Morphing

Demo

- *Show morphing of Clinton to Bush*

General Comments

- Morphing is an example of image-based graphics, in which we manipulate images to produce new ones. This can produce greater realism than rendering.
  - Key idea is to interpolate images. This is done in other settings.
  - Of course, it's widely used.
- Morphing provides a good example of the connection between discrete and continuous. We take two discrete images and create a continuous set of intermediate ones.
- As in much of graphics, it is as much art as science.

Key Elements

- Transition from one image to another, with intermediate images that look good.
- Interpolation
  - Intensities interpolate from one image to another.
- Correspondence.
  - Without correspondence we just have "fade"
    - EG., phantom eyes.
  - Instead, we want to interpolation *position* as well as appearance of features.

Fading

- This is morphing with trivial correspondence.
- Images I, J. Generate n+1 images.
- Image K, at time k, is generated so:
  - K(x,y) = (1-k/n)*I(x,y) + (k/n)*J(x,y).
- *Demo*

Morphing – correspondence

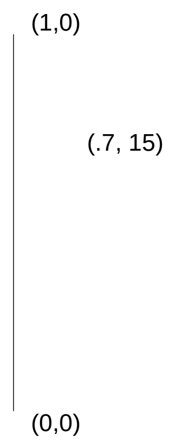
- Basic idea is we want to provide a dense correspondence between images.
- Then we fade intensities between corresponding pixels.
- We also move position of pixels.
- How do we specify correspondence?
  - This is basically a user interface question.

Morphing line segments

- Correspondence of end points.
- Linearly interpolate positions.
- This also implies a correspondence between all points on line.
  - Midpoint of one line is being linearly interpolated to midpoint of other.
- *Demo*

Point Correspondence from lines

- Impractical to put a line through every pixel.
- So we want correspondence of lines to imply similar correspondence for points near lines.
- We do this by defining coordinate system using lines.



1st end point is origin. Vector to second end point is (1,0). Vector of length one perpendicular to this is (0,1).

(Why not make basis vectors same length? Heuristic, but the idea is that the direction specified is compressed while other isn't. So if we match two line segments of different length, image will be compressed in direction of segments, but not in orthogonal direction.)

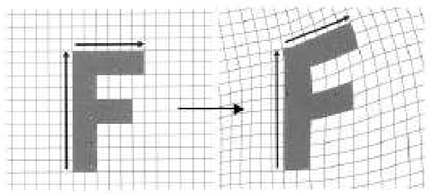
Line segments in two images define two coordinate systems. Two pixels correspond if they have the same coordinates.

Examples

- Morph an F to a blank page, without the fading, just to see the geometry.
- Translating line -> translation
- Rotating line -> Rotation
- Changing line size in one direction is compression (**not scaling).**

Warping with Multiple Line Pairs

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

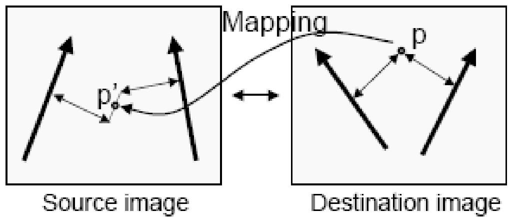


Beier & Neeley, Figure 4

(Funkhouser)

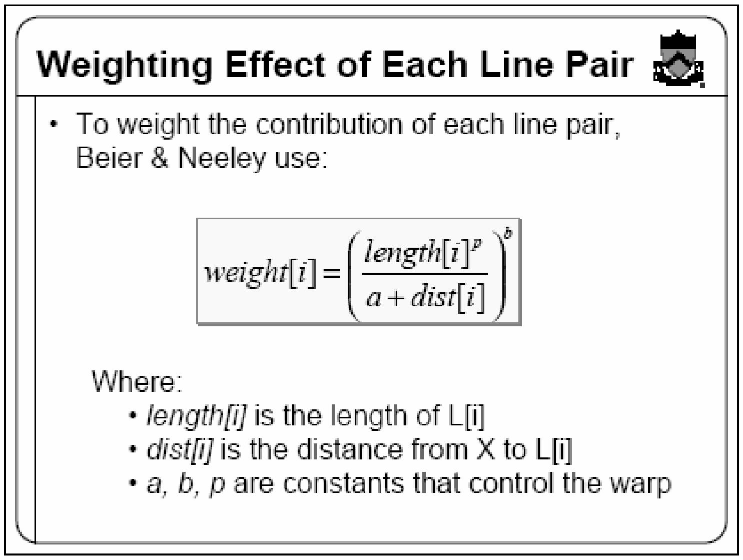
Warping with Multiple Line Pairs

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



p' is a weighted average

(Funkhouser)



