Computer Animation
Algorithms and Techniques

Interpolation-based animation
Interpolation based animation

Key-frame systems – in general

Interpolating shapes
  Deforming an single shape
  3D interpolation between two shapes
  Morphing – deforming an image
Keyframing - interpolating values

Simple key frames in which each curve of a frame has the same number of points as its counterpart in the other frame.

Keys and three intermediate frames with linear interpolation of a single point (with reference showing the progression of the interpolation in x and y)
Keyframing

keys, in-betweens
track-based
Avars – articulation variables

variable name: A

value

frames
Keyframing curves
Time-Curve interpolation

Implement using surface patch technology
Time-Curve interpolation

Establish point correspondence
Time-Curve interpolation

Define time – space-curve “patches”

Interpolate in one dimension for curve (spatially)
Interpolate in other dimension temporally
Object interpolation

Correspondence problem
Interpolation problem

1. Modify shape of object interpolate vertices of different shapes

2. Interpolate one object into second object

3. Interpolate one image into second image
Object Modification

Modify the vertices directly  

OR

2D grid-based deforming
Free Form Deformations
Skeletal bending
Global transforms

Modify the space the vertices lie in

Vertex warping
Warping

Displacement of seed vertex

Attenuated displacement propagated to adjacent vertices
Power functions
For attenuating warping effects

\[ S(i) = 1.0 - \left( \frac{i}{n+1} \right)^{k+1} \quad k \geq 0 \]

\[ = (1.0 - \frac{i}{n+1})^{-k+1} \quad k < 0 \]
2D grid-based deforming

Assumption
Easier to deform grid points than object vertices
2D grid-based deforming

Inverse bilinear mapping (determine u,v from points)
2D grid-based deforming
2D skeleton-based bending
2D skeleton-based bending

\[ r = \frac{d_2}{d_1} \]

\[ L_1 \]
\[ L_2 \]
\[ L_3 \]
2D skeleton-based bending
Global Transformations

Common linear transform of space

\[ p' = Mp \]

In GT, Transform is a function of where you are in space

\[ p' = M(p)p \]
Global Transformations

Original object

\[ s(z) = \frac{\text{maxz} - z}{\text{maxz} - \text{minz}} \]
\[ x' = s(z)x \]
\[ y' = s(z)y \]
\[ z' = z \]

Tapered object

\[ \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} s(z) & 0 & 0 \\ 0 & s(z) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \]

\[ P' = M(p)p \]

Rick Parent

Computer Animation
Global Transformations

\[ k = \text{twist factor} \]
\[ x' = x\cos(kz) - y\sin(kz) \]
\[ y' = x\sin(kz) + y\cos(kz) \]
\[ z' = z \]
Global Transformations

z above $z_{\text{min}}$: rotate $\theta$

z between $z_{\text{min}}$, $z_{\text{max}}$: Rotate from 0 to $\theta$

z below $z_{\text{min}}$: no rotation
Compound global transformations
# Free-Form Deformations (FFDs)

<table>
<thead>
<tr>
<th>2D grid-based deforming</th>
<th>FFDs</th>
</tr>
</thead>
<tbody>
<tr>
<td>2D grid</td>
<td>3D grid</td>
</tr>
<tr>
<td>bi-linear interpolation</td>
<td>tri-cubic interpolation</td>
</tr>
</tbody>
</table>
Free-Form Deformations

Embed object in rectilinear grid
Free-Form Deformations

Register points in grid: cell x, y, z; (s, t, u)
Free-Form Deformations

As in Bezier curve interpolation
Continuity controlled by coplanarity of control points
FFDs: alternate grid organizations
FFDs: bending

Bulging

Bending
FFDs
hierarchical

Working at a coarser level

Working at a finer level

Rick Parent
Computer Animation
FFDs – as tools to design shapes

Undeformed object

Deformed object
FFDs

Animate by passing over object
FFDs

Animate by passing object through FFD

Object traversing the logical FFD coordinate space
Object traversing the distorted space

