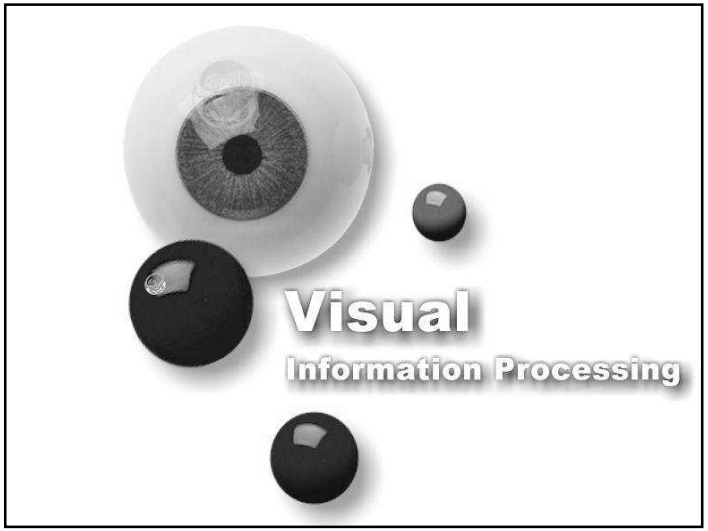
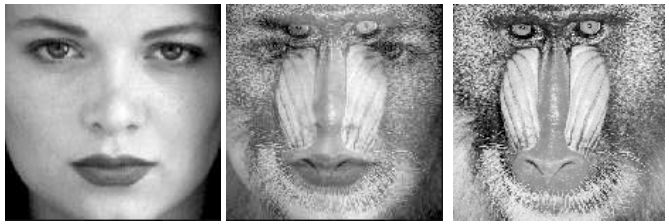


Interactive Computer Graphics

- Lecture 15: Warping and Morphing

Warping and Morphing

- What is
  - warping ?
  - morphing ?



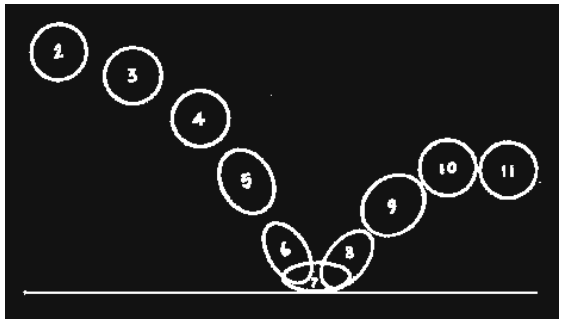
Warping

- The term warping refers to the geometric transformation of graphical objects (images, surfaces or volumes) from one coordinate system to another coordinate system.
- Warping does not affect the attributes of the underlying graphical objects.
- Attributes may be
  - color (RGB, HSV)
  - texture maps and coordinates
  - normals, etc.

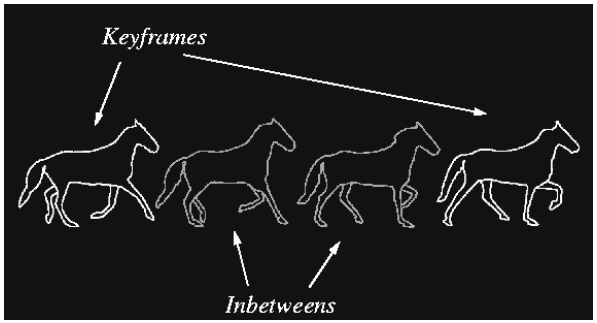
Morphing

- The term morphing stands for metamorphosing and refers to an animation technique in which one graphical object is gradually turned into another.
- Morphing can affect both the shape and attributes of the graphical objects.