Weighting Effect of Each Line Pair

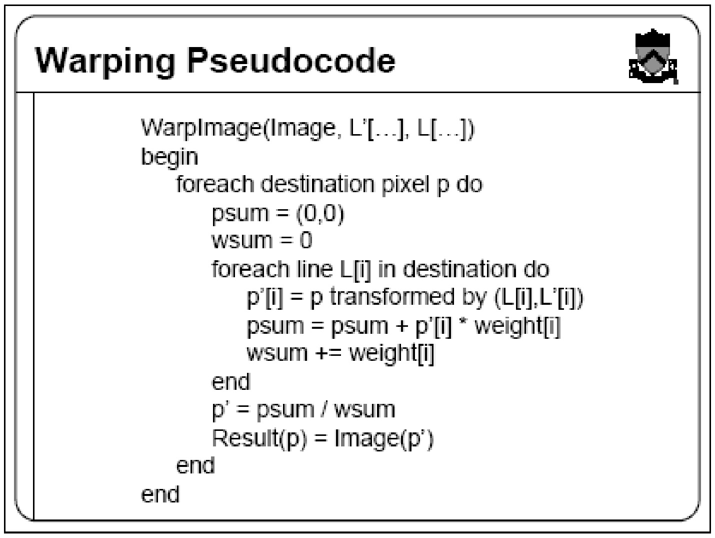
- To weight the contribution of each line pair, Beier & Neeley use:

$$weight[i] = \left(\frac{length[i]^p}{a + dist[i]}\right)^b$$

Where:
- *length[i]* is the length of L[i]
- *dist[i]* is the distance from X to L[i]
- *a, b, p* are constants that control the warp

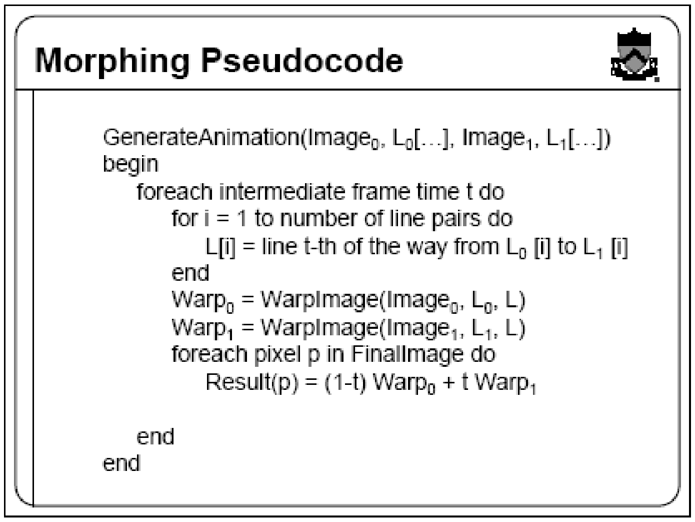
Strategy

- Interpolate lines to get position in fresh image.
- Warp each original image onto fresh image.
- Interpolate (one step of a fade) between two images.



Warping Pseudocode

```
WarpImage(Image, L'[…], L[…])
begin
    foreach destination pixel p do
        psum = (0,0)
        wsum = 0
        foreach line L[i] in destination do
            p'[i] = p transformed by (L[i],L'[i])
            psum = psum + p'[i] * weight[i]
            wsum += weight[i]
        end
        p' = psum / wsum
        Result(p) = Image(p')
    end
end
```



Morphing Pseudocode

```
GenerateAnimation($Image_0$, $L_0$[…], $Image_1$, $L_1$[…])
begin
    foreach intermediate frame time t do
        for i = 1 to number of line pairs do
            L[i] = line t-th of the way from $L_0$ [i] to $L_1$ [i]
        end
        $Warp_0$ = WarpImage($Image_0$, $L_0$, L)
        $Warp_1$ = WarpImage($Image_1$, $L_1$, L)
        foreach pixel p in FinalImage do
            Result(p) = (1-t) $Warp_0$ + t $Warp_1$

    end
end
```

