Computer Vision Fundamentals
Lecture 3

Transformations

Transformations

- 2D-to-2D (image-to-image)
- 3D-to-3D (world-to-world)
- 3D-to-2D (camera model)
- 2D-to-3D (shape from X)
  - Shape from Stereo
  - Shape from Shading
  - Shape from Texture
  - Structure from Motion

Points

- World points are 3-Dimensional

  $\mathbf{P} = [X, Y, Z]^T$

  Or in homogeneous coordinates

  $\mathbf{P} = [hX, hY, hZ, h]^T$

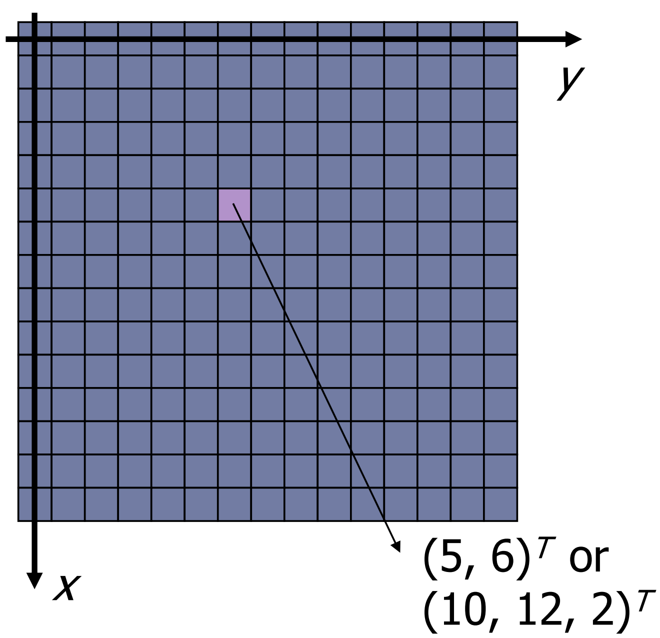
- Image points are 2-Dimensional

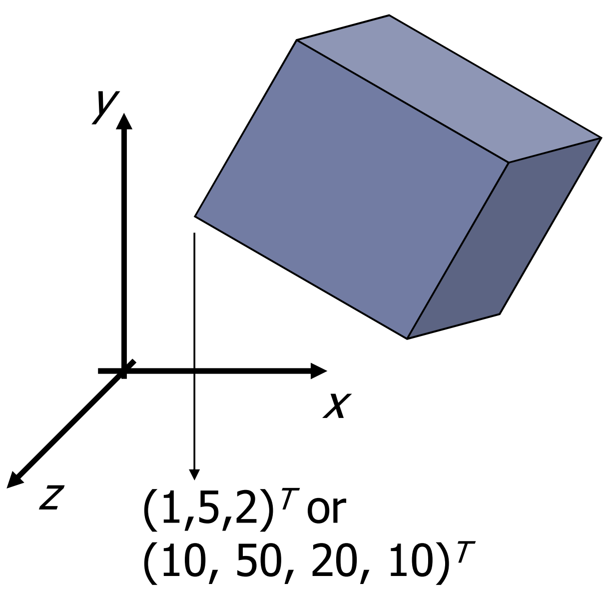
  $\mathbf{p} = [x, y]^T$

  Or in homogeneous coordinates

  $\mathbf{p} = [hx, hy, h]^T$

Points

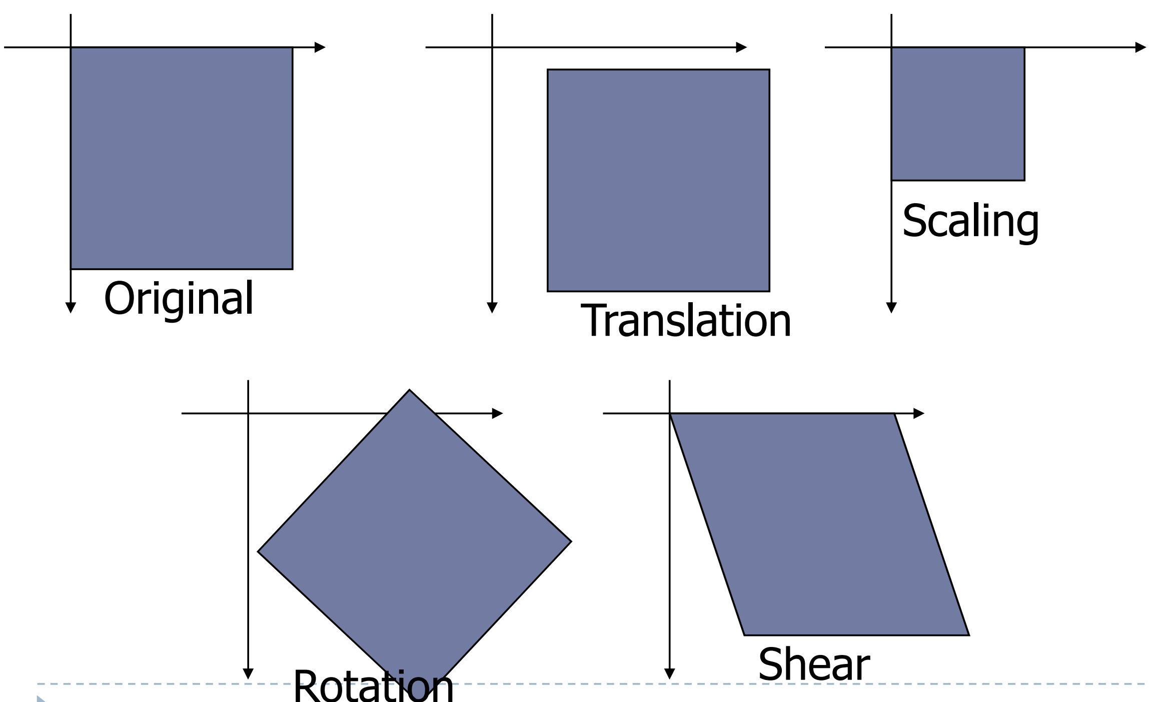




Transformations

- To define a point, we have to define a **coordinate system**
- Transformations are functions that convert points from one coordinate system to another
- Translation, Rotation, Shear, Scaling…