Examples

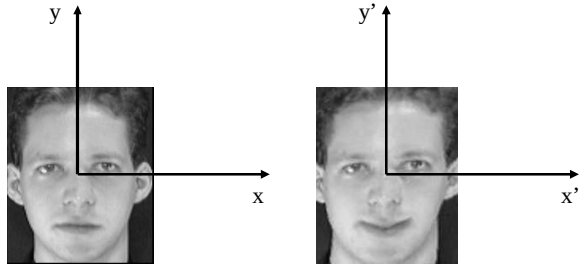


Examples



Mapping

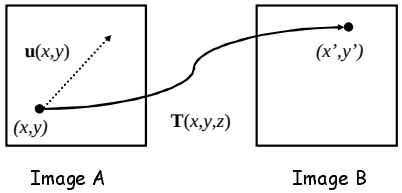
- Define transformation which maps every point (x, y) in the source coordinates to the corresponding point (x', y') in the destination coordinates (or vice-versa, if invertible)



Warping

- Transformation $T:(x,y)\rightarrow(x',y')$ which defines the spatial relationship between two images:

$$(x',y')=\mathbf{T}(x,y)=(x,y)+\mathbf{u}(x,y)$$



Transformations

- Dimensions of transformation
  - 1D: curves
  - 2D: images
  - 3D: volumes
- Types of transformations
  - rigid
  - affine
  - polynomial
    - quadratic
    - cubic
  - splines

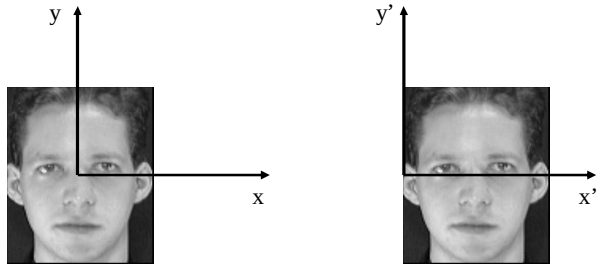
Transformations in 2D: Rigid

- Rigid transformation (3 degrees of freedom)

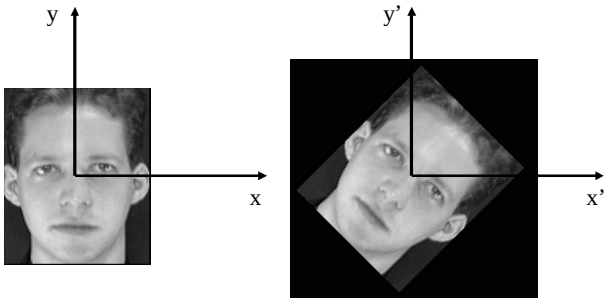
$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\alpha & -\sin\alpha & t_x \\ \sin\alpha & \cos\alpha & t_y \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \mathbf{R} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \\ 0 \end{pmatrix}$$

- t_x and t_y describe the two translations in x and y
- α describes the rotation around the center of the image

Transformations in 2D: Rigid



Transformations in 2D: Rigid



Transformations in 3D: Rigid

- Rigid transformation (6 degrees of freedom)

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} r_{01} & r_{02} & r_{03} & t_x \\ r_{11} & r_{12} & r_{13} & t_y \\ r_{21} & r_{22} & r_{23} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = T^x_{rigid} \cdot T^y_{rigid} \cdot T^z_{rigid} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \\ t_z \\ 0 \end{pmatrix}$$

- t_x, t_y, t_z describe the 3 translations in x, y and z
- r_{11}, ..., r_{33} describe the 3 rotations around x, y, z

Transformations in 3D: Rigid

$$\mathbf{T}^x_{rigid} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad \mathbf{T}^y_{rigid} = \begin{pmatrix} \cos\alpha & 0 & \sin\alpha & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\alpha & 0 & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\mathbf{T}^z_{rigid} = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 & 0 \\ \sin\alpha & \cos\alpha & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

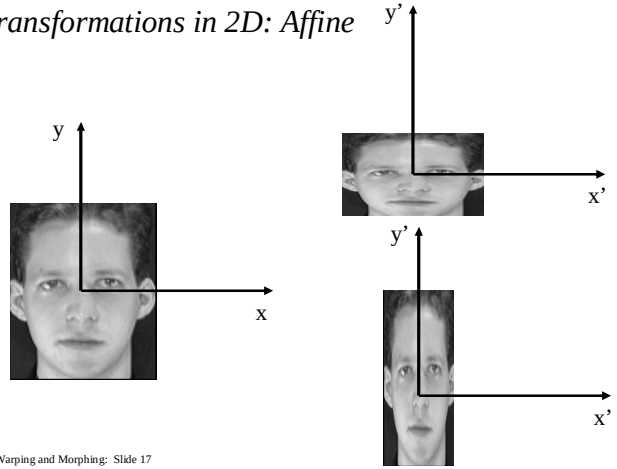
Transformations in 2D: Affine

- Affine transformations (6 degrees of freedom)

