma$)
- Direct manipulation (Iterative layered refinement)
Pros & Cons

Cons:

• Accuracy is reliant on the modeler/ animator
• $\sigma$ is manually tuned
• Performance
• Memory requirements
Critiques

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

- If \( N \) poses then 3 \( N \times N \) matrices must be inverted for each control vertex?
  Only one \( N \times N \) matrix must be inverted \( \Psi \)

- Close coordinates in PS results in a numerically unstable matrix \( \Psi \)
  (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
Simone Croci

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