2-D Transformations



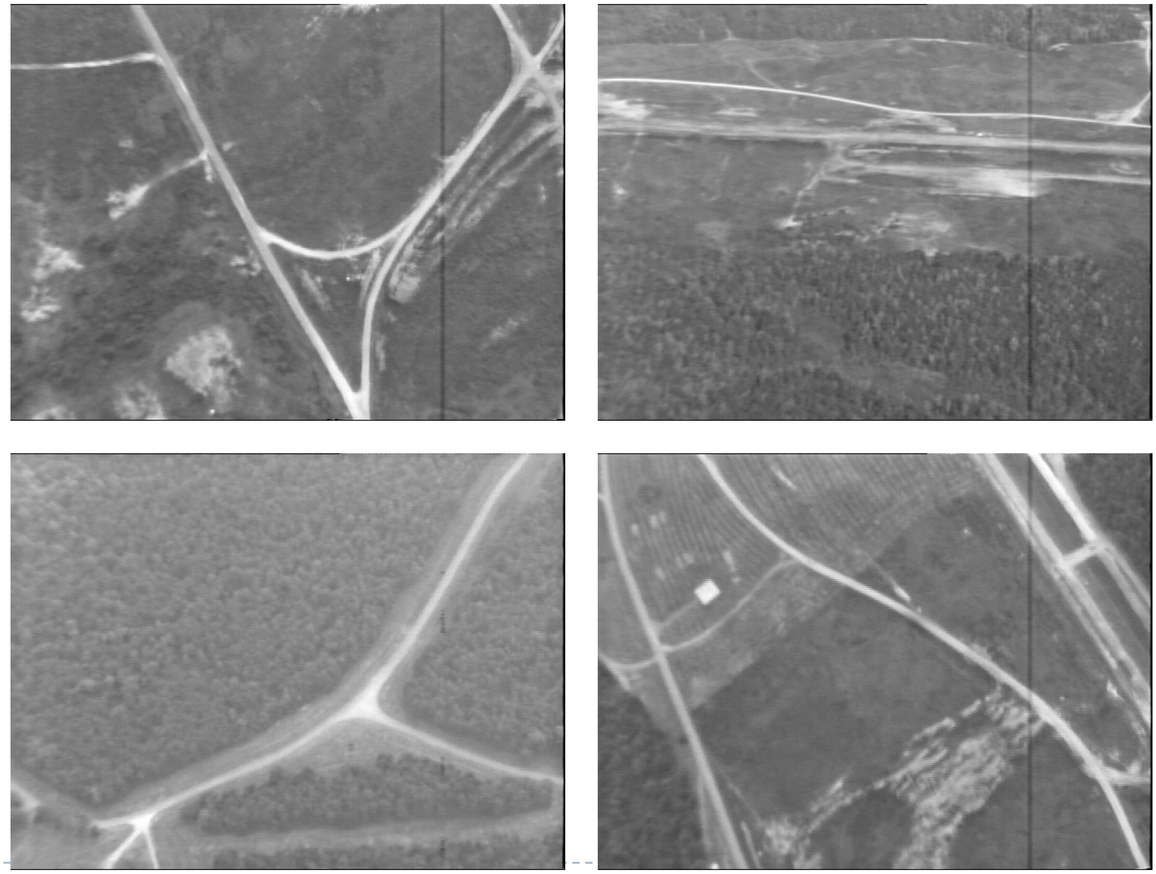
2D Transformations

- **Definition:** A mapping from one 2D coordinate system to another
- Also called
  - *spatial transformation,*
  - *geometric transformation,*
  - *warp*
- Image Registration: Process of transforming two images so that same features overlap