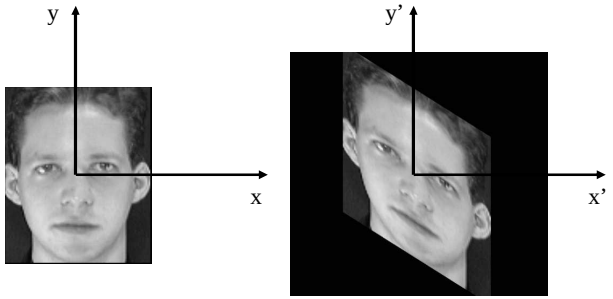
$$\mathbf{T}_{scale} = \begin{pmatrix} sx & 0 & 0 \\ 0 & sy & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad \mathbf{T}_{shear} = \begin{pmatrix} 1 & sh & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\mathbf{T}(x, y) = T_{shear} \cdot T_{scale} \cdot T_{rigid} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

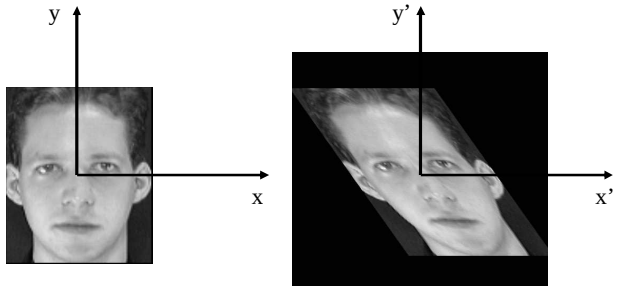
Transformations in 2D: Affine



Transformations in 2D: Affine



Transformations in 2D: Affine



Transformations in 3D: Affine

- Affine transformations (12 degrees of freedom)

$$\mathbf{T}_{scale} = \begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad \mathbf{T}_{shear}^{xy} = \begin{pmatrix} 1 & 0 & sh_x & 0 \\ 0 & 1 & sh_y & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\mathbf{T}(x, y, z) = \mathbf{T}_{shear} \cdot \mathbf{T}_{scale} \cdot \mathbf{T}_{rigid} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

Non-rigid transformations

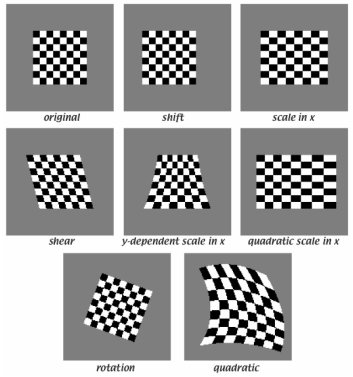
- Quadratic transformation (30 degrees of freedom)

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} r_{00} & \cdots & r_{08} & r_{09} \\ r_{10} & \cdots & r_{18} & r_{19} \\ r_{20} & \cdots & r_{28} & r_{29} \\ 0 & \cdots & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x^2 \\ y^2 \\ \vdots \\ 1 \end{pmatrix}$$

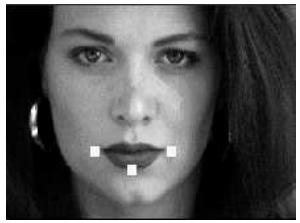
Non-rigid transformations

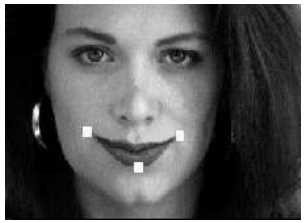
- Can be extended to other higher-order polynomials:
  - 3rd order (60 DOF)
  - 4th order (105 DOF)
  - 5th order (168 DOF)
- Problems:
  - can model only global shape changes, not local shape changes
  - higher order polynomials introduce artifacts such as oscillations

