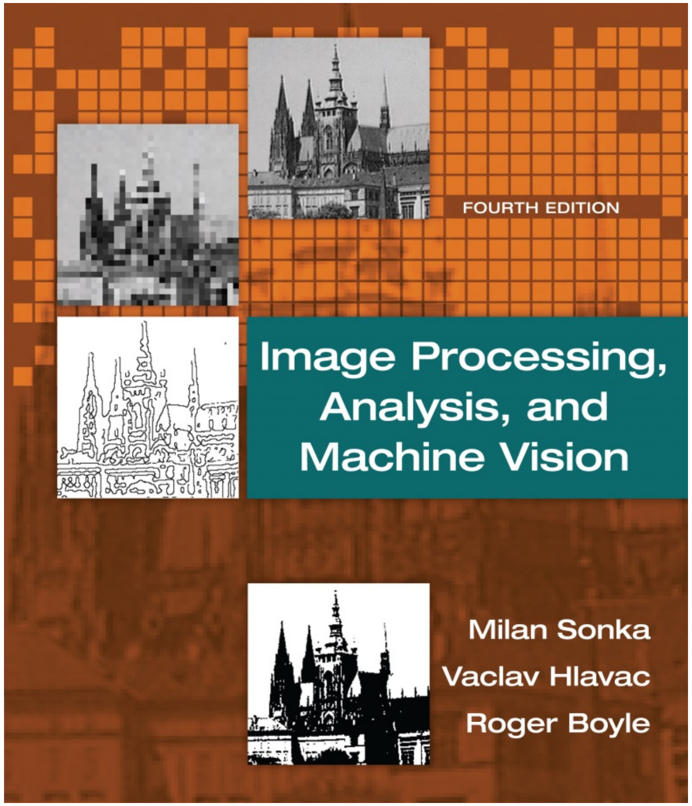


Chapter 5.2 Geometric Transformations

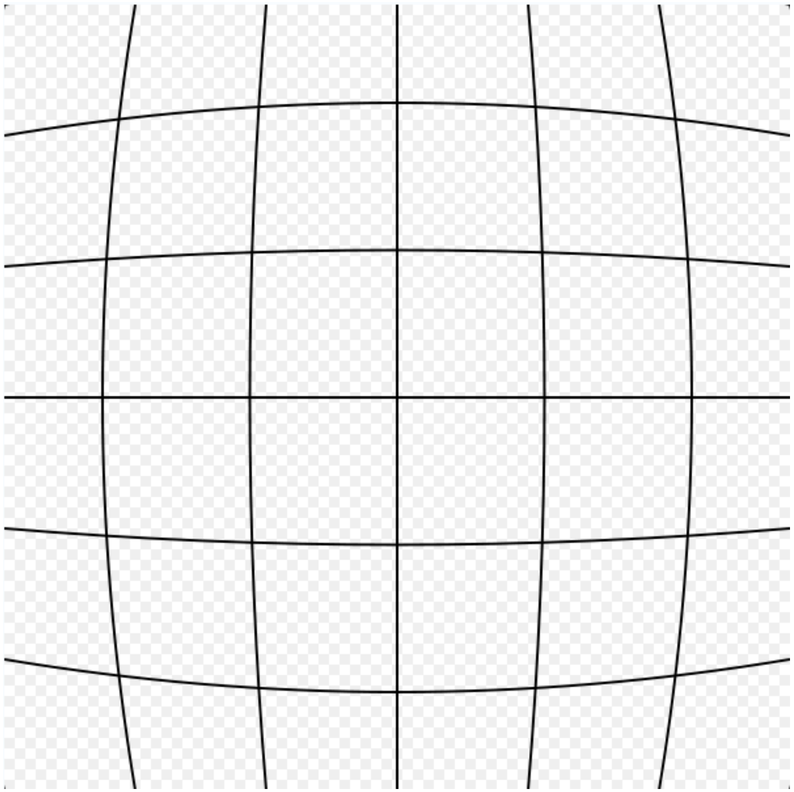
Sources:

- Sonka Textbook
- Gonzalez/Woods DIP textbook

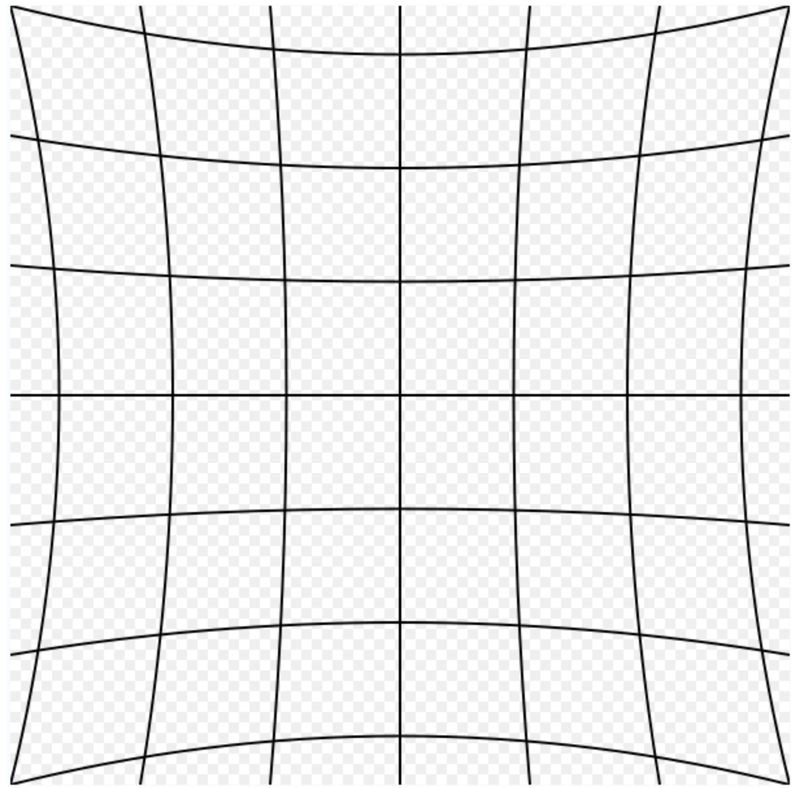
Geometric Transformations

- Greyscale transformations -> operate on range/output (e.g. via histograms)
- Geometric transformations -> operate on image domain
  - Coordinate transformations
  - Moving image content from one place to another
- Two parts:
  - Define transformation
  - Resample greyscale image in new coordinates

Geom Trans: Distortion From Optics

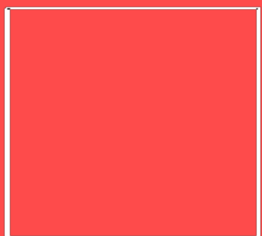


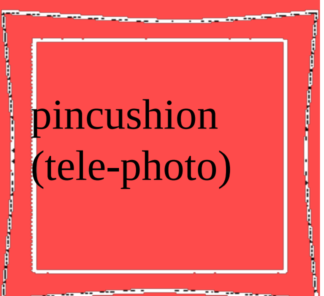
Barrel Distortion



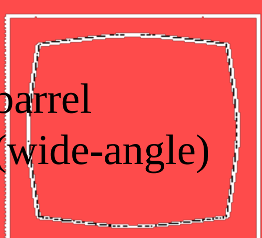
Pincushion Distortion

Radial Distortion



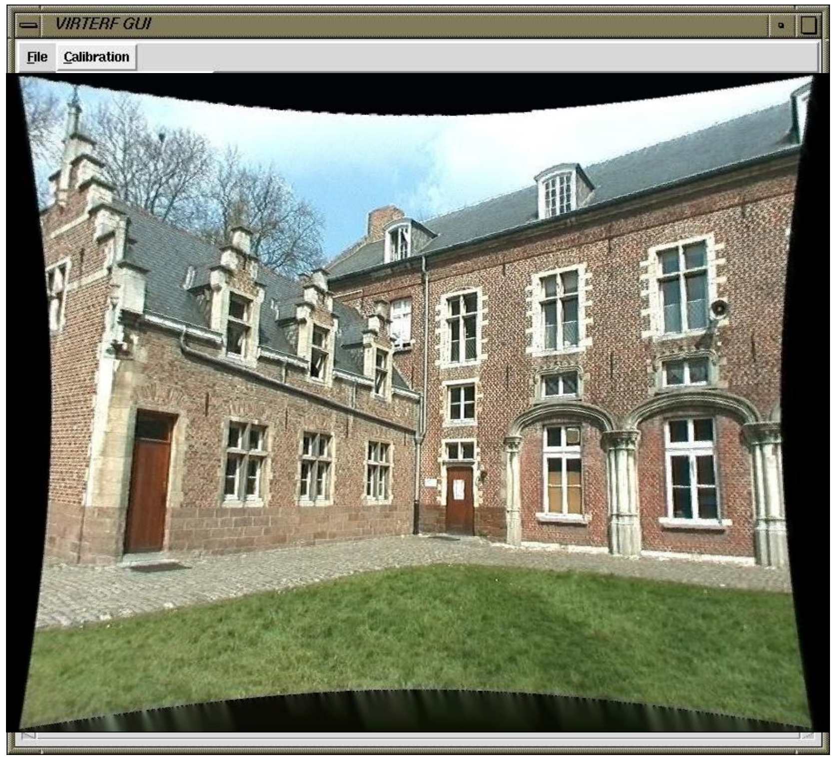


pincushion
(tele-photo)



barrel
(wide-angle)