Announcements

- Completed Problem Set 3 available later today.
- I'll be gone next Monday-Wednesday.
  - Carlos will teach on Tuesday
  - Office hours next week by appointment

Other morphing applications Blanz and Vetter (SIGGRAPH)

- Build 3D morphable model of faces.
  - Dense correspondence between (~100) 3D face models.
  - Morphing between them can create huge range of faces
  - (all pictures from their paper).

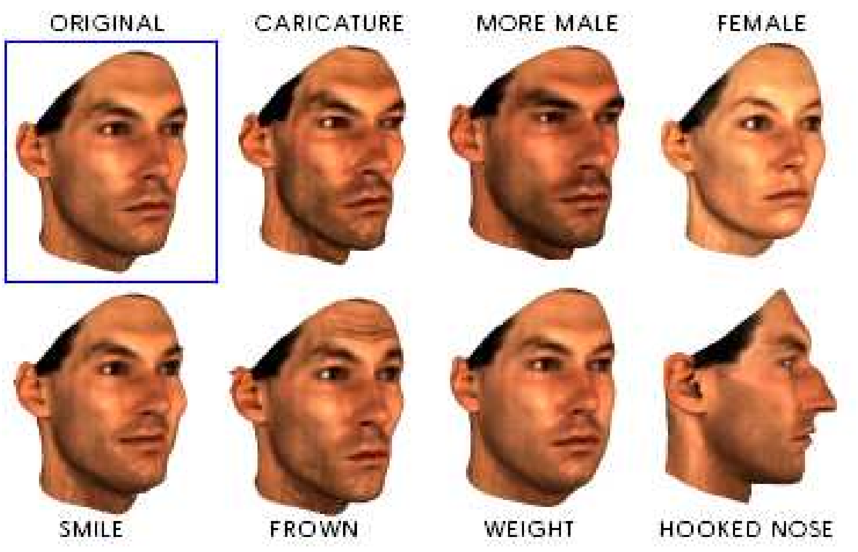
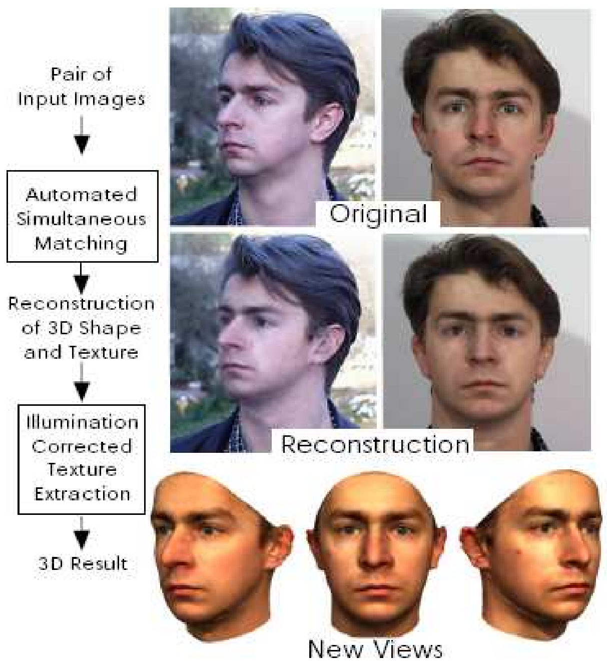


Figure 3: Variation of facial attributes of a single face. The appearance of an original face can be changed by adding or subtracting shape and texture vectors specific to the attribute.


Pair of Input Images
Automated Simultaneous Matching
Reconstruction of 3D Shape and Texture
Illumination Corrected Texture Extraction
3D Result
Original
Reconstruction
New Views

Can fit an image to a model.

Find 3D morph (+ lighting, pose) that best fits image.

Then, model can synthesize new images.

