Image Morphing

Michael Kazhdan

(601.457/657)

HB Ch. 16.5

Feature Based Image Metamorphosis,
Beier and Neely 1992
Image Morphing

- Animate transition between two images

Figure 16-9
Transformation of an STP oil can into an engine block. (Courtesy of Silicon Graphics, Inc.)
Image Morphing

- Animate transition between two images

Two Components:
- Blending
- Warping
Blending

Combine images using an $\alpha$-blend ($\alpha \in [0,1]$):

$$\text{blend}(i, j, \alpha) = (1 - \alpha) \cdot \text{Img}_1(i, j) + \alpha \cdot \text{Img}_2(i, j)$$
Image Warping

Deform $\text{Img}_1$ so its shape matches that of $\text{Img}_2$...

$\alpha = 0.0$ $\alpha = 0.5$ $\alpha = 1.0$
Image Warping

Deform \( \text{Img}_1 \) so its shape matches that of \( \text{Img}_2 \)…

\[
\alpha = 0.0 \\
\alpha = 0.5 \\
\alpha = 1.0
\]
Image Warping

Deform $\text{Img}_2$ so its shape matches that of $\text{Img}_1$...

$\text{Img}_1$

$\alpha = 0.0$

$\text{Img}_2$

$\alpha = 1.0$

$\alpha = 0.5$

$\alpha = 0.0$
Image Morphing

... then blend

\[ \alpha = 0.0 \]

\[ \alpha = 0.5 \]

\[ \alpha = 1.0 \]
Image Morphing

• The warping step is the hard one
  ◦ Aim to align features in images

How do we specify the mapping for the warp?

H&B Figure 16.9
Feature-Based Warping

- [Beier & Neeley, 1992] use a pair of lines to specify the warp
  - Given \( p \) in destination image, where is \( p' \) in the source?
  - Describe \( p \) relative to the destination line
  - Map the description to the source

\[ u \text{ is a signed fraction} \]
\[ v \text{ is a signed length (in pixels)} \]
Feature-Based Warping

How do we calculate $u$ and $v$?

Recall:

$$\langle v_1, v_2 \rangle = x_1 \cdot x_2 + y_1 \cdot y_2$$

$$= \|v_1\| \cdot \|v_2\| \cdot \cos \theta$$

The (signed) length of the projection of $v_1$ on the line through $v_2$, scaled by the length of $v_2$.

$\Rightarrow$ The signed length of the projection of $v_1$ onto the line through $v_2$ is:

$$\frac{\langle v_1, v_2 \rangle}{\|v_2\|}$$
How do we calculate $u$ and $v$?

$\langle v_1, v_2 \rangle$ is the (signed) length of the projection of $v_1$ on the line through $v_2$, scaled by the length of $v_2$.

$$v = \frac{\langle p - s, (t - s)\perp \rangle}{\| (t - s)\perp \|}$$
How do we calculate $u$ and $v$?

$\langle v_1, v_2 \rangle$ is the (signed) length of the projection of $v_1$ on the line through $v_2$, scaled by the length of $v_2$.

$$u = \frac{\langle p - s, t - s \rangle}{\|t - s\|} \cdot \frac{1}{\|t - s\|}$$

Remember: $u$ is a fraction
Feature-Based Warping

- [Beier & Neeley, 1992] use a pair of lines to specify the warp
  - Given \( p \) in destination image, where is \( p' \) in the source?
  - Describe \( p \) relative to the destination line
  - Map the description to the source

\[
\begin{align*}
\text{Source image} & \quad \rightarrow \quad \text{Destination image} \\
\text{\( u \) is a signed \underline{fraction}} & \\
\text{\( v \) is a signed \underline{length} (in pixels)}
\end{align*}
\]
Warping with One Line Pair

• What happens to the “F”?

Translation!
Warping with One Line Pair

- What happens to the “F”?

Non-uniform scale!
Warping with One Line Pair

• What happens to the “F”? 

Rotation!
Warping with One Line Pair

• What happens to the “F”?