magnification/focal length different
for different angles of inclination



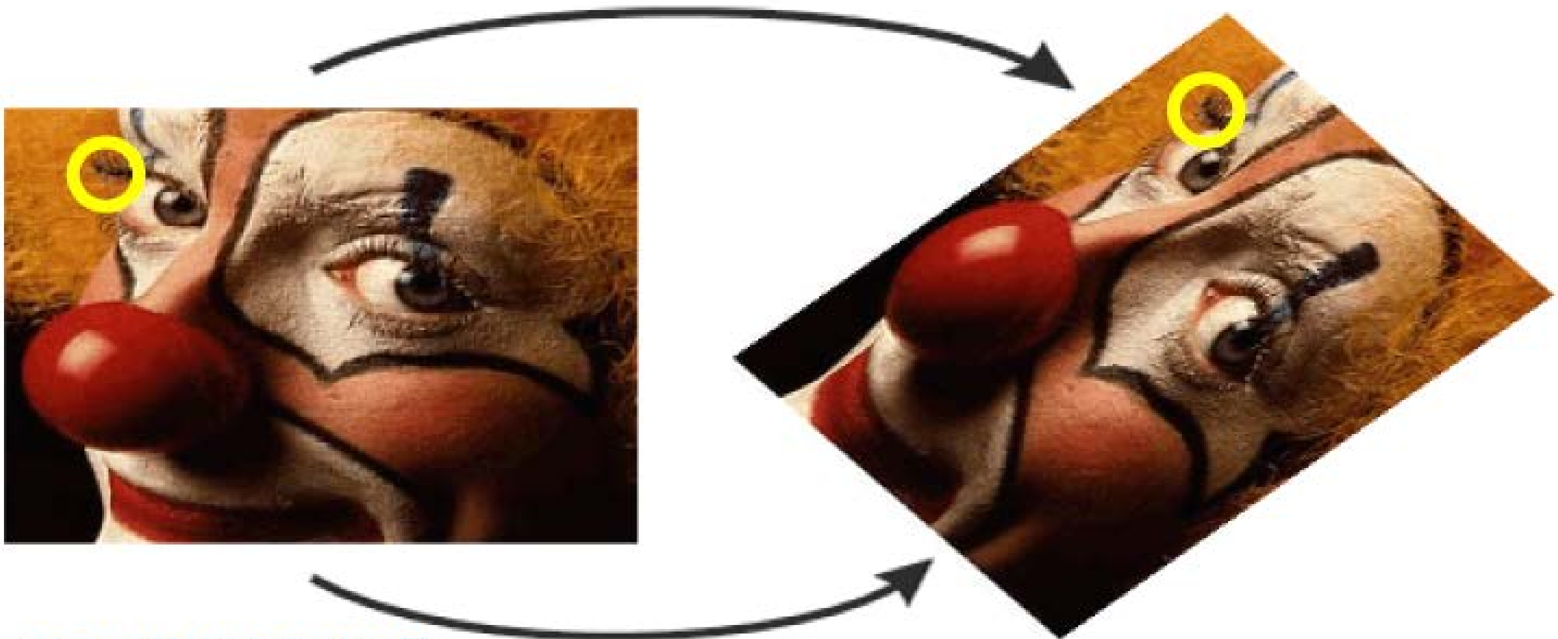
Can be corrected! (if parameters are know)

Geom Trans: Distortion From Optics

Geom. Trans.: Mosaicing

Saint-Guénolé Church of Batz-sur-Mer Equirectangular 360º by Vincent Montibus

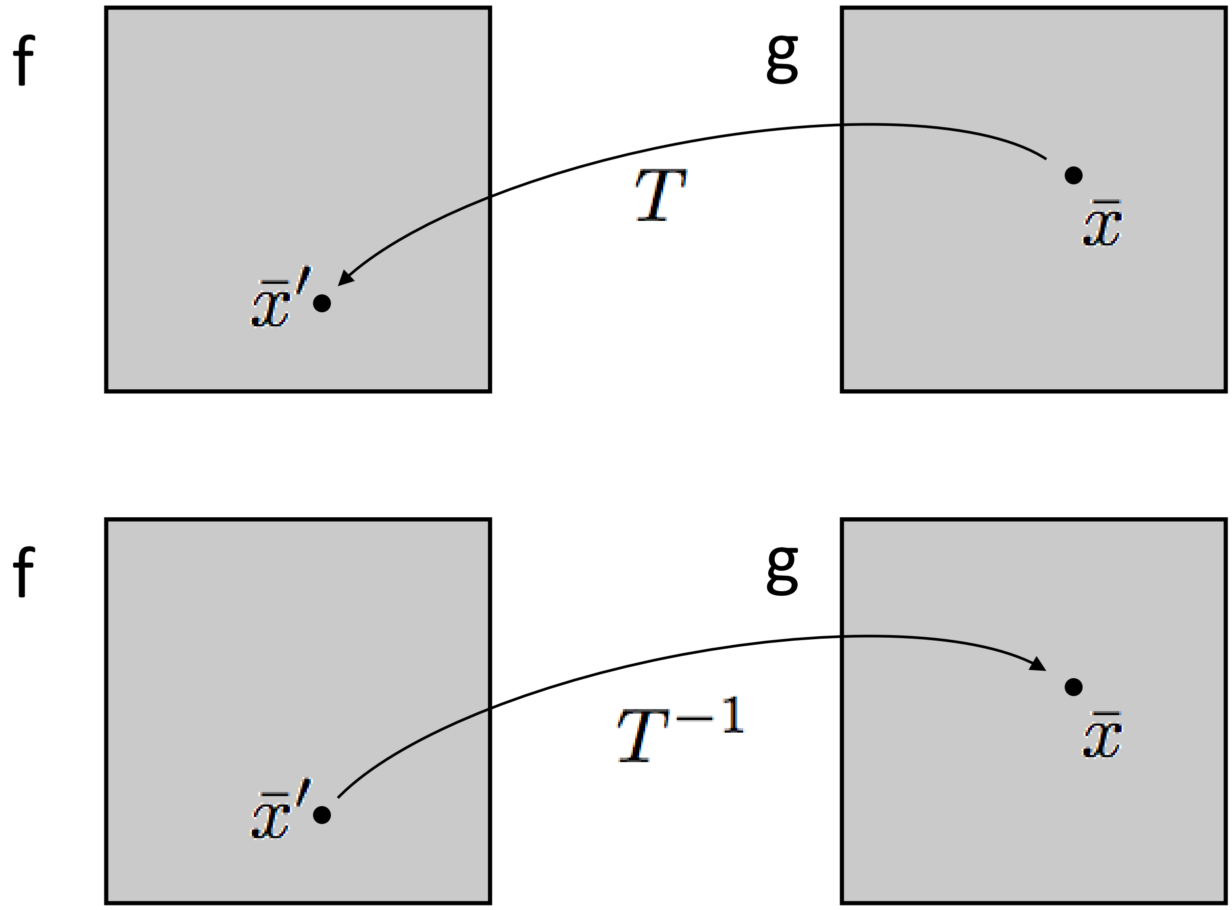
Domain Mappings Formulation



(E. H. W. Meijering)

g is the same (intensity) image as f, but sampled on these new coordinates

Domain Mappings



Domain Mappings Formulation

$$f \longrightarrow g$$

New image from old one

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = T(x,y) = \begin{pmatrix} T_1(x,y) \\ T_2(x,y) \end{pmatrix}$$

Coordinate transformation
Two parts – vector valued

$$g(x,y) = f(x',y')$$
$$g(x,y) = f(x',y') = \tilde{f}(x,y)$$

g is the same image as f, but sampled on these new coordinates

Domain Mappings Formulation

$$\bar{x}' = T(\bar{x})$$

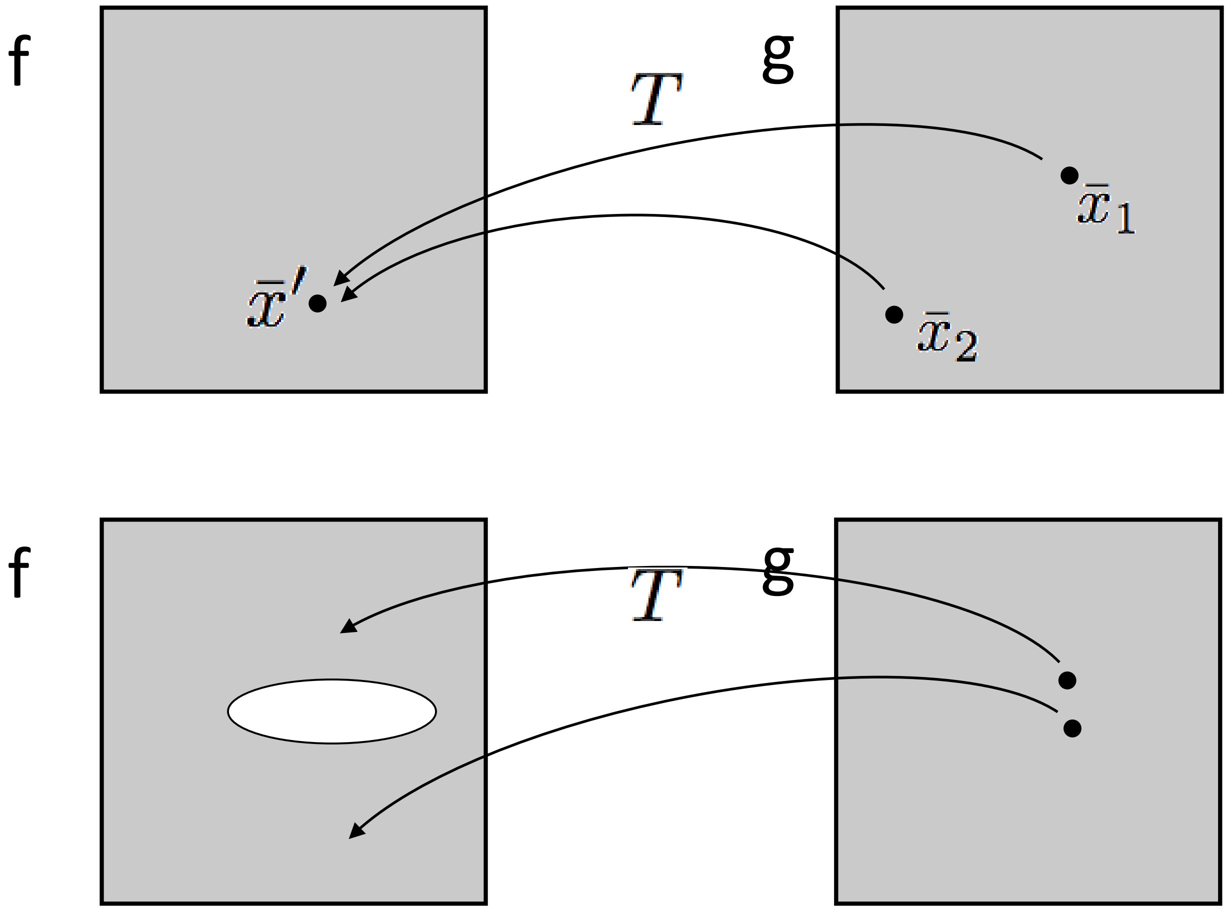
Vector notation is convenient.
Bar used some times, depends on context.

$$g(\bar{x}) = \tilde{f}(\bar{x}) = f(\bar{x}') = f(T(\bar{x}))$$

$$\bar{x} = T^{-1}(\bar{x}')$$

T may or may not have an inverse. If not, it means that information was lost.