Non-rigid transformations

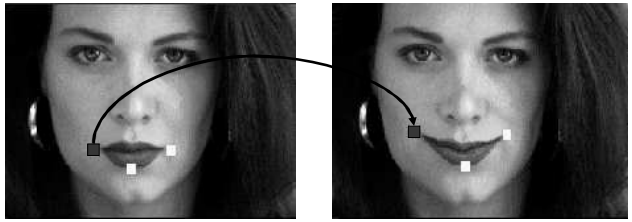


Non-rigid transformations

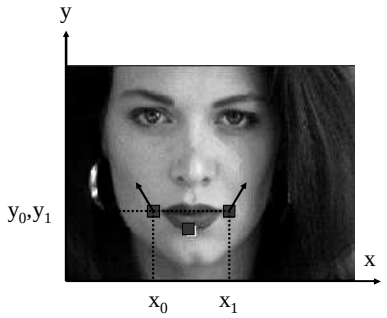




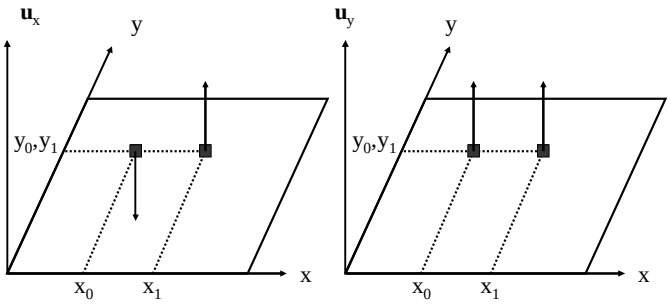
Non-rigid transformations: Correspondences



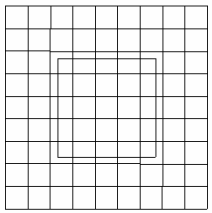
Non-rigid transformations: Correspondences



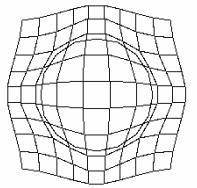
Non-rigid transformations



Representing deformations

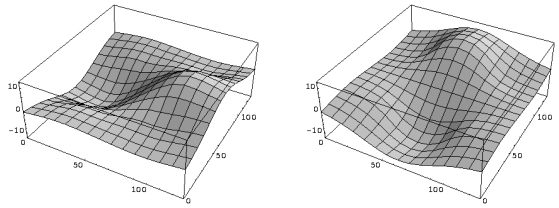


Before deformation



After deformation

Representing deformations

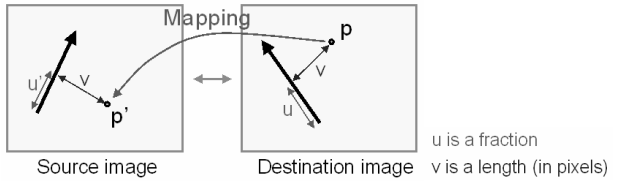


Displacement in the horizontal direction

Displacement in the vertical direction

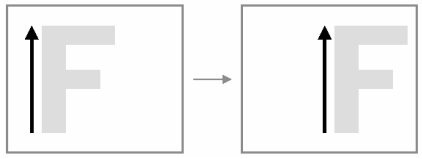
Feature-Based Warping: Beier-Neeley

- Beier & Neeley use pairs of lines to specify warp
  - Given p in dst image, where is p' in source image?



Warping with One Line Pair: Beier-Neeley

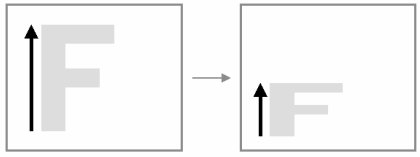
- What happens to the "F" ?



Translation !

Warping with One Line Pair (cont.): Beier-Neeley

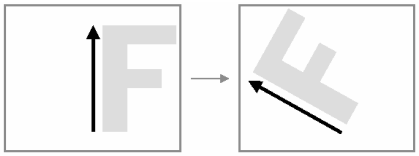
- What happens to the "F" ?



Scale !

Warping with One Line Pair (cont.): Beier-Neeley

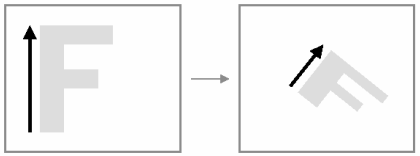
- What happens to the "F" ?



Rotation !

Warping with One Line Pair (cont.): Beier-Neeley

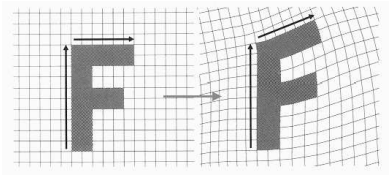
- What happens to the "F" ?