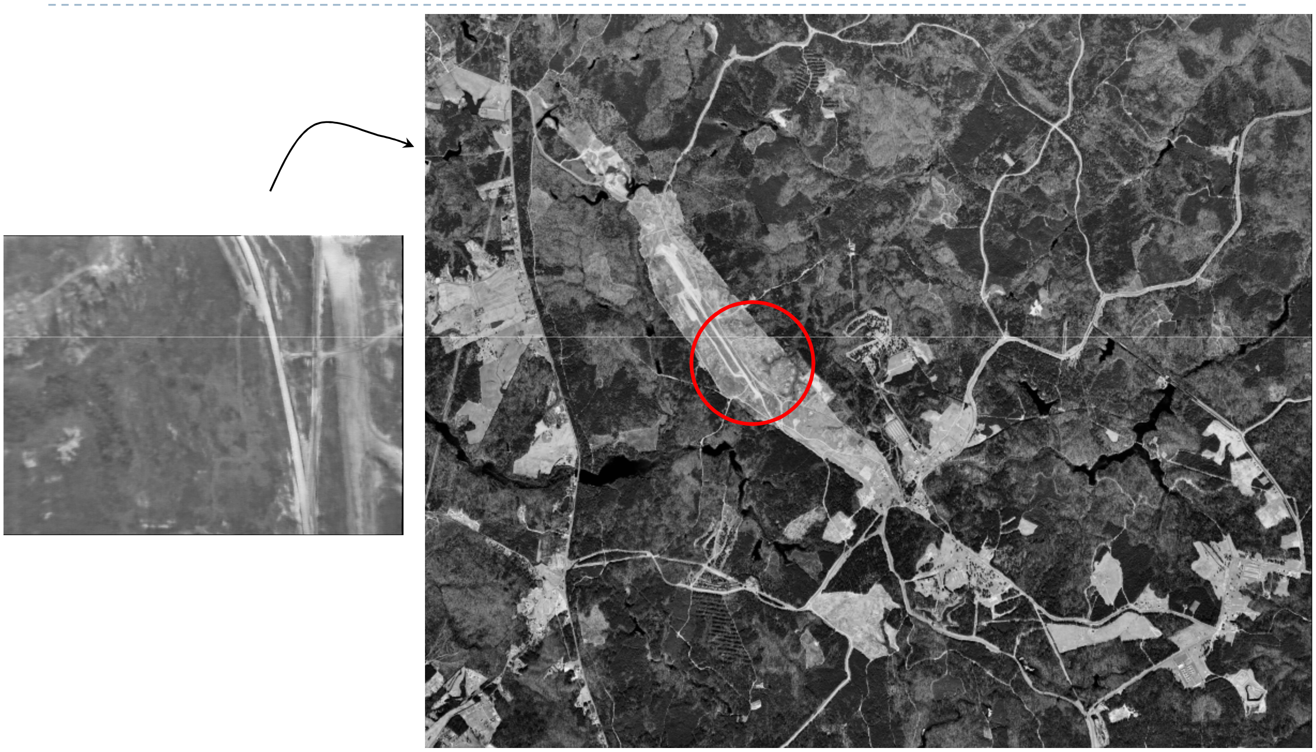
Example Application: Image Registration

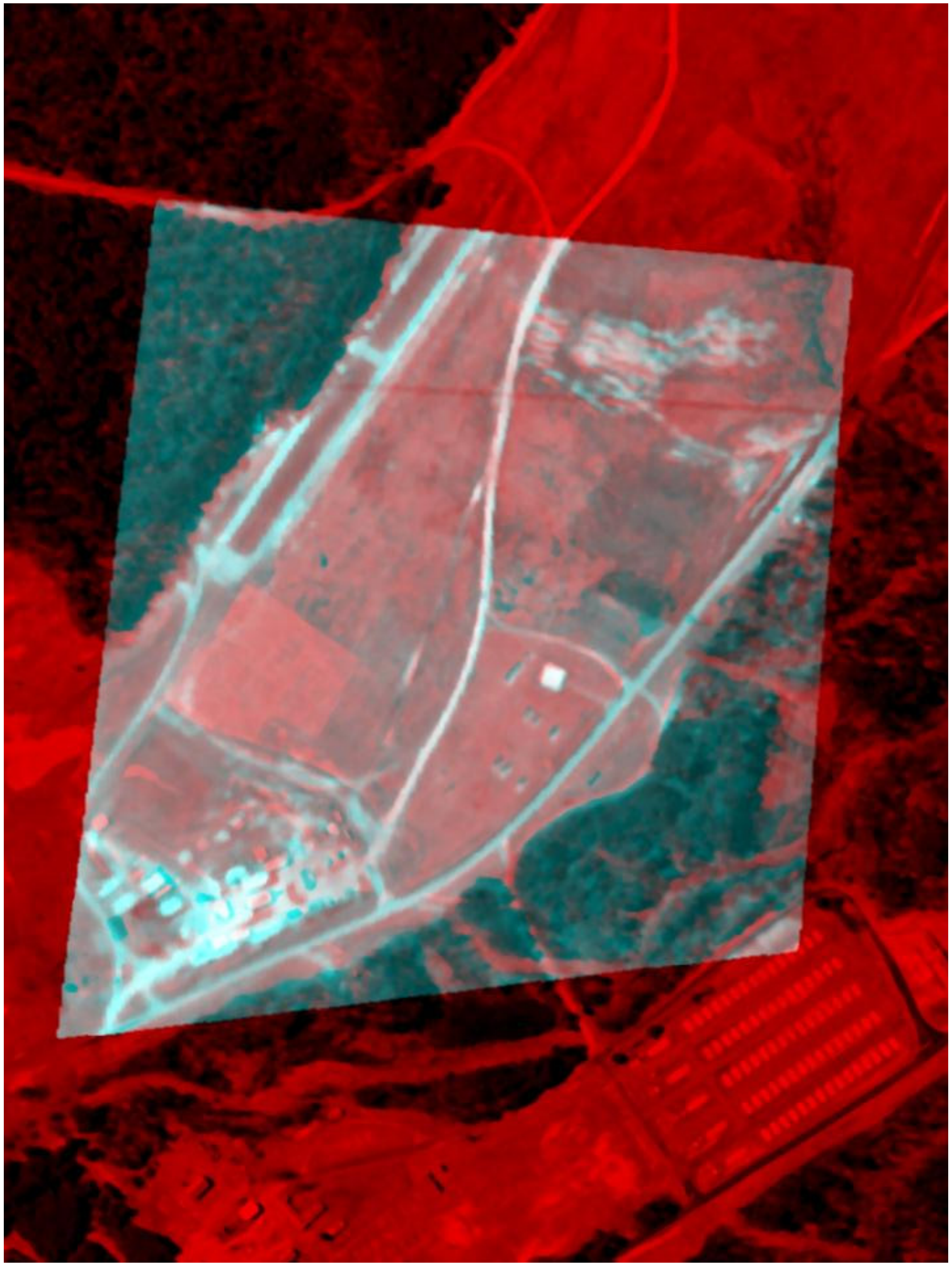
Reference Image