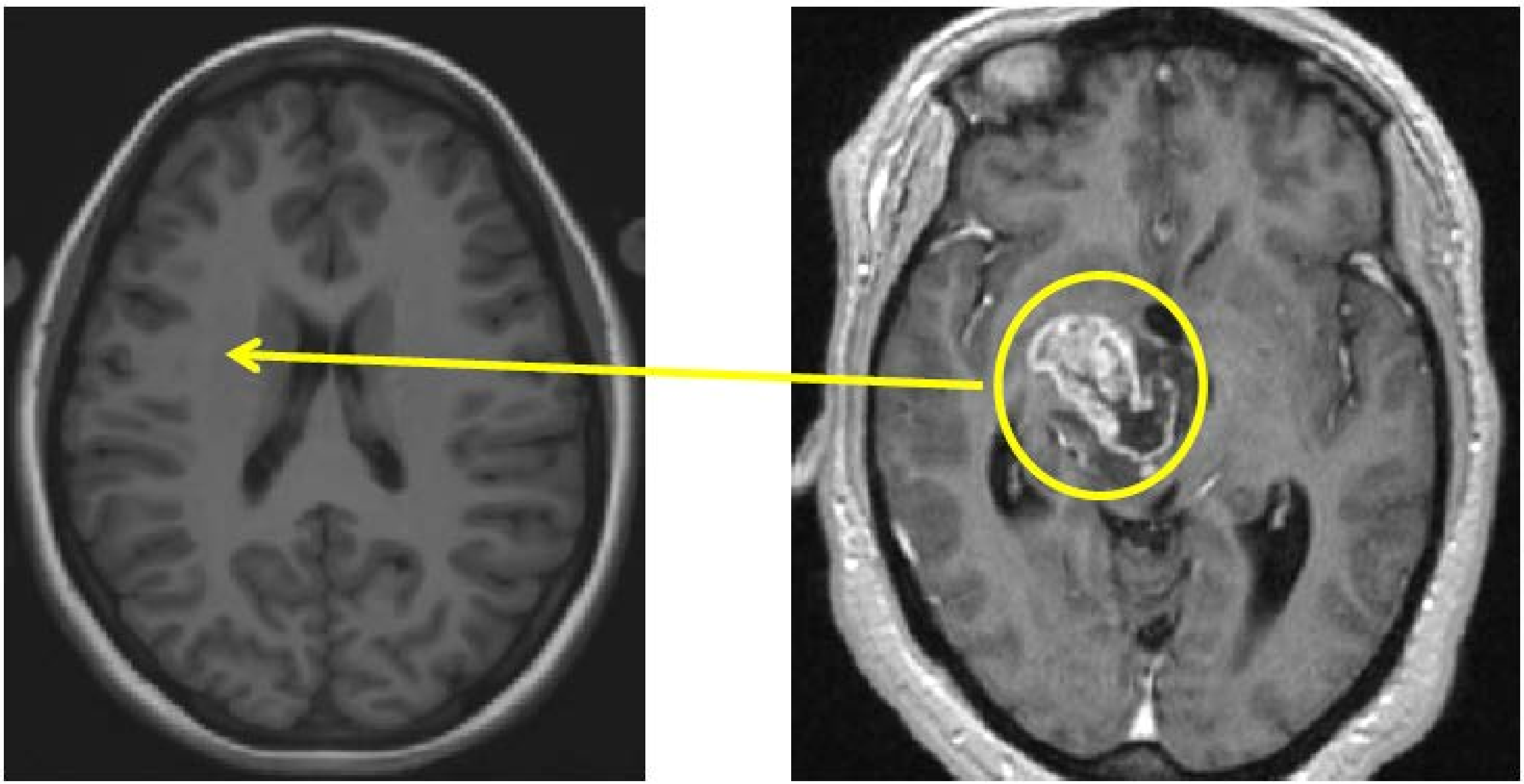
No Inverse?



Not "one to one"

Not "onto" - doesn't cover f

Example



Transformation Examples

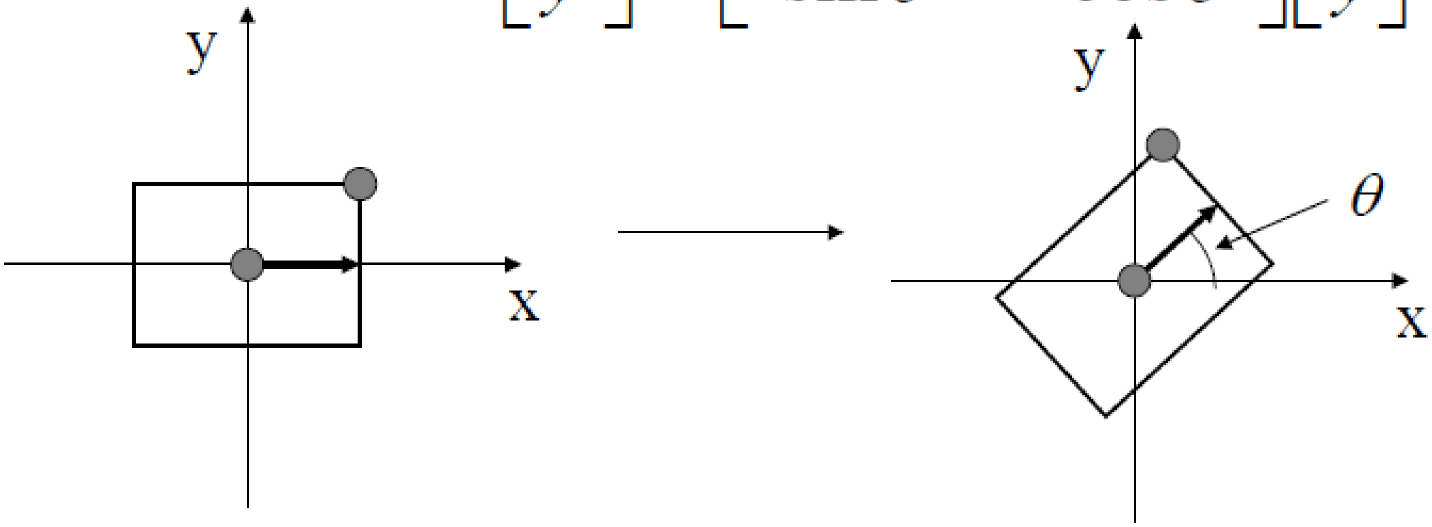
- Linear $\bar{x}' = A\bar{x} + \bar{x}_0$ $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$

$$x' = ax + by + x_0$$
$$y' = cx + dy + y_0$$

2D Rotation

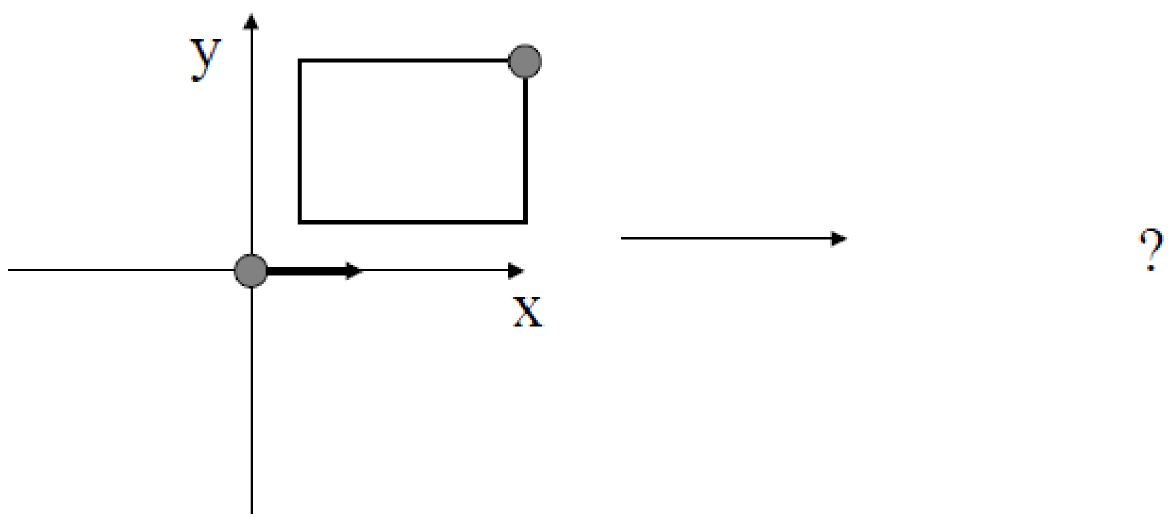
- Rotate counter-clockwise about the origin by an angle θ

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



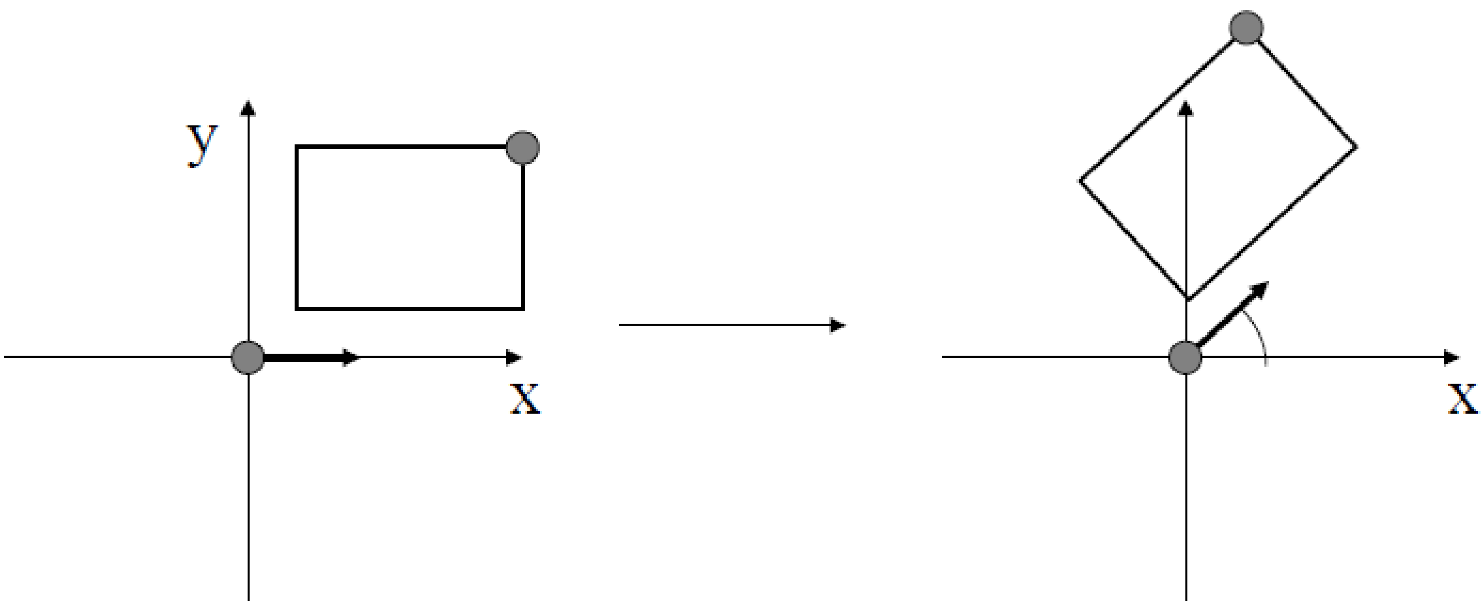
Rotating About An Arbitrary Point

- What happens when you apply a rotation transformation to an object that is not at the origin?