Rick Parent

Computer Animation
FFDs

Facial animation by manipulating FFD
FFDs

Exo-muscular system
Skeleton -> changes FFD -> changes skin

Initial configuration

Surface distorted after joint articulation

Rick Parent

Computer Animation
Interpolate between 2 objects

Correspondence problem: what part of one object to map into what part of the other object

How to handle objects of different genus?
Volumetric approaches with remeshing

Some surface-based approaches
Slice along one dimension; interpolate in other two
Map both to sphere
Recursively divide into panels
Object interpolation

For cylinder-like objects
Radial mapping

If central axis intersects polygonal slice inside kernel
Then simple radial mapping possible
Object interp

Sampling Object 1 along rays

Sampling Object 2 along rays

Points interpolated halfway between objects

Resulting object

Rick Parent

Computer Animation
Object interpolation
Object interp

Sampling Object 1 along rays

Sampling Object 2 along rays

Points interpolated halfway between objects

Resulting object
Original shapes sliced into contours

Interpolated shapes
Object interpolation

Spherical mapping to establish matching edge-vertex topology

1. Map to sphere
2. Intersect arc-edges
3. Retriangulate
4. Remap to object shapes
5. Vertex-to-vertex interpolation
Map to sphere

partial mesh from Object A

FB

partial mesh from Object B

VA
Object interpolation
Object interpolation - recursive sheets

Continually add vertices to make corresponding boundaries have an equal number
Object interpolation
Object interp

First vertex of boundary

<table>
<thead>
<tr>
<th>Normalized distances</th>
<th>Normalized distances</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0.00</td>
<td>0 0.00</td>
</tr>
<tr>
<td>1 0.15</td>
<td>1 0.30</td>
</tr>
<tr>
<td>2 0.20</td>
<td>2 0.55</td>
</tr>
<tr>
<td>3 0.25</td>
<td>3 0.70</td>
</tr>
<tr>
<td>4 0.40</td>
<td></td>
</tr>
<tr>
<td>5 0.70</td>
<td></td>
</tr>
</tbody>
</table>

Boundary after adding additional vertices
Morphing

Image blending
Move pixels to corresponding pixels
Blend colors
Morphing

Image A

Image B

Image A with grid points and curves defined

Image B with grid points and curves defined
Morphing

Interpolate intermediate grid for time 't'

Warp source image to intermediate grid

Warp destination image to intermediate grid

Cross-dissolve the two images
Morph

Source image grid

use x-coordinates of these points

Intermediate grid

use y-coordinates of these points

Auxiliary grid

Details showing relationship of source image grid point, intermediate grid point, and auxiliary grid point

Rick Parent

Computer Animation
Morphing

Source image grid

Auxiliary grid

Scanline

Grid coordinates

Pixel coordinates

Pixel coordinate to grid coordinate graph for source image

Pixel coordinate to grid coordinate graph for auxiliary image

Use the graph to see where the column indices map to image pixels. (Here, half of pixel 3 and all of pixels 4 and 5 are useful)

Use the graph to determine the image pixel’s range in terms of the column indices (pixel 6 is shown)
Morphing

Auxiliary grid

Intermediate grid

“column lines”

grid coordinates

pixel coordinates

Use row index coordinates to determine the pixel coordinates in auxiliary image

For a given pixel in the intermediate image, determine the coordinates in terms of row indices
Morphing

Source image sequence

interpolated grid

intermediate grid

Destination image sequence

interpolated grid

key grids

Rick Parent

Computer Animation
Morphing: feature based

Given: corresponding user-defined feature lines in source and destination images
Morphing: feature based

Locate each pixel relative to each feature line in source and destination images

\[ v = (P - P_1) \cdot \frac{(P_2 - P_1)}{|P_2 - P_1|^2} \]

\[ u = \left| (P - P_1) \times \frac{(P_2 - P_1)}{|P_2 - P_1|^2} \right| \]

\[ T = Q_2 - Q_1 \]
\[ S = (T_y, -T_x) \]
\[ Q = Q_1 + uS + vT \]
Morphing: feature based

First example

Second example