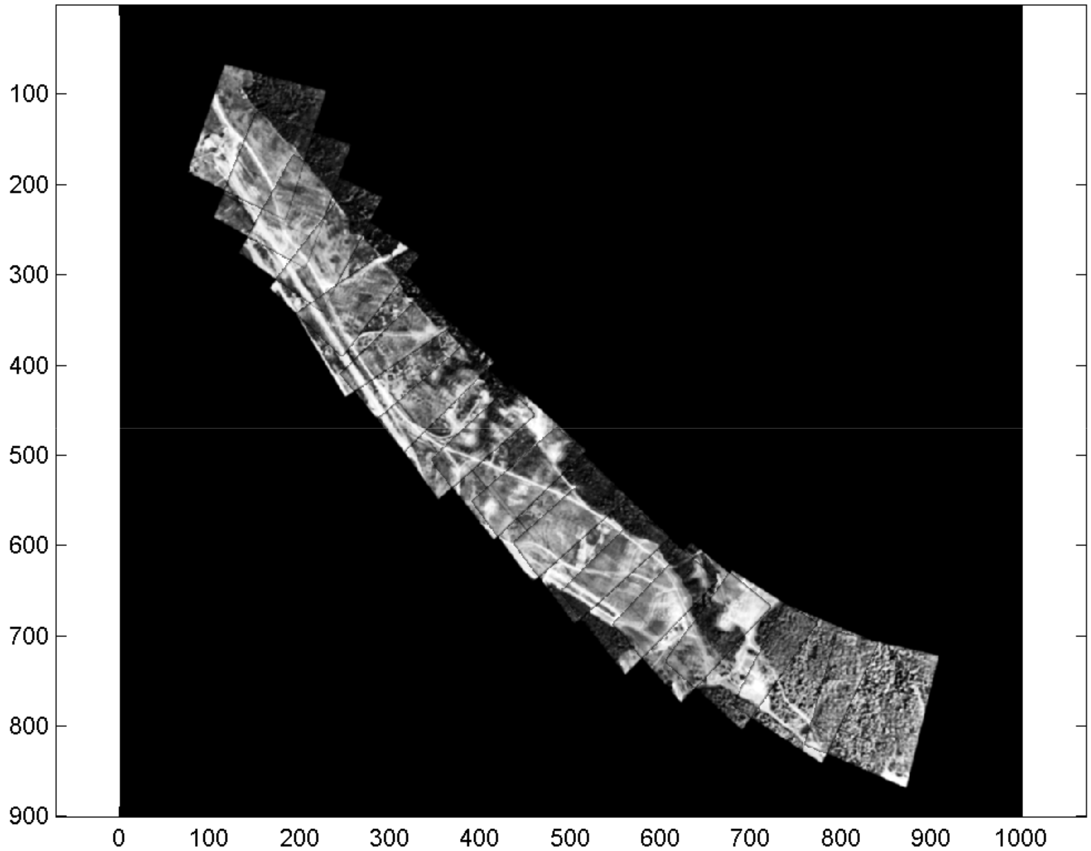
Mission Images



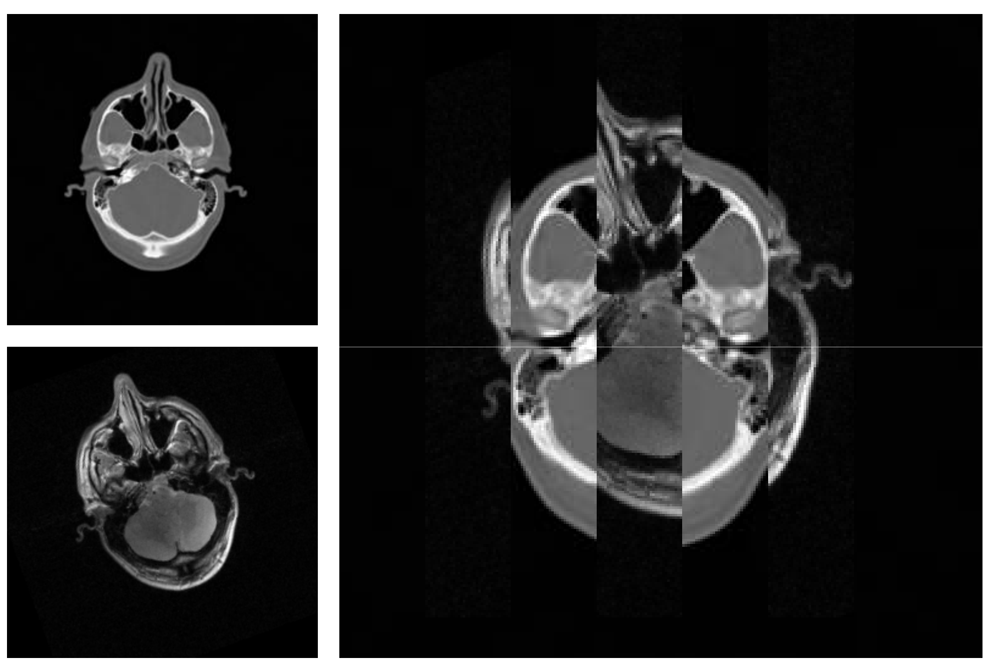
Registration = Computing Transformation & Warping