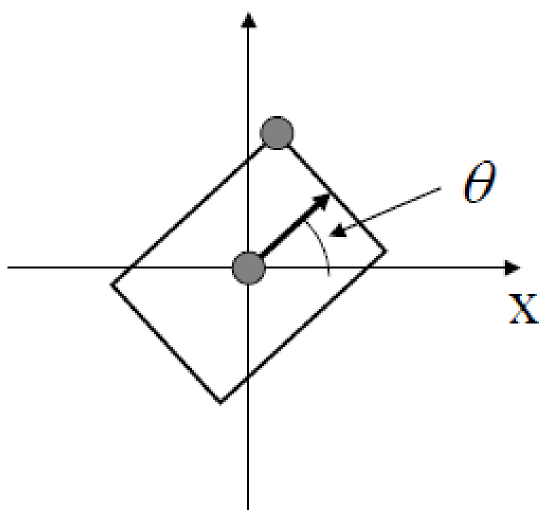
Rotating About An Arbitrary Point

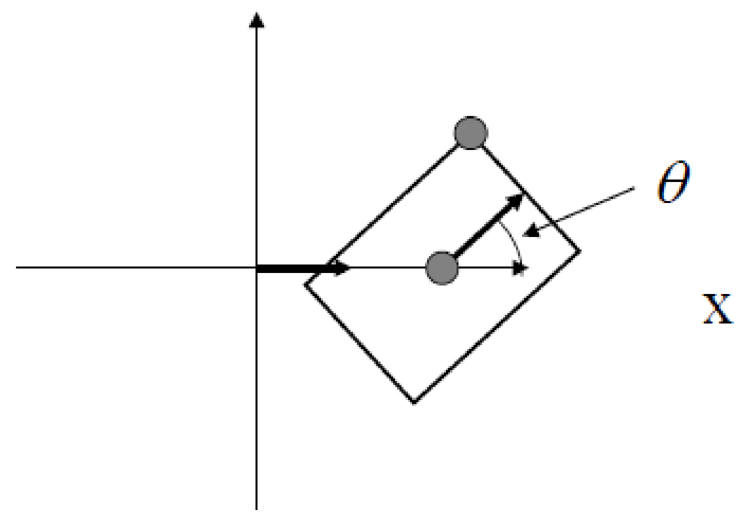
- What happens when you apply a rotation transformation to an object that is not at the origin?
  - It translates as well



Now: First Rotate, then Translate

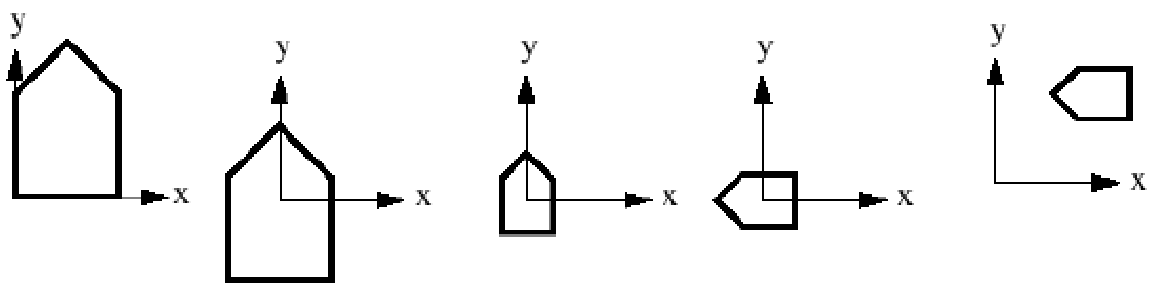
- Rotation followed by translation is **not the same** as translation followed by rotation:
- $T(R(\text{object})) \neq R(T(\text{object}))$





Series of Transformations

2D Object: Translate, scale, rotate, translate again



$$\overrightarrow{P'} = T2 + (R \cdot S \cdot (T1 + \vec{P}))$$

☛ Problem: Rotation, scaling, shearing are multiplicative transforms, but translation is additive.

Transformation Examples

- Linear $\bar{x}' = A\bar{x} + \bar{x}_0$ $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$

$$x' = ax + by + x_0$$
$$y' = cx + dy + y_0$$

Transformation Examples

- Linear $\bar{x}' = A\bar{x} + \bar{x}_0$ $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$

$$x' = ax + by + x_0$$
$$y' = cx + dy + y_0$$

- Trick: Add one dimension

$$\bar{x} = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

$$A = \begin{pmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \\ 0 & 0 & 1 \end{pmatrix}$$

Example: Translation

$$\bar{x}' = A\bar{x}$$

$$x' = x + x_0$$
$$y' = y + y_0$$
$$1 = 1$$

Transformation Examples

- Linear $\bar{x}' = A\bar{x} + \bar{x}_0$ $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$

$$x' = ax + by + x_0$$
$$y' = cx + dy + y_0$$

- Homogeneous coordinates

$$\bar{x} = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \quad A = \begin{pmatrix} a & b & x_0 \\ c & d & y_0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\bar{x}' = A\bar{x}$$

Homogeneous Coordinates

- Use three numbers to represent a point
- $(x,y)=(wx,wy,w)$ for any constant $w \neq 0$
  - Typically, (x,y) becomes $(x,y,1)$
  - To go backwards, divide by w
- Translation can now be done with matrix multiplication!

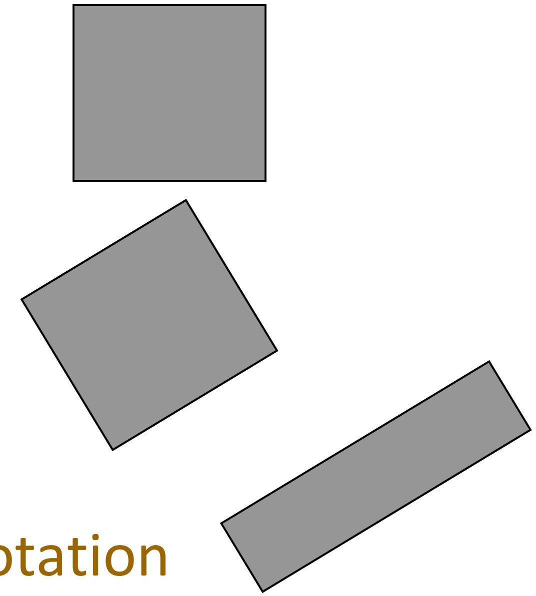
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_{xx} & a_{xy} & b_x \\ a_{yx} & a_{yy} & b_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Basic Transformations

- Translation: $\begin{bmatrix} 1 & 0 & b_x \\ 0 & 1 & b_y \\ 0 & 0 & 1 \end{bmatrix}$ Rotation: $\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$

- Scaling: $\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$

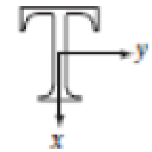
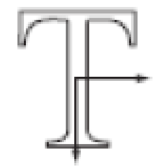
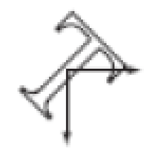
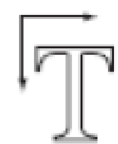
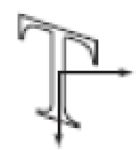
Special Cases of Linear



- Translation $$A = \begin{pmatrix} 0 & 0 & x_0 \\ 0 & 0 & y_0 \\ 0 & 0 & 1 \end{pmatrix}$$
- Rotation $$A = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$$
- Rigid = rotation + translation
  - Include forward and backward rotation for arbitrary axis
- Scaling $$A = \begin{pmatrix} p & 0 & 0 \\ 0 & q & 0 \\ 0 & 0 & 1 \end{pmatrix}$$ p, q < 1 : expand
- Skew

- Reflection

Linear Transformations

Transformation Name	Affine Matrix, T	Coordinate Equations	Example
Identity	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v$ $y = w$	
Scaling	$\begin{bmatrix} c_x & 0 & 0 \\ 0 & c_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = c_x v$ $y = c_y w$	
Rotation	$\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v\cos\theta - w\sin\theta$ $y = v\cos\theta + w\sin\theta$	
Translation	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{bmatrix}$	$x = v + t_x$ $y = w + t_y$	
Shear (vertical)	$\begin{bmatrix} 1 & 0 & 0 \\ s_v & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v + s_v w$ $y = w$	
Shear (horizontal)	$\begin{bmatrix} 1 & s_h & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v$ $y = s_h v + w$	

$$[x \;\; y \;\; 1] = [v \;\; w \;\; 1]\,\mathbf{T} = [v \;\; w \;\; 1]\begin{bmatrix} t_{11} & t_{12} & 0 \\ t_{21} & t_{22} & 0 \\ t_{31} & t_{32} & 1 \end{bmatrix}$$

Resulting Transformations

$$S = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad R = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \boxed{T = \begin{bmatrix} 1 & 0 & d_x \\ 0 & 1 & d_y \\ 0 & 0 & 1 \end{bmatrix}}$$

new: $\boxed{\overrightarrow{P'} = T2 \cdot R \cdot S \cdot T1 \cdot \vec{P}}$

before: $\overrightarrow{P'} = T2 + (R \cdot S \cdot (T1 + \vec{P}))$

Cascading of Transformations

Excellent Introduction Materials (MIT):
http://groups.csail.mit.edu/graphics/classes/6.837/F01/Lecture07/

Demo:
http://groups.csail.mit.edu/graphics/classes/6.837/F01/Lecture07/Slide09.html

Excellent Materials for self study

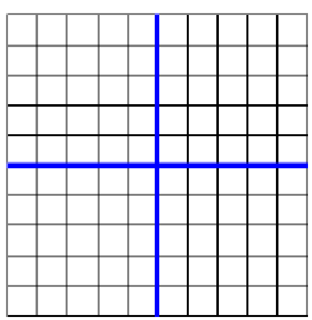
http://groups.csail.mit.edu/graphics/classes/6.837/F01/Lecture07/Slide01.html

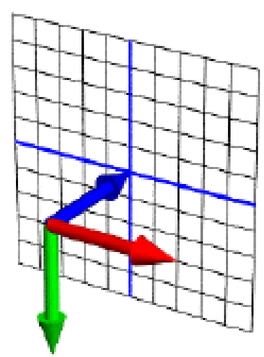
Problems with this Form

- Must consider Translation and Rotation separately
- Computing the inverse transform involves multiple steps
- Order matters between the R and T parts

$$R(T(\bar{x})) \neq T(R(\bar{x}))$$

These problem can be remedied by considering our 2 dimensional image plane as a 2D subspace within 3D.





Link: http://groups.csail.mit.edu/graphics/classes/6.837/F01/Lecture07/Slide01.html

Linear Transformations

- Also called "affine"
  - 6 parameters
- Rigid -> 3 parameters
- Invertibility $T^{-1}(\bar{x}) = A^{-1}\bar{x}$
  - Invert matrix
- What does it mean if A is not invertible?