In general, similarity transformations

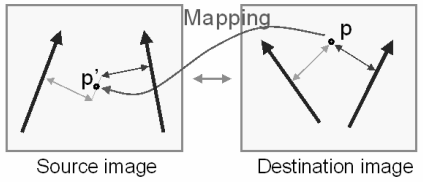
Warping with Multiple Line Pairs: Beier-Neeley

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



Warping with Multiple Line Pairs: Beier-Neeley

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



p' is a weighted average

Weighting Effect of Each Line Pair: Beier-Neeley

- To weight the contribution of each line pair

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

  - 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

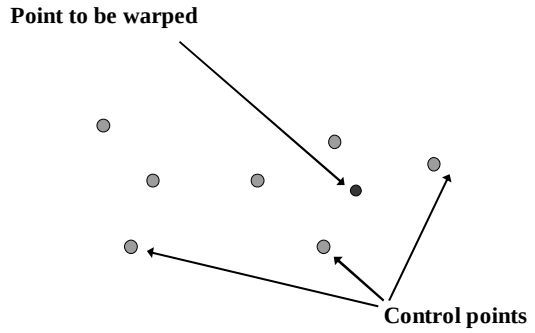
Warping Pseudocode: Beier-Neeley

```
WarpImage(Image, L'[], L[])
begin
    foreach destination pixel p do
        psum = (0, 0)
        wsum = (0, 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: Beier-Neeley

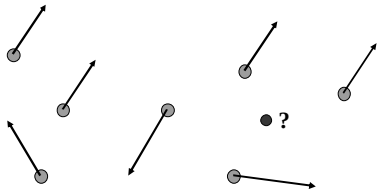
```
GenerateAnimation(Image0, L0[], Image1, L1[])
begin
  foreach intermediate frame time t do
      for i = 1 to number of line pairs do
          L[i] = line t-th of the way from L0[i] to L1[i]
      end
      Warp0 = WarpImage(Image0, L0, L)
      Warp1 = WarpImage(Image1, L1, L)
      foreach pixel p in FinalImage do
          Result(p) = (1-t)Warp0 + tWarp1
      end
  end
end
```


Non-rigid transformation



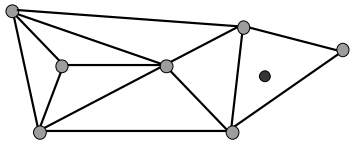
Non-rigid transformation

- For each control point we have a displacement vector
- How do we interpolate the displacement at a pixel?



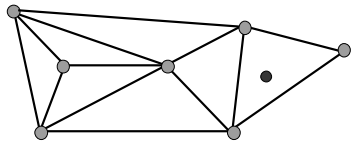
Non-rigid transformation: Piecewise affine

- Partition the convex hull of the control points into a set of triangles



Non-rigid transformation: Piecewise affine

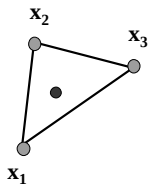
- Partition the convex hull of the control points into a set of triangles



Non-rigid transformation: Piecewise affine

- Find triangle which contains point **p** and express in terms of the vertices of the triangle:

$$\mathbf{p} = \mathbf{x}_1 + \alpha(\mathbf{x}_2 - \mathbf{x}_1) + \beta(\mathbf{x}_3 - \mathbf{x}_1)$$

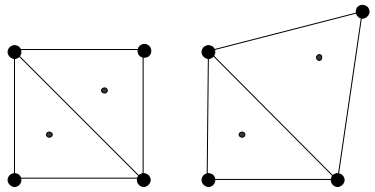


Non-rigid transformation: Piecewise affine

- Or $\mathbf{p} = \gamma\mathbf{x}_1 + \alpha\mathbf{x}_2 + \beta\mathbf{x}_3$ with $\gamma = 1 - (\alpha + \beta)$
- Under the affine transformation this point simply maps to

$$\mathbf{p}' = \gamma\mathbf{x}_1' + \alpha\mathbf{x}_2' + \beta\mathbf{x}_3'$$

Non-rigid transformation: Piecewise affine



Non-rigid transformation: Piecewise affine

- Problem: Produces continuous deformations, but the deformation may not be smooth. Straight lines can be kinked across boundaries between triangles