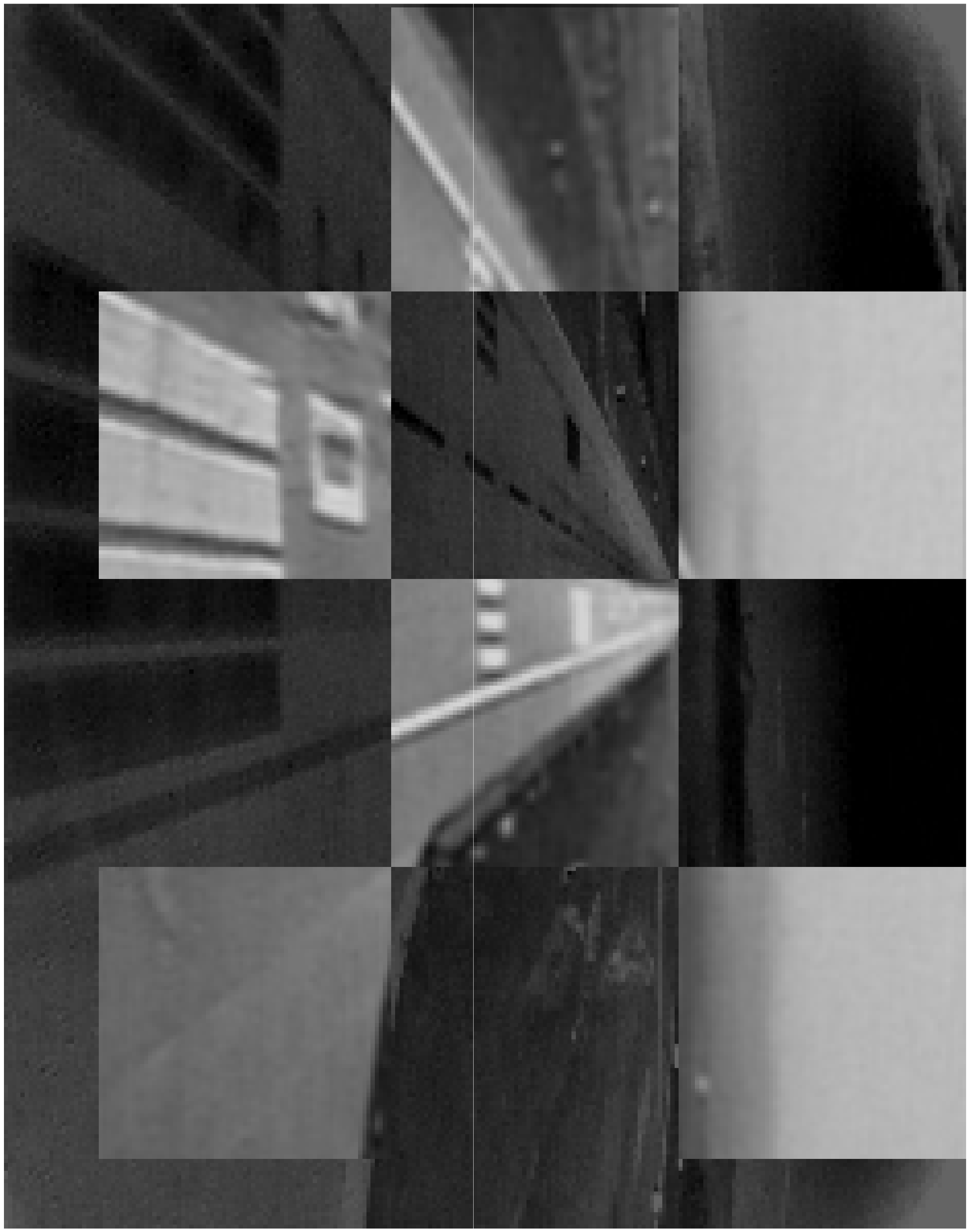


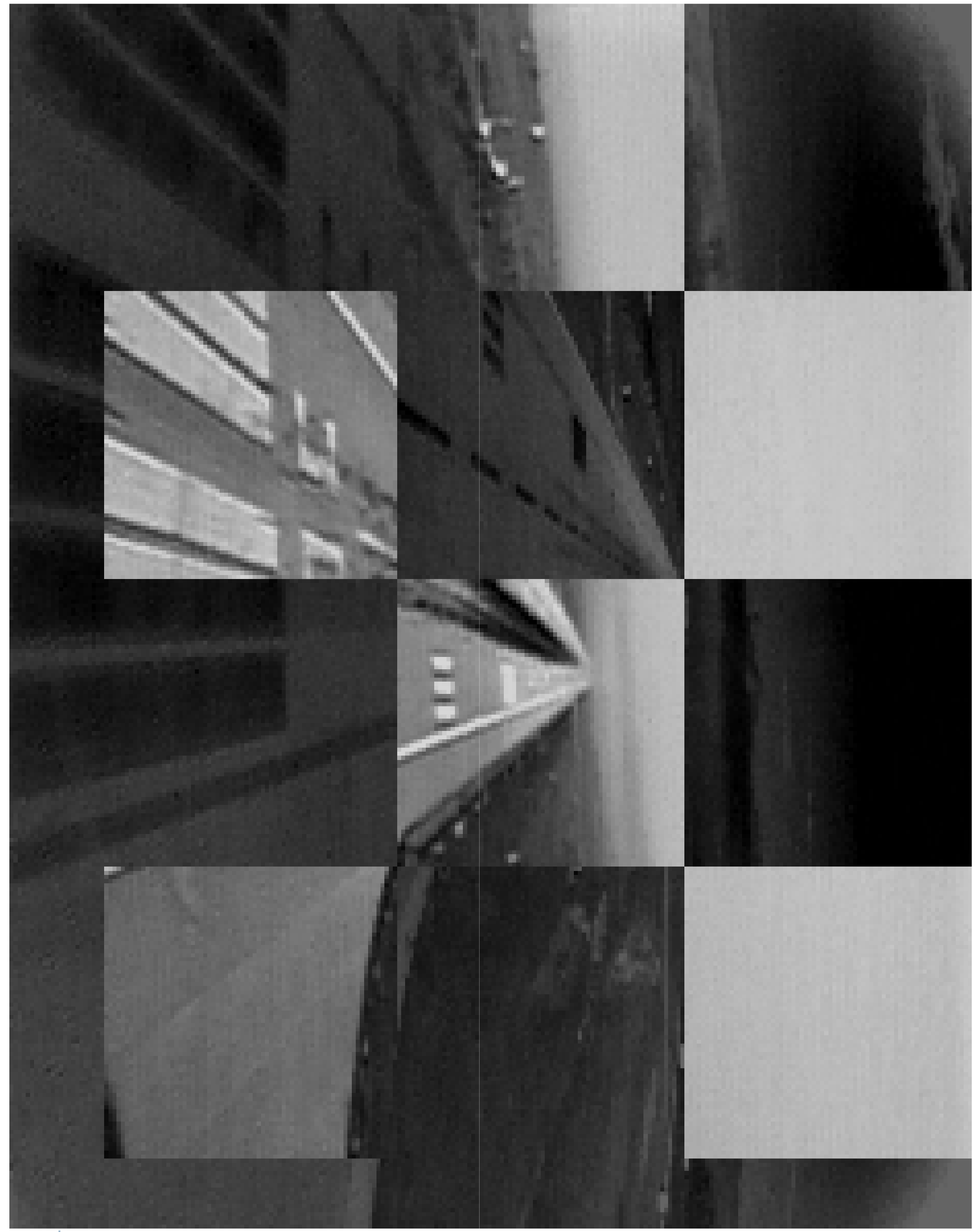




100
200
300
400
500
600
700
800
900
0
100
200
300
400
500
600
700
800
900
1000







2D Transformations

- Basic operation of all 2D transformations is simple

  Point to be transformed: [*x*, *y*]

  Point after transformation: [*x'*, *y'*]

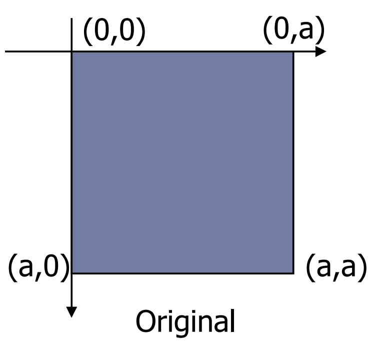
$$\begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_1 x & a_2 y \\ a_3 x & a_4 y \end{bmatrix} = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

Transformation Matrix

Position before transformation

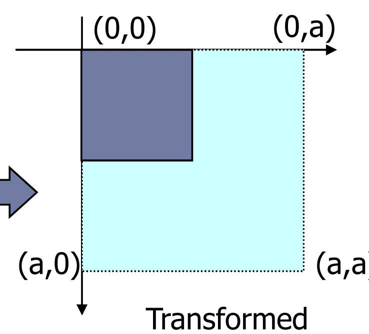
Position after transformation

Example



Apply

$$\begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}$$



$$\begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}\begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}\begin{bmatrix} a \\ a \end{bmatrix} = \begin{bmatrix} 0.5a \\ 0.5a \end{bmatrix}$$

$$\begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}\begin{bmatrix} a \\ 0 \end{bmatrix} = \begin{bmatrix} 0.5a \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}\begin{bmatrix} 0.5a \\ 0.5a \end{bmatrix} = \begin{bmatrix} 0.25a \\ 0.25a \end{bmatrix}$$

2D Transformations

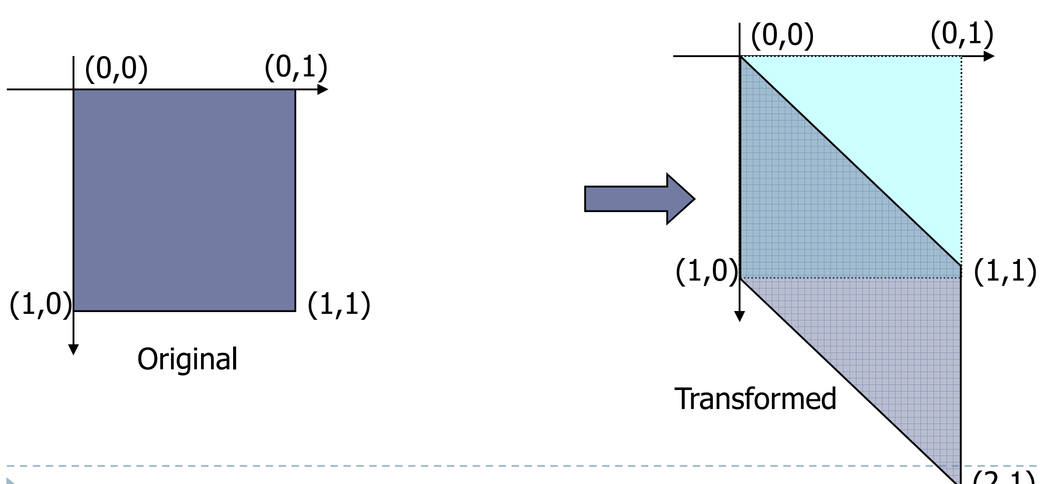
$$\begin{bmatrix} 1.5 & 0 \\ 0 & 0.5 \end{bmatrix} = ?$$