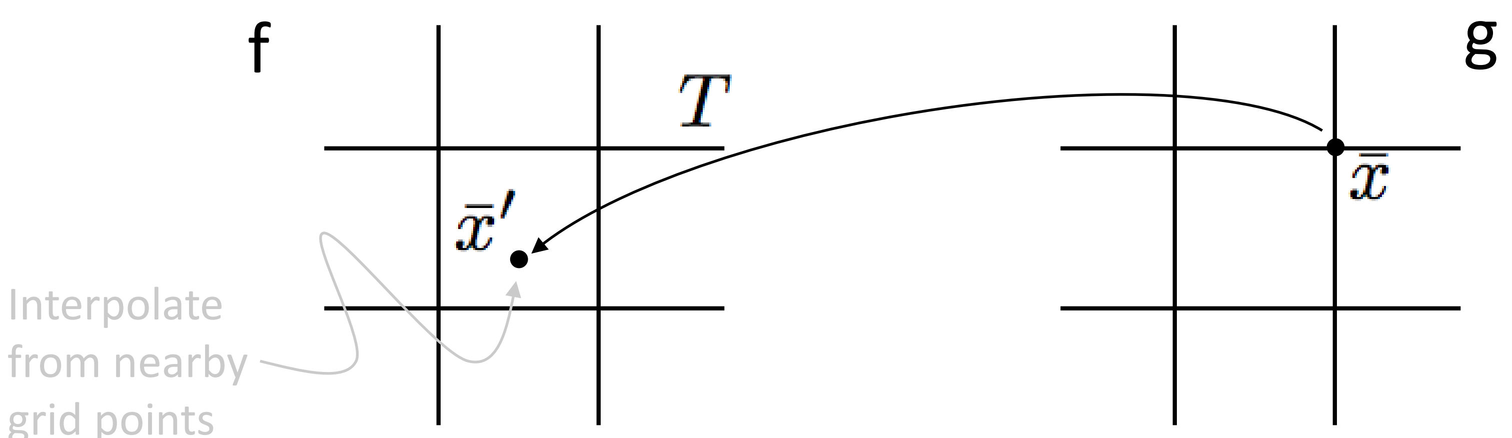
Implementation

Two major procedures:

1. Definition or estimation of transformation type and parameters
2. Application of transformation: Actual transformation of image

Implementation – Two Approaches

1. Pixel filling – backward mapping
   - T() takes you from coords in g() to coords in f()
   - Need random access to pixels in f()
   - Sample grid for g(), interpolate f() as needed



- Nearest-neighborhood interpolation
  - Assigned to point (x,y) brightness value of nearest point g in discrete raster

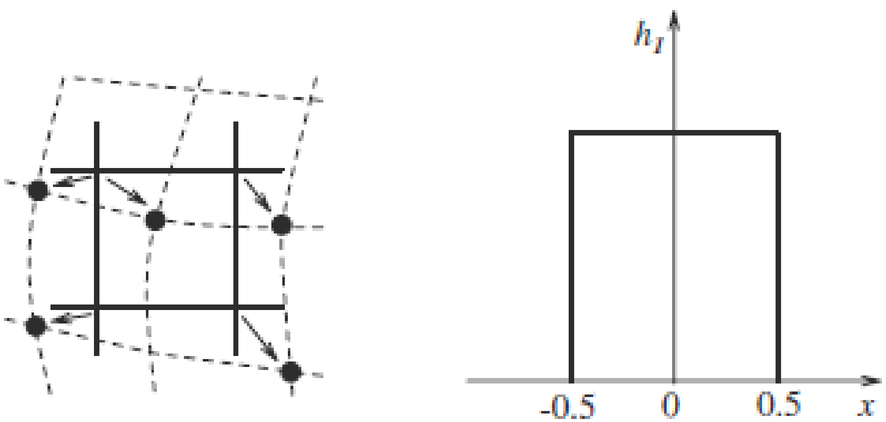
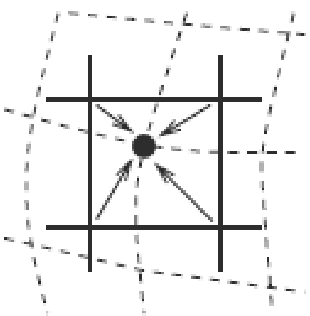


Figure 5.6: Nearest-neighborhood interpolation. The discrete raster of the original image is depicted by the solid line. © *Cengage Learning 2015.*

- Linear interpolation
  - Explores four points neighboring the point (x,y) and assumes that the brightness function is linear in this neighborhood



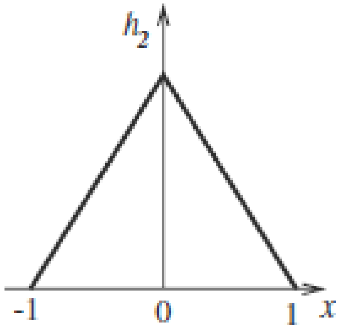
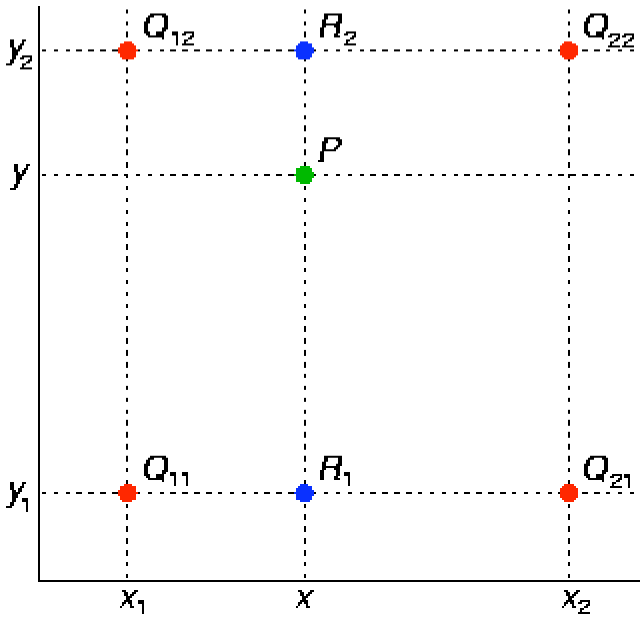


Figure 5.7: Linear interpolation. The discrete raster of the original image is depicted by the solid line. *© Cengage Learning 2015.*

Interpolation: Binlinear

- Successive application of linear interpolation along each axis



$$f(R_1) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21})$$

$$f(R_2) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22})$$

$$f(P) \approx \frac{y_2 - y}{y_2 - y_1} f(R_1) + \frac{y - y_1}{y_2 - y_1} f(R_2).$$

Source: WIkipedia

Binlinear Interpolation

- *Not* linear in x, y

$$
\begin{aligned}
f(x,y) \approx\ & \frac{f(Q_{11})}{(x_2-x_1)(y_2-y_1)}(x_2-x)(y_2-y) \\
&+ \frac{f(Q_{21})}{(x_2-x_1)(y_2-y_1)}(x-x_1)(y_2-y) \\
&+ \frac{f(Q_{12})}{(x_2-x_1)(y_2-y_1)}(x_2-x)(y-y_1) \\
&+ \frac{f(Q_{22})}{(x_2-x_1)(y_2-y_1)}(x-x_1)(y-y_1).
\end{aligned}
$$

$$b_1 + b_2 x + b_3 y + b_4 xy$$

$$
\begin{aligned}
b_1 &= f(0,0) \\
b_2 &= f(1,0) - f(0,0) \\
b_3 &= f(0,1) - f(0,0) \\
b_4 &= f(0,0) - f(1,0) \\
&\quad - f(0,1) + f(1,1).
\end{aligned}
$$

Binlinear Interpolation

- Convenient form
  - Normalize to unit grid [0,1]x[0,1]

$$f(x,y) \approx f(0,0)\,(1-x)(1-y) + f(1,0)\,x(1-y) + f(0,1)\,(1-x)y + f(1,1)xy.$$

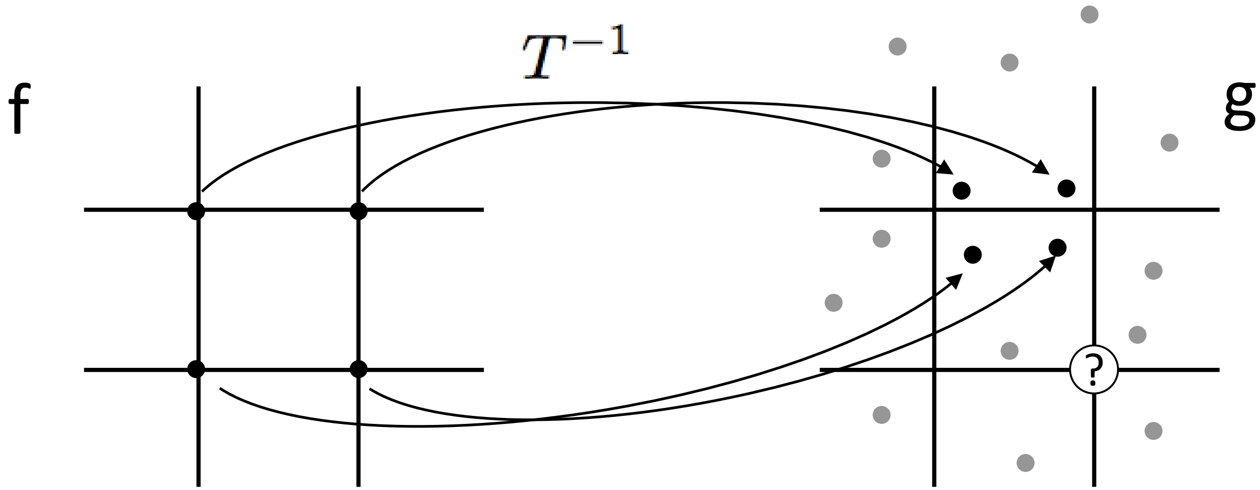
$$f(x,y) \approx \begin{bmatrix} 1-x & x \end{bmatrix} \begin{bmatrix} f(0,0) & f(0,1) \\ f(1,0) & f(1,1) \end{bmatrix} \begin{bmatrix} 1-y \\ y \end{bmatrix}.$$

- Bilinear is NONLINEAR in x and y !