What types of affine transformations can’t be specified?
Warping with One Line Pair

• Can’t specify arbitrary scales, skews, mirrors, angular changes…
Warping with Multiple Line Pairs

• Use weighted combination of points defined by each pair of corresponding lines

Beier & Neeley, Figure 4
Warping with Multiple Line Pairs

- Use weighted combination of points defined by each pair of corresponding lines
Warping with Multiple Line Pairs

- Use weighted combination of points defined by each pair of corresponding lines

\[ p' \text{ is a weighted average} \]
Weighting Effect of Each Line Pair

• Given a set of line pairs \( \{\mathit{Lin}[0], \ldots, \mathit{Lin}[N]\} \) and \( \{\mathit{Out}[0], \ldots, \mathit{Out}[N]\} \), to weight the contribution of each line pair, [Beier & Neeley, 1992] use:

\[
\text{weight}[i](p) \sim \left( \frac{\text{length}[i]^c}{a + \text{dist}[i](p)} \right)^b
\]

where:

• length\([i]\) is the length of line \( \mathit{Out}[i] \)
• dist\([i](p)\) is the distance from \( p \) to \( \mathit{Out}[i] \)
• \( a \) (small), \( b \in [0.5,2.0] \), \( c \in [0.0,1.0] \) are constants that control the warp
How do we calculate $\text{dist}$?

$$\text{dist}(p) = \begin{cases} 
\|v\| & \text{if } u \in [0,1] \\
\|p - s\| & \text{if } u < 0 \\
\|p - t\| & \text{if } u > 1 
\end{cases}$$
Warping

Given a source image and set of corresponding line segment pairs:

• Iterate over each pixel in the target
  ◦ For each pair of line segments
    » Compute the corresponding position in the source
    » Compute the weights
  ◦ Average to get the final source position
  ◦ Sample the source (at the source position) to get the color at the target pixel
Warping Pseudocode

Warp( Img_in, L_in[...], L_out[...] )
{
    foreach destination pixel p_out:
        p_in = (0,0)
        sum = 0
        for i = 1 to number of line pairs:
            q_in = p_out transformed by ( L_in[i], L_out[i] )
            p_in += q_in * weight[i]( p_out )
            sum += weight[i]( p_out )
        p_in /= sum
        Img_out(p_out) = Img_in( p_in )
    return Img_out
}
Morphing at $\alpha \in [0, 1]$

Given two images, given a set of corresponding line segment pairs, and an interpolation time $\alpha \in [0,1]$:

- Compute the $\alpha$-blend of the line segments (by blending the end-points)
- Warp the first image using the first set of line segments and the blended line segments
- Warp the second image using the second set of line segments and the blended line segments
- Compute the $\alpha$-blend of the warped images
Morphing at $\alpha \in [0, 1]$ Pseudocode

Morph($Img_0, L_0[N], Img_1, L_1[N], \alpha$)
{
    foreach $i \in \{1, \ldots, N\}$:
        $L[i] = \text{line } \alpha\text{-th of the way from } L_0[i] \text{ to } L_1[i]$

    $Warp_0 = \text{Warp}(Img_0, L_0[], L[])$
    $Warp_1 = \text{Warp}(Img_1, L_1[], L[])$

    return $(1-\alpha) \times Warp_0 + \alpha \times Warp_1$
}
Animation Pseudocode

```plaintext
Animate( Img₀, L₀[N] , Img₁ , L₁[N] , Imgs_out[T+1] )
{
    foreach t∈{0,...,T}:
        Imgs_out[t] = Morph(Img₀ , L₀[N] , Img₁ , L₁[N] , t/T )
}
```
Morphing

Check out Michael Jackson’s “Black or White” video at:

https://www.youtube.com/watch?v=pTFE8cirkdQ
[Beier & Neeley, 1992] Example

Image_0

Result

Image_1
[Beier & Neeley, 1992] Example

$\text{Img}_0$

$\text{Warp}_0$

Result

$\text{Img}_1$

$\text{Warp}_1$
[Beier & Neeley, 1992] Example

\[ \text{Img}_0 \rightarrow \text{Warp}_0 \rightarrow \text{Result} \rightarrow \text{Img}_1 \rightarrow \text{Warp}_1 \]
Extensions:

- Apply to consecutive frames in a video to generate in-between frames (for slow-motion).
- Automate the calculation of correspondences.

Limitations:

- Occlusions
Image Processing

• Quantization
  ○ Uniform quantization
  ○ Random dither
  ○ Ordered dither
  ○ Floyd-Steinberg dither

• Pixel operations
  ○ Add random noise
  ○ Compute luminance
  ○ Change contrast
  ○ Change saturation

• Filtering
  ○ Blur
  ○ Detect edges

• Sampling
  ○ Nyquist rate
  ○ Aliasing
  ○ Ideal filter

• Morphing
  ○ Blending
  ○ Warp