In general, scaling transformation is given by

$$\begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix}$$

2D Transformations

$$\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} = ?$$



Shear in x-direction

$$\begin{bmatrix} 1 & e \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x + ey \\ y \end{bmatrix}$$

- x-coordinate moves with an amount proportional to the y-coordinate

Shear in y-direction

$$\begin{bmatrix} 1 & 0 \\ e & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x \\ ex + y \end{bmatrix}$$

- y-coordinate moves with an amount proportional to the x-coordinate

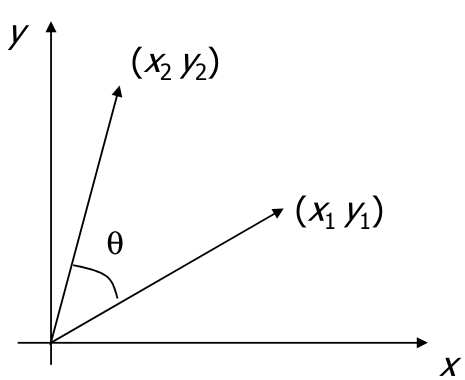
2D Transformations

$$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} = ? \qquad \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} = ?$$

$$\begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} = ?$$

Reflection is negative scaling

Rotation

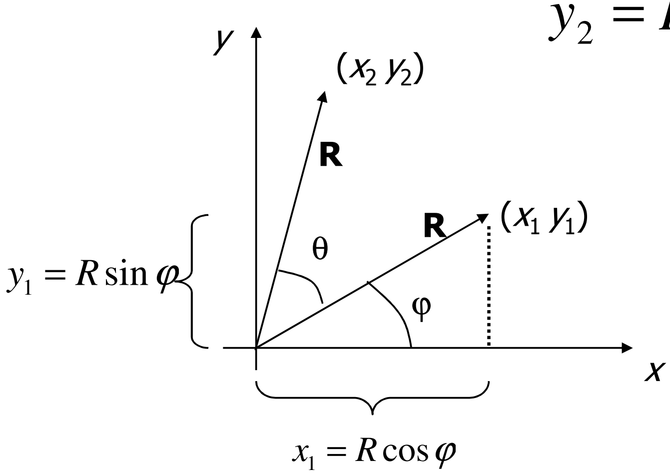


- Task: Relate $(x_2\ y_2)$ to $(x_1\ y_1)$

Rotation

$$x_2 = R\cos(\theta+\varphi)$$
$$y_2 = R\sin(\theta+\varphi)$$

$$x_2 = R\cos\theta\cos\varphi - R\sin\theta\sin\varphi$$
$$y_2 = R\sin\theta\cos\varphi + R\cos\theta\sin\varphi$$



$$x_2 = x_1\cos\theta - y_1\sin\theta$$
$$y_2 = x_1\sin\theta + y_1\cos\theta$$

$$\begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \underbrace{\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}}_{\mathbf{R}} \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}$$

R is rotation by θ **counterclockwise about origin**

Rotation

- Rotation Matrix has some special properties
  - Each row/column has norm of 1 [prove]
  - Each row/column is orthogonal to the other [prove]
  - So Rotation matrix is an **orthonormal** matrix

2D Translation

- Point in 2D given by (x_1 y_1)
- Translated by (d_x d_y)

$$x_2 = x_1 + d_x$$
$$y_2 = y_1 + d_y$$

Translation

- In matrix form

$$\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & d_x \\ 0 & 1 & d_y \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{T}} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

- We could not have written **T** multiplicatively without using homogeneous coordinates

Basic 2D Transformations

$$\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & d_x \\ 0 & 1 & d_y \\ 0 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} 1 & e_x & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} 1 & 0 & 0 \\ e_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Inverse Scaling

$$\mathbf{S} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{S}^{-1} = \begin{bmatrix} \frac{1}{s_x} & 0 & 0 \\ 0 & \frac{1}{s_y} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{S}\,\mathbf{S}^{-1} = \boldsymbol{I}$$

Inverse Translation

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & d_x \\ 0 & 1 & d_y \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{T}^{-1} = \begin{bmatrix} 1 & 0 & -d_x \\ 0 & 1 & -d_y \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{T}\,\mathbf{T}^{-1} = \mathbf{I}$$

Inverse Rotation

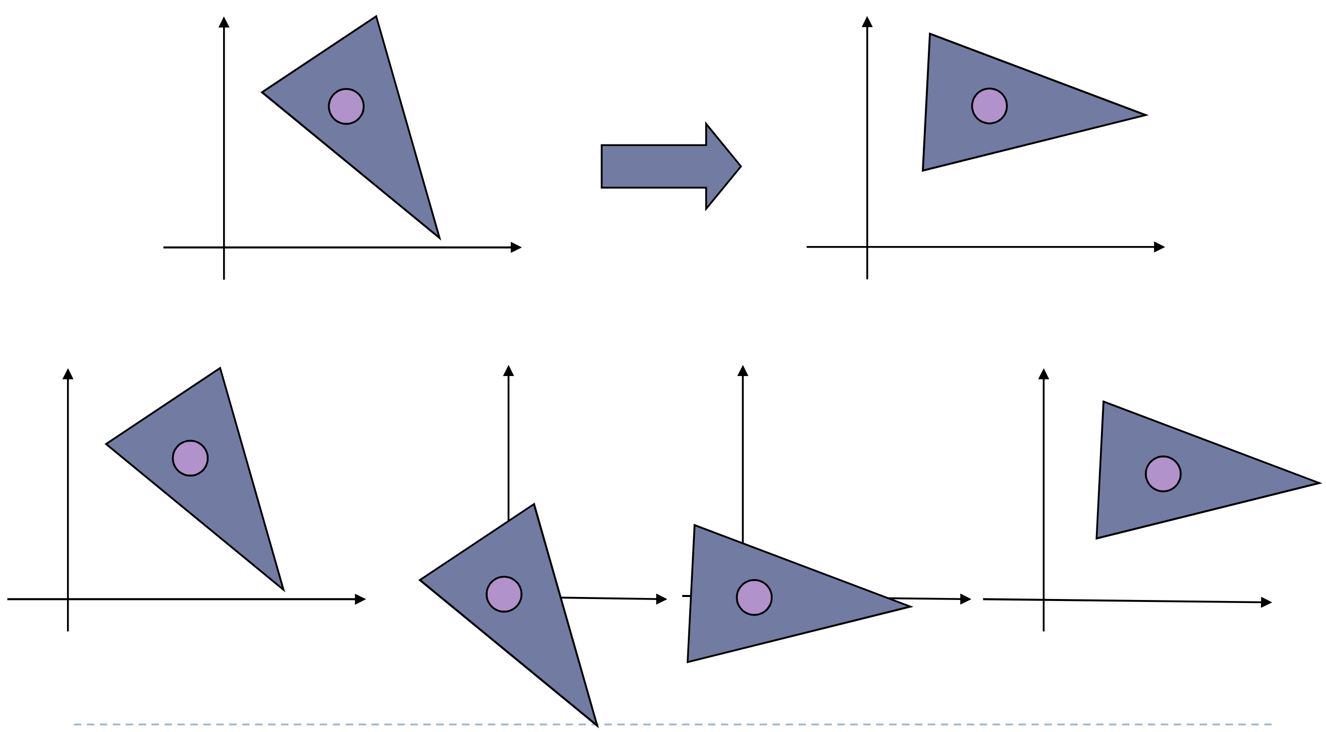
$$\mathbf{R} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{R}^{-1} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{R}\mathbf{R}^{-1} = \mathbf{I}$$