Implementation – Two Approaches

2. Splatting – forward mapping

- T^{-1}() takes you from coords in f() to coords in g()
- You have f() on grid, but you need g() on grid
- Push grid samples onto g() grid and do interpolation from unorganized data (kernel)



Nearby points are not organized – "scattered"

Scattered Data Interpolation With Kernels Shepard's method

- Define kernel
  - Falls off with distance, radially symmetric

$$K(\bar{x}_1, \bar{x}_2) = K(|\bar{x}_1 - \bar{x}_2|)$$

Kernel examples

$$K(\bar{x}_1, \bar{x}_2) = \frac{1}{2\pi\sigma^2} e^{\frac{|\bar{x}_1 - \bar{x}_2|^2}{2\sigma^2}}$$

$$K(\bar{x}_1, \bar{x}_2) = \frac{1}{|\bar{x}_1 - \bar{x}_2|^p}$$

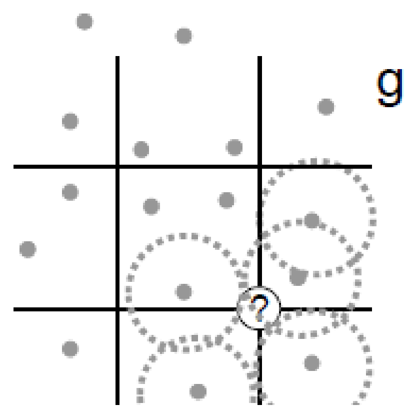
$$g(x) = \frac{1}{\sum_{j=1}^{N} w_j} \sum_{i=1}^{N} w_i f\left(x'_i\right)$$

$$w_j = K\left(|\bar{x} - T^{-1}(\bar{x}'_j)\right)$$

Required grid coordinates in g

Transformed coord. from f

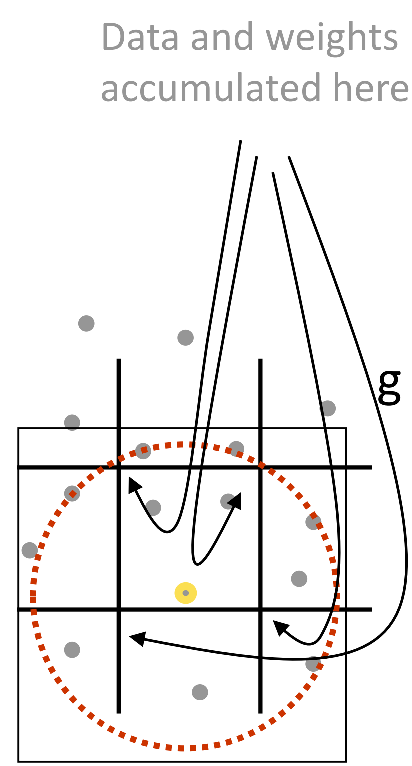
Grid coordinates in f



Shepard's Method Implementation

- If points are dense enough
  - Truncate kernel
  - For each point in f()
    - Form a small box around it in g() – beyond which truncate
    - Put weights and data onto grid in g()
  - Accumulate contributions at grid g()
  - Divide total data by total weights: B/A

$$A = \sum_{j=1}^{N} w_j \qquad B = \sum_{i=1}^{N} w_i f\left(T^{-1}(x_i')\right)$$



- Bi-cubic interpolation
  - Improves the model of the brightness function by approximating it locally by a bi-cubic polynomial surface
  - 16 neighboring points are used for interpolation
  - Does not suffer from step-like boundary problem of nearest-neighborhood interpolation
  - Often used in raster displays that enable zooming with respect to an arbitrary point

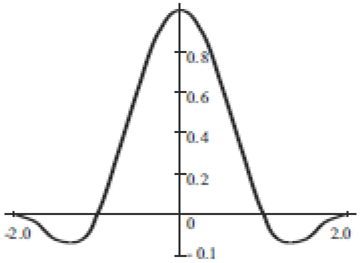
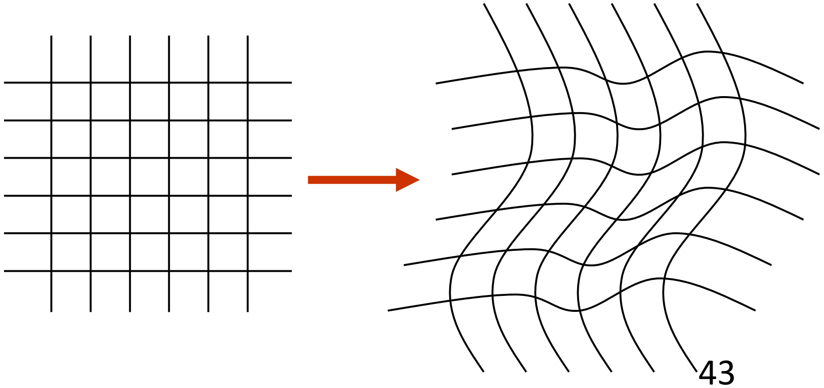


Figure 5.8: Bi-cubic interpolation kernel.
© Cengage Learning 2015.

ESTIMATION OF TRANSFORMATIONS

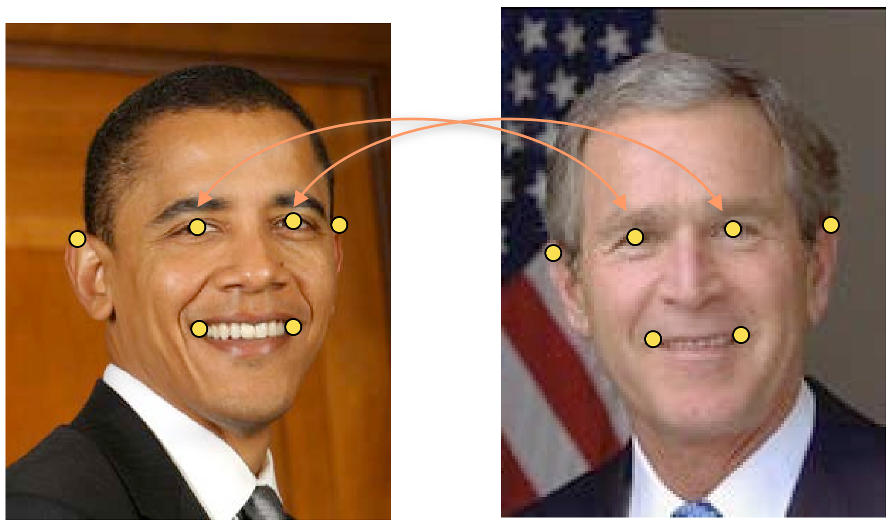
Determine Transformations

- All polynomials of (x,y)
- Any vector valued function with 2 inputs
- How to construct transformations?
  - Define form or class of a transformation
  - Choose parameters within that class
    - Rigid - 3 parameters (T,R)
    - Affine - 6 parameters



Correspondences

- Also called “landmarks” or “fiducials”



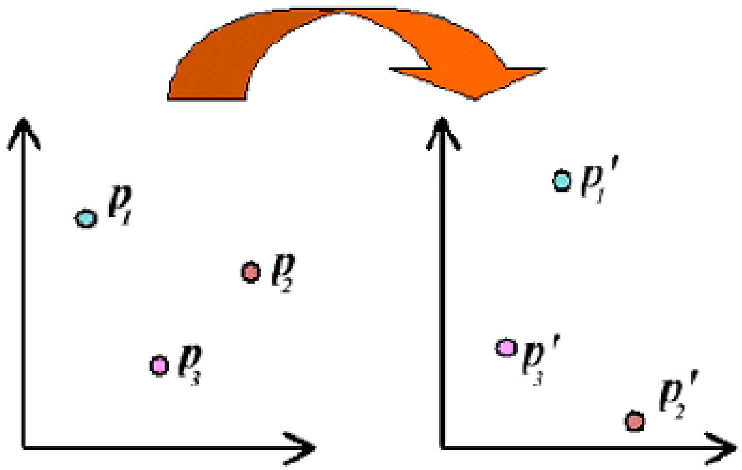
$$\bar{c}_1, \bar{c}'_1$$
$$\bar{c}_2, \bar{c}'_2$$
$$\bar{c}_3, \bar{c}'_3$$
$$\bar{c}_4, \bar{c}'_4$$
$$\bar{c}_5, \bar{c}'_5$$
$$\bar{c}_6, \bar{c}'_6$$

Question: How many landmarks for affine T?

Question: How many landmarks for affine T?

- Estimation of 6 parameters → 3 corresponding point pairs with (x,y) coordinates

The coordinates of three corresponding points uniquely determine and Affine Transform



If we know where we would like at least three points to map to, we can solve for an Affine transform that will give this mapping.

Transformations/Control Points Strategy

1. Define a functional representation for T with k parameters (B) $T(\beta, \bar{x})$ $\beta = (\beta_1, \beta_2, \ldots, \beta_K)$
2. Define (pick) N correspondences
3. Find B so that
$$\bar{c}_i' = T(\beta, \bar{c}_i) \;\; i = 1, \ldots, N$$
4. If overconstrained (K < 2N) then solve
$$\arg\min_{\beta} \left[\sum_{i=1}^{N} (\bar{c}_i' - T(\beta, \bar{c}_i))^2 \right]$$

Example Affine Transformation: 3 Corresponding Landmarks

Solution Method

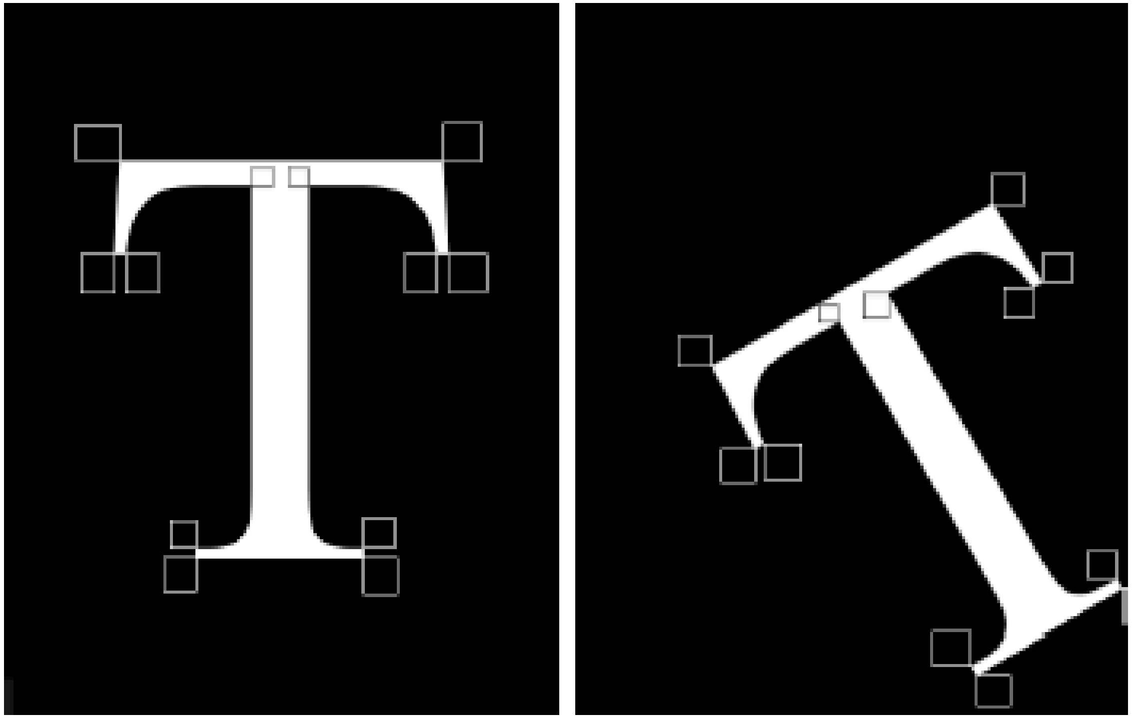
We've used this technique several times now. We set up 6 linear equations in terms of our 6 unknown values. In this case, we know the coordinates before and after the mapping, and we wish to solve for the entries in our Affine transform matrix.

