

Geometrical Manipulation & Morphing

Min Wu

Electrical & Computer Engineering
Univ. of Maryland, College Park

<http://www.ece.umd.edu/class/enee631/>

minwu@eng.umd.edu



Last Time

- Continued on image enhancement
 - Image sharpening
 - LPF/HPF/BPF
 - Image interpolation
- Edges
- Textures
- Today
 - Geometric manipulation and warping
 - Image morphing



Geometrical Manipulations of Images



Rotation, Translation, and Scaling

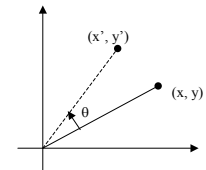
- R.S.T. of an image object

- Original pixel location $(x,y) \rightarrow$ New location (x',y')

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} \quad \text{translation by } \begin{bmatrix} t_x & t_y \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad \begin{array}{l} \text{rotate by } \theta \text{ around origin} \\ \text{counter-clockwise} \end{array}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad \text{scaling by } s_x \text{ and } s_y$$



Preserve length & angle

Uniform scaling $S_x = S_y$
(preserve angle and shape)
Differential scaling $S_x \neq S_y$

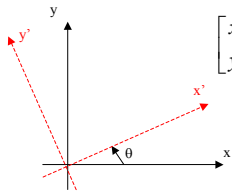


Rotation, Translation, and Scaling (cont'd)

- **Rotation and translation of image coordinates**

- Note the connections with rotation and translation of image objects

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} -t_x \\ -t_y \end{bmatrix} \quad \text{translate origin to } (t_x, t_y)$$



$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad \begin{array}{l} \text{rotate axis by } \theta \\ \text{counter - clockwise} \end{array}$$



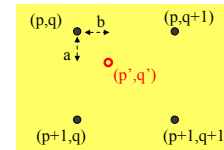
Implementation Issues of Geometric Transf.

- **Forward transform**

- Index mapping from input to output image
 - What if most values obtained for an output image are at fractional coordinate indices?

- **Reverse transform**

- Map integer indices of output image to input image
 - Get values of input image at fractional indices through interpolation



2-D Homogeneous Coordinate

- **Describe R.S.T. transform by $P' = MP + \underline{T}$**

- Need calculating intermediate coordinate values for successive transf.

- **Homogeneous coordinate**

- Allow R.S.T. represented by matrix multiplication operations
 - successive transf. can be calculated by combining transf. matrices
- Cartesian point $(x,y) \rightarrow$ Homogeneous representation $(s x', s y', s)$
 - represent the same pixel location for all nonzero parameter s
 - commonly use $s = 1$
- $f(x,y) = 0 \rightarrow$ homogeneous equation in $(s x', s y', s)$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \sim \begin{bmatrix} s x' \\ s y' \\ s \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$



R.S.T. in Homogeneous Coordinates

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- **Successive R.S.T.**

- Left multiply the basic transform matrices



Reflection

- Reflect about x-axis, y-axis, and origin

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- Reflect about $y=x$ and $y=-x$

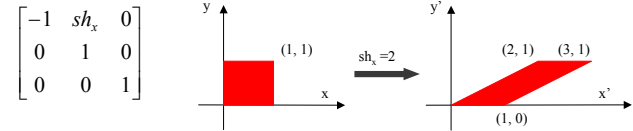
$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 0 & -1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- Reflect about a general line $y=ax+b$
 - Combination of translate-rotate \Rightarrow reflect \Rightarrow inverse rotate-translate



Shear

- Shear ~ a transformation that distorts the shape
 - Cause opposite layers of the object slide over each other
- Shear relative to x-axis



- Extend to shears relative to other reference lines



General Composite Transforms

- Combined R.S.T.
 - $\{a_{ij}\}$ is determined by R.S.T. parameters

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- Rigid-body transform
 - Only involve translations and rotations
 - 2x2 rotation submatrix is orthogonal
 - ♦ row vectors are orthonormal
- Extension to 3-D homogeneous coordinate
 - (sX, sY, sZ, s) with 4x4 transformation matrices



General Composite Transforms (cont'd)

- Affine transforms ~ 6 parameters

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- Projective transforms ~ 8 parameters
 - Cover more general geometric transformations between two planes
 - Widely used in computer vision problems such as image mosaicing, and synthesized views

$$\begin{bmatrix} sx' \\ sy' \\ s \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ c_1 & c_2 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad \Rightarrow \quad \underline{w} = \frac{A\underline{w} + \underline{b}}{1 + \underline{c}^T \underline{w}} \quad \underline{w} = [x', y']^T$$

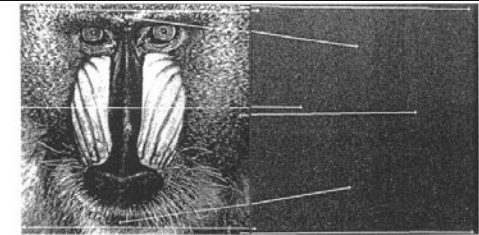


Non-linear Spatial Warping

- Analogous to “rubber sheet stretching”
 - Forward and reverse mapping of pixels’ coordinate indices
 $(x, y) \Leftrightarrow (x', y')$
- Polynomial warping
 - Extend affine transform to higher-order polynomial mapping
 - 2nd-order warping
 - ♦ $x' = a_0 + a_1 x + a_2 y + a_3 x^2 + a_4 xy + a_5 y^2$
 - ♦ $y' = b_0 + b_1 x + b_2 y + b_3 x^2 + b_4 xy + b_5 y^2$
- Spatial distortion in imaging system (lens)
 - Pincushion and Barrel distortion