Rotation about an Arbitrary Point

- The rotation matrix that we have derived is for rotations about the **origin**
- We may want to rotate about some other point
- **Solution**?
- Translate point of rotation to origin, rotate using normal rotation matrix, translate back

Rotation about an Arbitrary Point



Concatenation or Composition of Transformations

- Suppose we first want to scale, then rotate

  x' = Sx

  x'' = Rx'

  = R(Sx)

  = (RS) x
- So two transformations can be represented by a **single** transformation matrix

  M = RS
- Important: read from right-side to get order of application of transformations

Concatenation or Composition of Transformations

- We can concatenate a large number of transformations into a single transformation
- $\mathbf{p}_2 = \mathbf{T}_{[dx\ dy]}\ \mathbf{S}_{[s\ s]} \mathbf{R}_{\theta}\ \mathbf{p}_1$
- Rules of matrix multiplication apply
- If we do not use homogeneous coordinates, what might be the problem here?

Example

$$\mathbf{S} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix}$$

$$\mathbf{R} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

Then

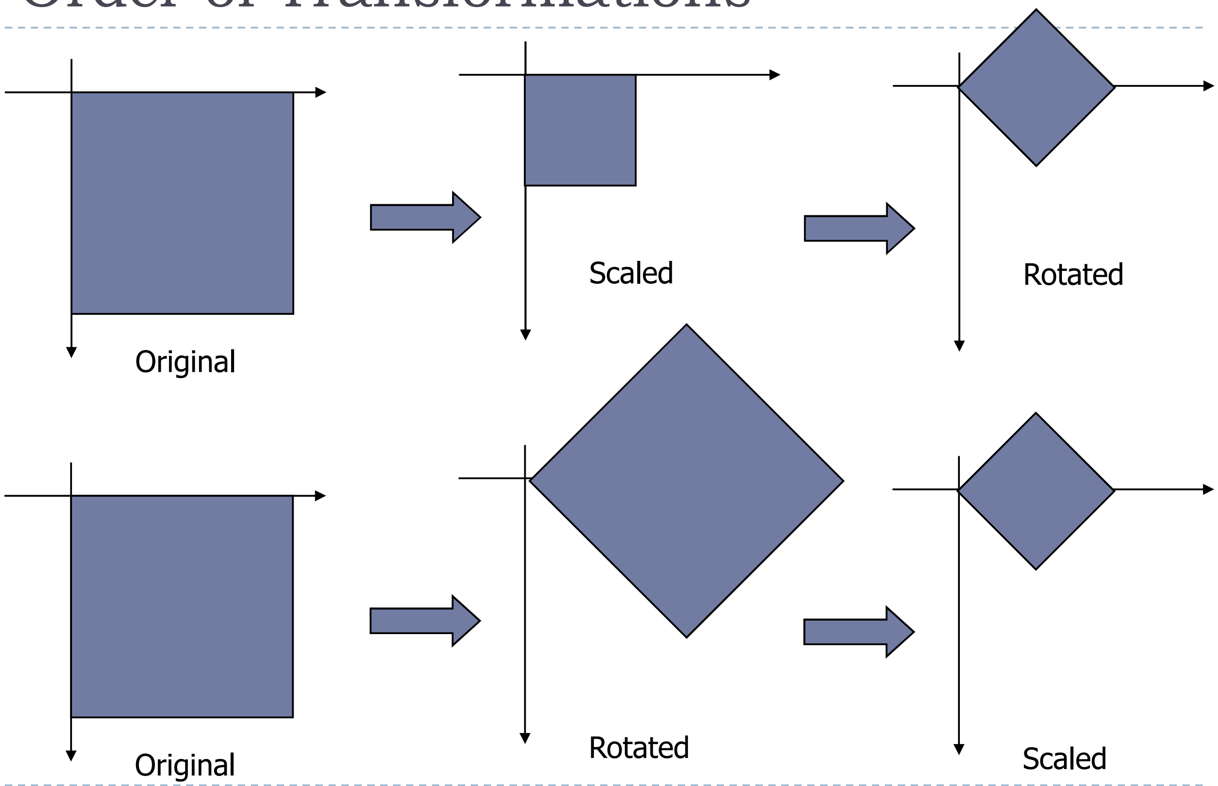
$$\mathbf{RS} = \begin{bmatrix} s_x\cos\theta & -s_y\sin\theta \\ s_x\sin\theta & s_y\cos\theta \end{bmatrix}$$

$$\mathbf{SR} = \begin{bmatrix} s_x\cos\theta & -s_x\sin\theta \\ s_y\sin\theta & s_y\cos\theta \end{bmatrix}$$