This gives the following solution:

$$\mathbf{X}^{-1}\mathbf{x}' = \mathbf{a}$$

$$\underbrace{\begin{bmatrix} x_1' \\ y_1' \\ x_2' \\ y_2' \\ x_3' \\ y_3' \end{bmatrix}}_{\mathbf{x}'} = \underbrace{\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_2 & y_2 & 1 \\ x_3 & y_3 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_3 & y_3 & 1 \end{bmatrix}}_{\mathbf{X}} \underbrace{\begin{bmatrix} a_{11} \\ a_{12} \\ a_{13} \\ a_{21} \\ a_{22} \\ a_{23} \end{bmatrix}}_{\mathbf{a}}$$

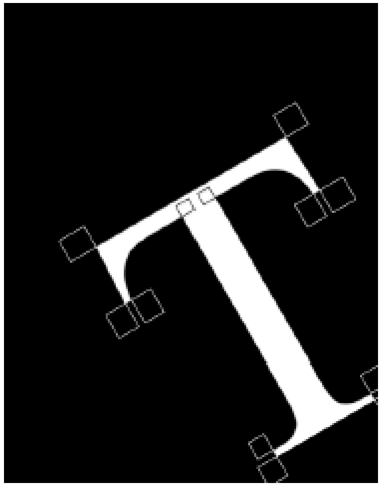
Example



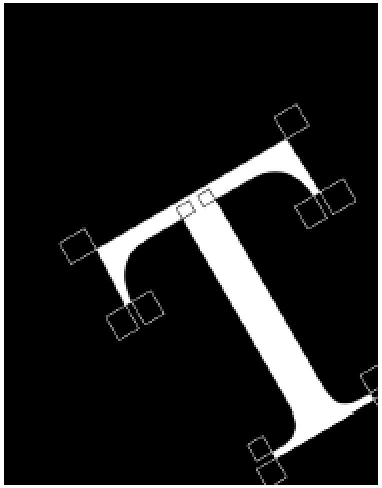
Left: source_letter_T.tif
Right: target_letter_T.tif

Example ctd.

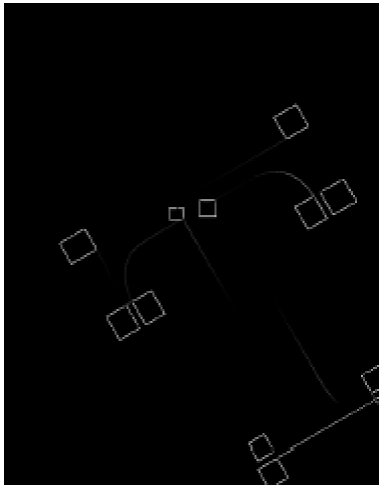
When choose all the marked points of the letter T image, I get the result:



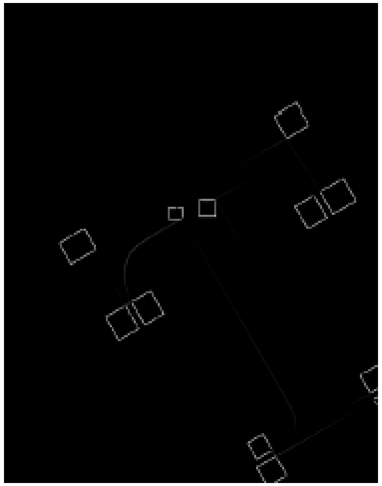
after_landmarks_affine_transform_3_points_source_letter_T.tif



after_landmarks_affine_transform_12_points_source_letter_T.tif



difference_between_source_and_target_3_points_source_letter_T.tif



difference_between_source_and_target_12_points_source_letter_T.tif

Example: Quadratic

Transformation

$$T_x = \beta_x^{00} + \beta_x^{10} x + \beta_x^{01} y + \beta_x^{11} xy + \beta_x^{20} x^2 + \beta_x^{02} y^2$$
$$T_y = \beta_y^{00} + \beta_y^{10} x + \beta_y^{01} y + \beta_y^{11} xy + \beta_y^{20} x^2 + \beta_y^{02} y^2$$

Denote $\bar{c}_i = (c_{x,i}, c_{y,i})$

Correspondences must match

$$c'_{y,i} = \beta_y^{00} + \beta_y^{10} c_{x,i} + \beta_y^{01} c_{y,i} + \beta_y^{11} c_{x,i} c_{y,i} + \beta_y^{20} c_{x,i}^2 + \beta_y^{02} c_{y,i}^2$$
$$c'_{x,i} = \beta_x^{00} + \beta_x^{10} c_{x,i} + \beta_x^{01} c_{y,i} + \beta_x^{11} c_{x,i} c_{y,i} + \beta_x^{20} c_{x,i}^2 + \beta_x^{02} c_{y,i}^2$$

Note: these equations are linear in the unkowns

Write As Linear System

$$
\begin{pmatrix}
\begin{matrix}
1 & c_{x,1} & c_{y,1} & c_{x,1}c_{y,1} & c_{x,1}^2 & c_{y,1}^2 \\
1 & c_{x,2} & c_{y,2} & c_{x,2}c_{y,2} & c_{x,2}^2 & c_{y,2}^2 \\
 & & & \vdots & & \\
1 & c_{x,N} & c_{y,N} & c_{x,N}c_{y,N} & c_{x,N}^2 & c_{y,N}^2
\end{matrix} & 0 \\
0 & \begin{matrix}
1 & c_{x,1} & c_{y,1} & c_{x,1}c_{y,1} & c_{x,1}^2 & c_{y,1}^2 \\
1 & c_{x,2} & c_{y,2} & c_{x,2}c_{y,2} & c_{x,2}^2 & c_{y,2}^2 \\
 & & & \vdots & & \\
1 & c_{x,N} & c_{y,N} & c_{x,N}c_{y,N} & c_{x,N}^2 & c_{y,N}^2
\end{matrix}
\end{pmatrix}
\begin{pmatrix}
\beta_x^{00} \\ \beta_x^{10} \\ \beta_x^{01} \\ \beta_x^{11} \\ \beta_x^{20} \\ \beta_x^{02} \\ \beta_y^{00} \\ \beta_y^{10} \\ \beta_y^{01} \\ \beta_y^{11} \\ \beta_y^{20} \\ \beta_y^{02}
\end{pmatrix}
=
\begin{pmatrix}
c'_{x,1} \\ c'_{x,2} \\ \vdots \\ c'_{x,N} \\ c'_{y,1} \\ c'_{y,2} \\ \vdots \\ c'_{y,N}
\end{pmatrix}
$$

$$Ax = b$$

A – matrix that depends on the (unprimed) correspondences and the transformation

x – unknown parameters of the transformation

b – the primed correspondences

Linear Algebra Background

$$Ax = b$$

$$\begin{aligned}
a_{11}x_1 + \ldots + a_{1N}x_N &= b_1 \\
a_{21}x_1 + \ldots + a_{2N}x_N &= b_2 \\
\ldots \qquad & \quad \ldots \\
a_{M1}x_1 + \ldots + a_{MN}x_N &= b_M
\end{aligned}$$

Simple case: A is square (M=N) and invertible (det[A] not zero)

$$A^{-1}Ax = Ix = x = A^{-1}b$$

Numerics: Don't find A inverse. Use Gaussian elimination or some kind of decomposition of A

Solving Least Squares Systems

- Psuedoinverse (normal equations)

$$A^T A x = A^T b$$
$$x = (A^T A)^{-1} A^T b$$

  – Issue: often not well conditioned (nearly singular)

- Alternative: *singular value decomposition SVD*

Singular Value Decomposition

$$\begin{pmatrix} & & \\ & A & \\ & & \end{pmatrix} = UWV^T = \begin{pmatrix} & & \\ & U & \\ & & \end{pmatrix} \begin{pmatrix} w_1 & & & 0 \\ & w_2 & & \\ & & \dots & \\ & & \dots & \\ 0 & & & w_N \end{pmatrix} \begin{pmatrix} & V^T & \end{pmatrix}$$

$$I = U^T U = U U^T = V^T V = V V^T$$

Invert matrix A with SVD

$$A^{-1} = V W^{-1} U^T \qquad W^{-1} = \begin{pmatrix} \frac{1}{w_1} & & & 0 \\ & \frac{1}{w_2} & & \\ & & \dots & \\ & & \dots & \\ 0 & & & \frac{1}{w_N} \end{pmatrix}$$

SVD for Singular Systems

- If a system is singular, some of the w's will be zero

$$x = VW^*U^Tb$$

$$w_j^* = \begin{cases} 1/w_j & |w_j| > \epsilon \\ 0 & \text{otherwise} \end{cases}$$