Example of 2nd-order Polynomial Spatial Warping



(a) Source (b) Destination

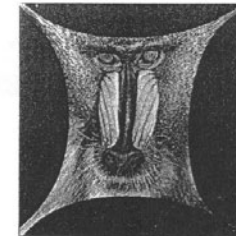


Illustration of Geometric Distortion

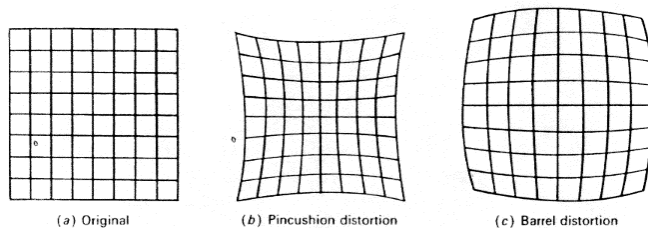


FIGURE 14.2-1. Example of geometric distortion.



Compensating Spatial Distortion in Imaging

- Control points
 - Coordinates before and after distortion are known
- Fit into polynomial warping model
 - Minimize the sum of squared error between a set of warped control points and the polynomial estimates
 - ♦ $x' = [x_1, x_2, \dots, x_M], Z = [1, x_1, y_1, x_1^2, x_1 y_1, y_1^2; \dots; 1, x_M, y_M, \dots]$
 - ♦ $E = (x' - Z a)^T (x' - Z a) + (y' - Z b)^T (y' - Z b)$
 - ♦ $\partial E / \partial a = 0 \Rightarrow x' = Z a$
 - Least square estimates
 - ♦ Solution expressed by generalized inverse
 - ♦ $a = Z^+ x' = (Z^T Z)^{-1} Z^T x'$
 - ♦ $b = Z^+ y'$
- Higher-order approximation
 - 2nd order polynomial usually suffices for many applications



Warping Based on Feature Lines

- Proposed by Beier-Neely (SIGGRAPH'92)
- Controlling warping through directed line pairs
 - Points on a line in I_0 is mapped to corresponding line in I_1
 - Other points are mapped by preserving certain relations w.r.t. the line
 - $u \in [0,1]$ (portion), v (#pixels)
- Transf. with multiple line pairs
 - Estimate warping coordinate for each line
 - Combine through weighed sum
 - closer lines impose more influence

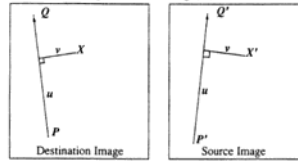


Figure 1: Single line pair

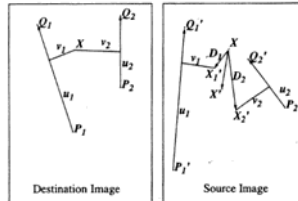


Figure 3: Multiple line pairs



Examples of Feature Line-Based Warping

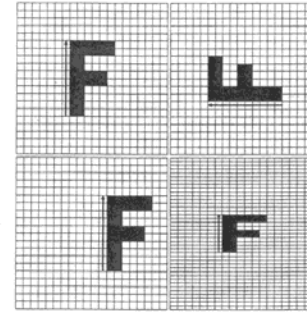


Figure 2: Single line pair examples

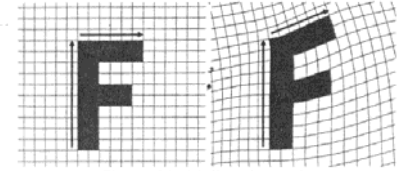


Figure 4: Multiple line pair example



Image Morphing



Image Morphing (Metamorphosis)



- Morph: gradually convert one image to another image
- Pixel-wise dissolve
 - $I_t = (1-t)I_0 + tI_1$
 - Simple but intermediate frames may be blurred and have ghost-effect
 - visual quality is not very good when played slowly
- Feature-points based (mesh model)
 - Require generate meshes and find correspondence
 - Require same amount of feature points in I_0 and I_1
 - Hard to morph between very different objects
- Feature-based warping plus dissolve (Beier-Neely '92)
 - Interpolate line pairs for intermediate frames
 - Warp both I_0 and I_1 toward the lines in I_t
 - Generate final intermediate frames by dissolving two warped images
 - Tradeoff between computation complexity and quality of visual effect



Summary

- **Geometrical manipulations**
 - Described by matrix multiplication in homogeneous coordinates
 - ◆ *R.S.T. of image object and image coordinates*
 - ◆ *Reflection and shear*
 - ◆ *General composite transforms*
 - Non-linear spatial warping
 - ◆ *Polynomial mapping*
 - ◆ *Compensating lens distortions through least square estimation*
 - Implementation issues: forward vs. reverse mapping
- **Image morphing**
 - Feature-line based warping + dissolve
- **Next time**
 - Image filtering and restoration



Assignment

- **Readings**
 - Chapt. 5 & 11 of “Computer Graphics” (Hearn-Baker, Prentice-Hall, 2nd Ed)
 - Image morphing paper by Beier-Neely (ACM SIGGRAPH'92)
- **Project BB-3 (Motion estimation/compensation)**
 - Due 11/8/2001 Thursday 1:59pm
- **Project proposal**
 - Due 11/13/2001 Tuesday 1:59pm
 - Submit electronically or a hard copy
- **Start working on project**
 - Don't wait till the last minutes