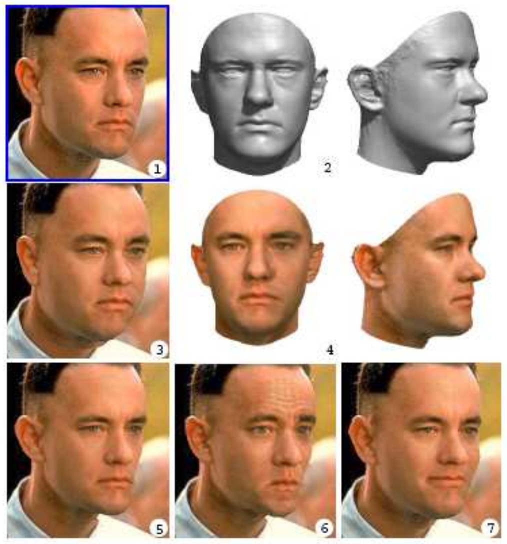


Figure 6: Matching a morphable model to a single image (1) of a face results in a 3D shape (2) and a texture map estimate. The texture estimate can be improved by additional texture extraction (4). The 3D model is rendered back into the image after changing facial attributes, such as gaining (3) and loosing weight (5), frowning (6), or being forced to smile (7).

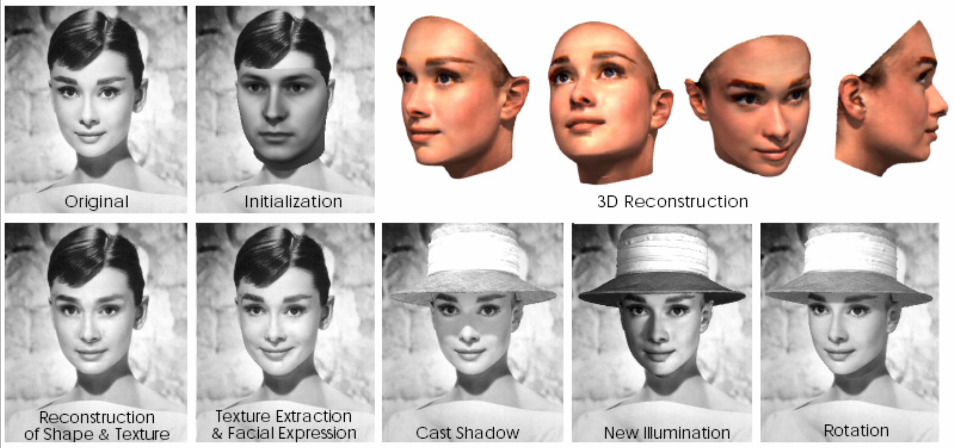


Figure 7: After manual initialization, the algorithm automatically matches a colored morphable model (color contrast set to zero) to the image. Rendering the inner part of the 3D face on top of the image, new shadows, facial expressions and poses can be generated.

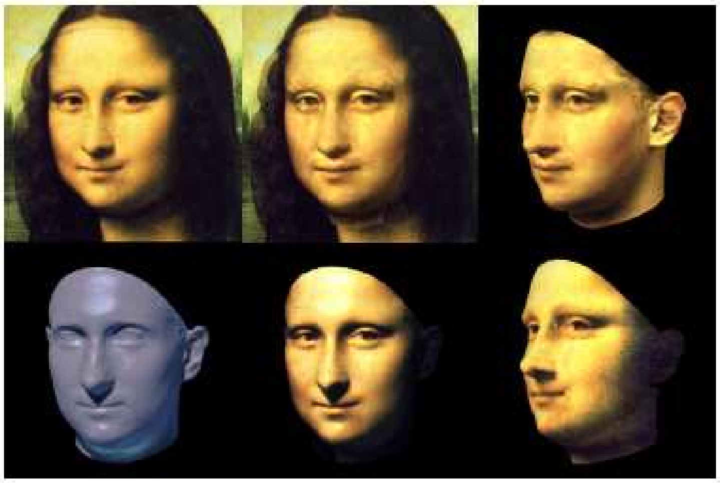


Figure 8: Reconstructed 3D face of Mona Lisa (top center and right). For modifying the illumination, relative changes in color (bottom left) are computed on the 3D face, and then multiplied by the color values in the painting (bottom center). Additional warping generates new orientations (bottom right, see text), while details of the painting, such as brush strokes or cracks, are retained.