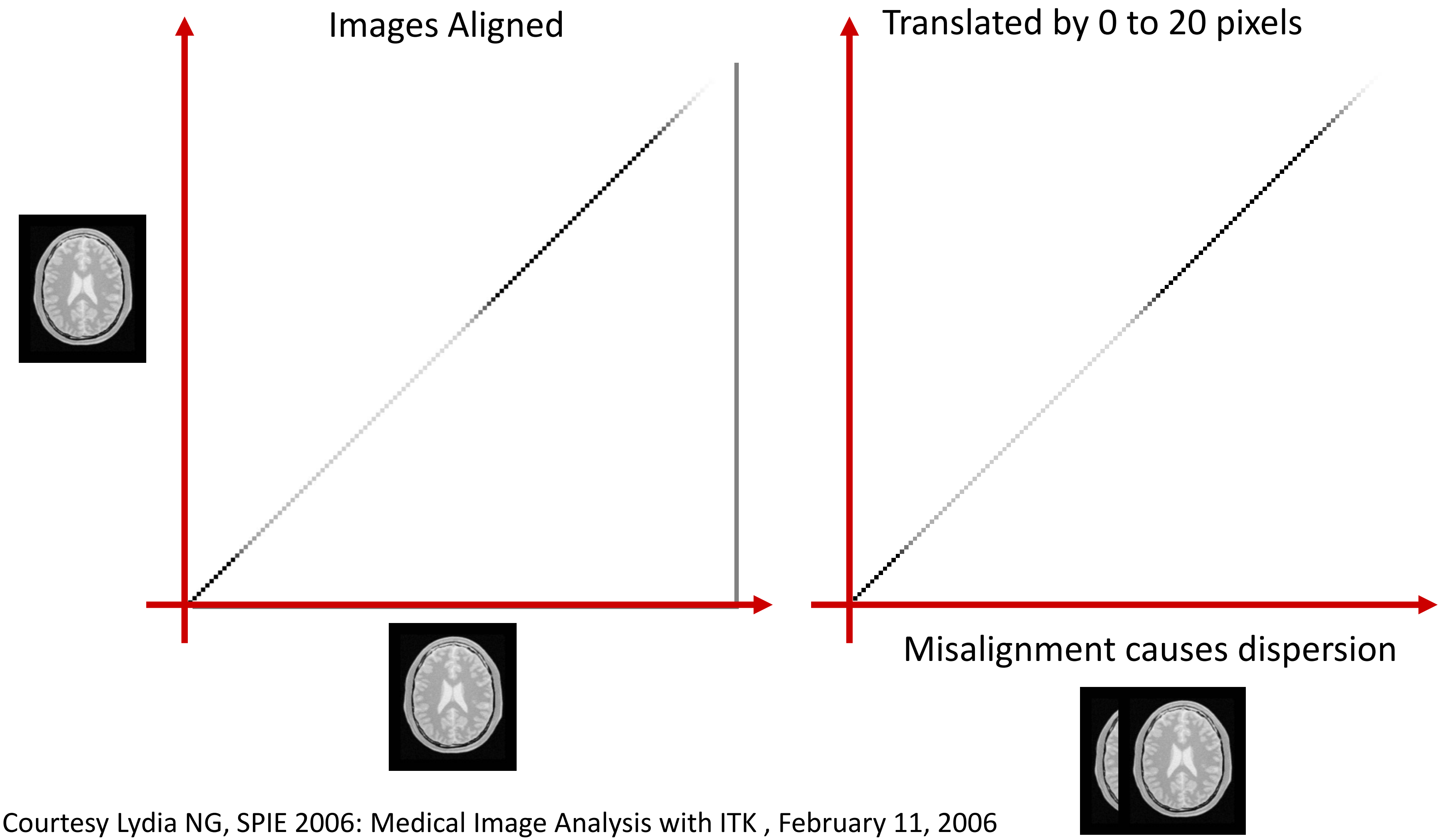
- Properties:
  - Underconstrained: solution with shortest overall length
  - Overconstrained: least squares solution

Landmark-free Image Registration

- Use “image match” function between source and transformed target image to calculate transformation parameters.
- Common: SSD between target and transformed source images:

$$\underset{\beta}{argmin}[\sum_{i=1}^{pixels}(I'(x_i) - T(\beta, I(x_i))^2$$

Concept via Joint Histograms: Intensity similarity btw transformed images

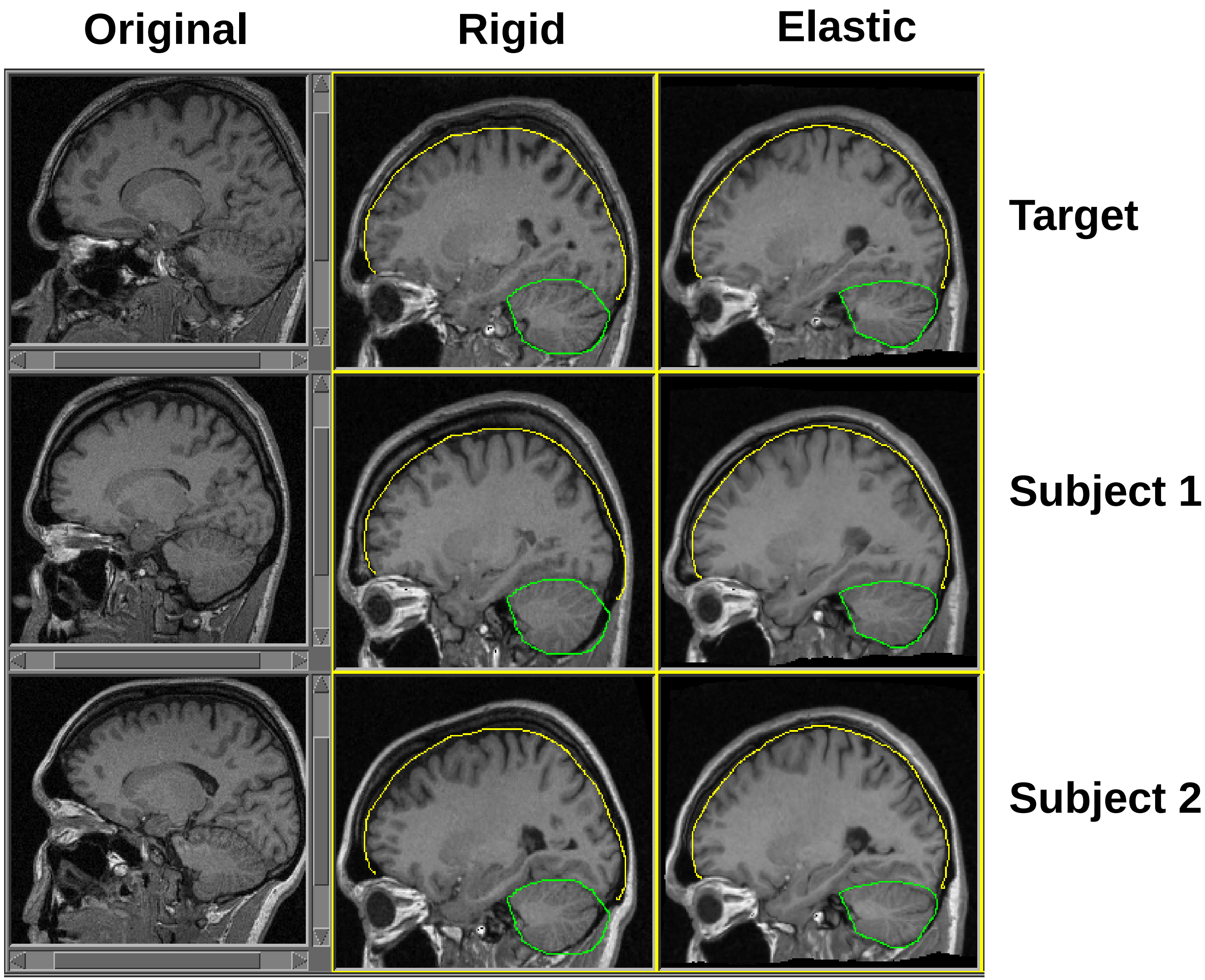


Courtesy Lydia NG, SPIE 2006: Medical Image Analysis with ITK , February 11, 2006

Choices: Linear/Nonlinear

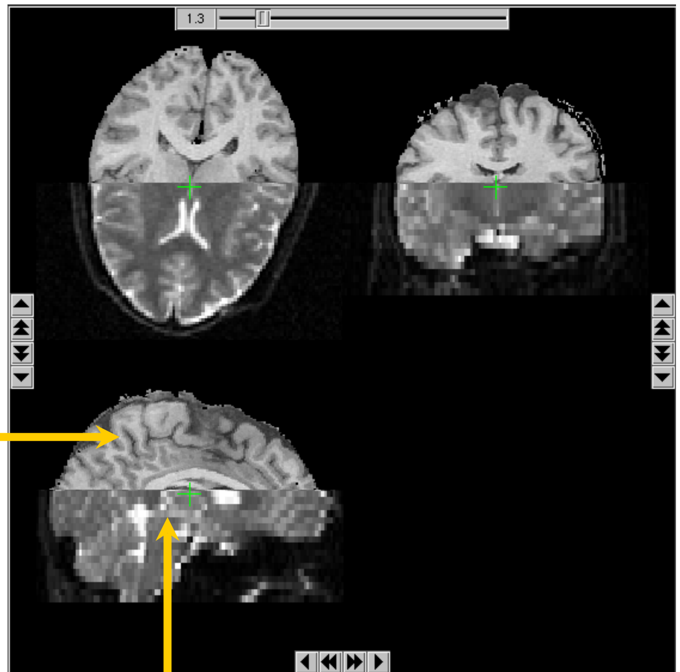
rigid: ‘Mirit’
(F.Maes)

elastic: ‘Demons’
(J.P. Thirion)

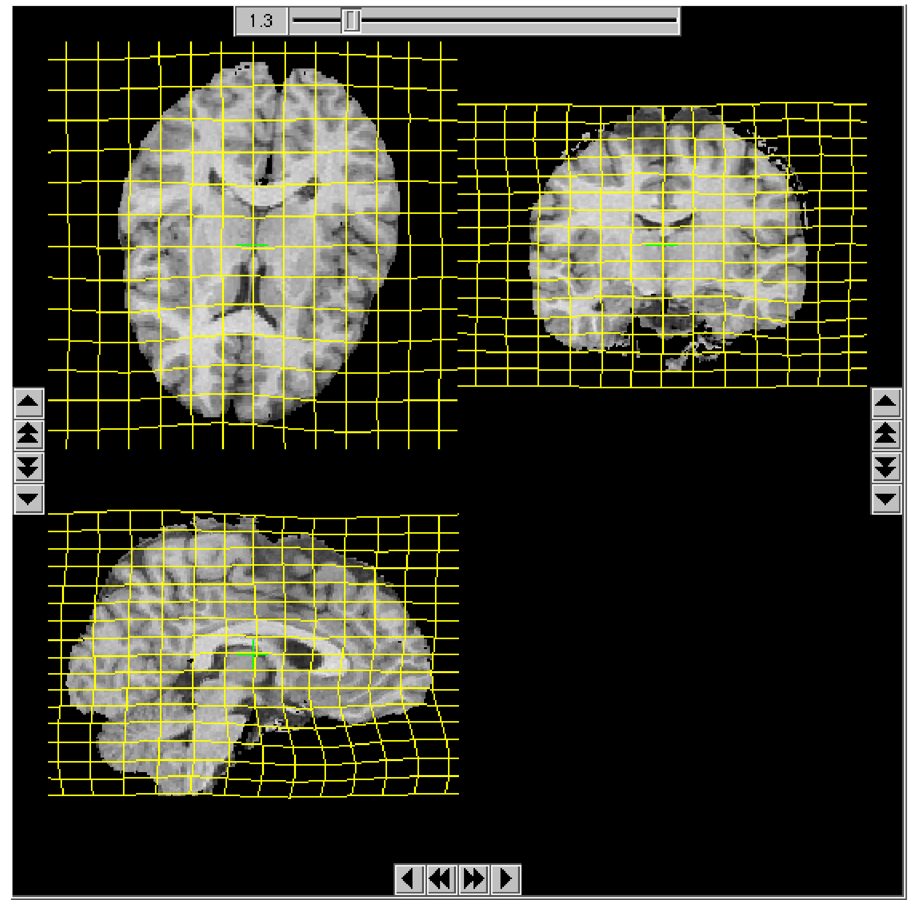


Example Nonlinear B-Spline warping

Target image (T1w MRI)



Source image (Baseline DTI image)



IRTK Software (Image Registration Toolkit, Daniel Rueckert, Imperial College: http://www.doc.ic.ac.uk/~dr/software/