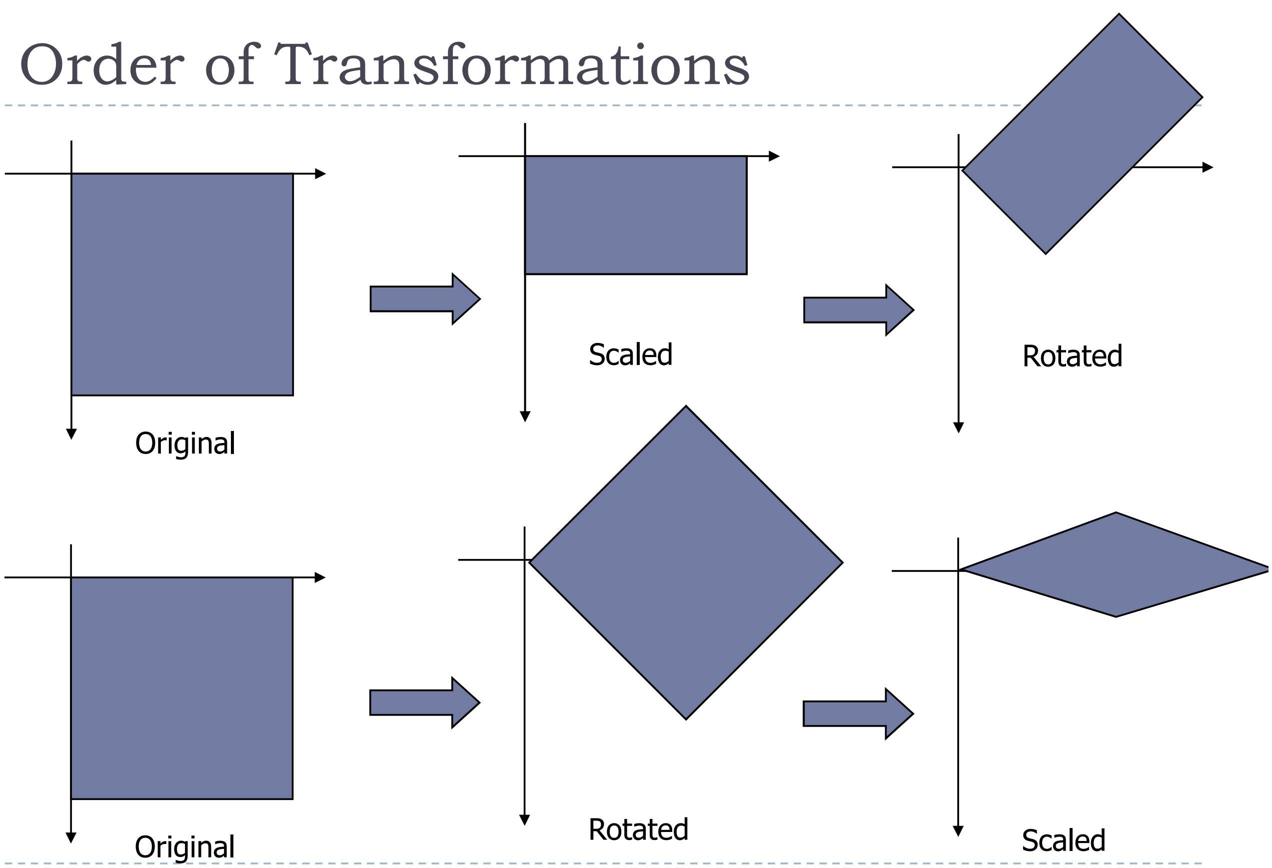
Order of Transformations

- In general $\mathbf{AB} \neq \mathbf{BA}$
- However, in specific cases, this might hold true
- In the previous example, if $s_x = s_y$, then order of transformations does not matter

Order of Transformations



Order of Transformations



Order of Transformations

- Rotation/Scaling/Shear, followed by Translation

$$\begin{bmatrix} 1 & 0 & b_1 \\ 0 & 1 & b_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a_1 & a_2 & 0 \\ a_3 & a_4 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & b_1 \\ a_3 & a_4 & b_2 \\ 0 & 0 & 1 \end{bmatrix}$$

- Translation, followed by Rotation/Scaling/Shear

$$\begin{bmatrix} a_1 & a_2 & 0 \\ a_3 & a_4 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & b_1 \\ 0 & 1 & b_2 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_1 b_1 + a_2 b_2 \\ a_3 & a_4 & a_3 b_1 + a_4 b_2 \\ 0 & 0 & 1 \end{bmatrix}$$

Affine Transformation

- Consider again the transformation matrix of Rotation/Scaling/Shear, followed by Translation

$$\begin{bmatrix} a_1 & a_2 & b_1 \\ a_3 & a_4 & b_2 \\ 0 & 0 & 1 \end{bmatrix}$$

- This is known as the **affine transformation**

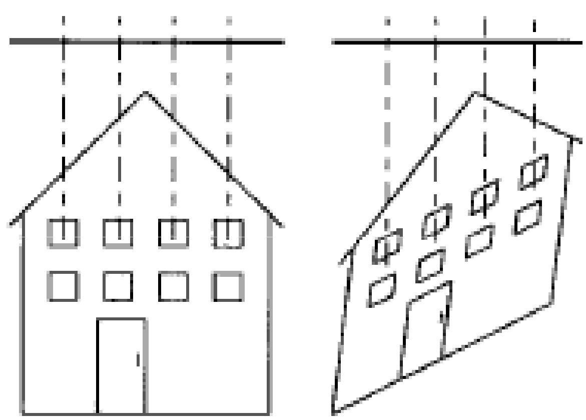
Affine Transformation

- Encodes rotation, scaling, translation and shear

$$x_2 = a_1 x_1 + a_2 y_1 + b_1$$
$$y_2 = a_3 x_1 + a_4 y_1 + b_2$$

- 6 parameters
- Linear transformation
- Parallel lines are preserved [proof ?]

Affine Transformation

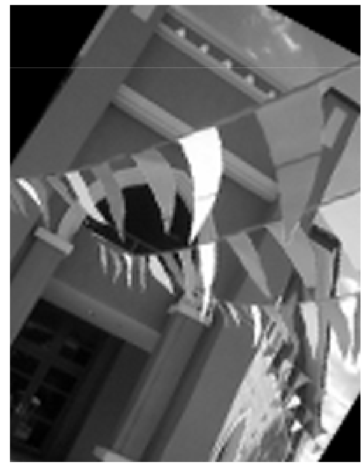


(ORIGINAL) (AFFINE)

Ref: Steve Mann & Rosalind W. Picard, "Video Orbits of the Projective Group: A simple approach to featureless estimation of parameters", IEEE Trans. on Image Processing, Vol. 6, No. 9, September 1997

Affine Transformation

- If $[a_1 \ldots a_4]$ are restricted to pure rotation matrix, then this case is called **rigid-body** or **euclidean** transformation



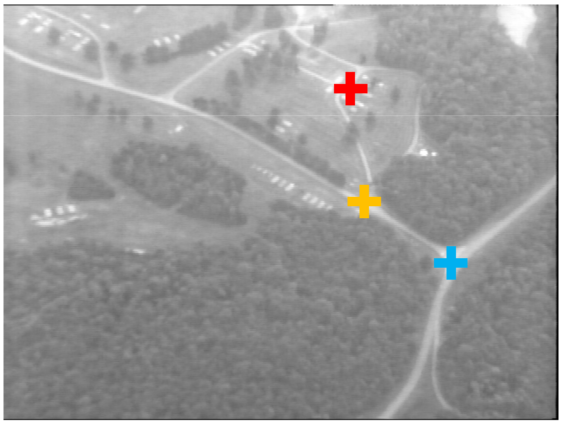
Recovering Best Affine Transformation

- Given two images with unknown transformation between them…

- Compute the values for $[a_1 \ldots a_6]$

Recovering Best Affine Transformation

- Input: we are given some correspondences
- Output: Compute $a_1 - a_6$ which relate the images

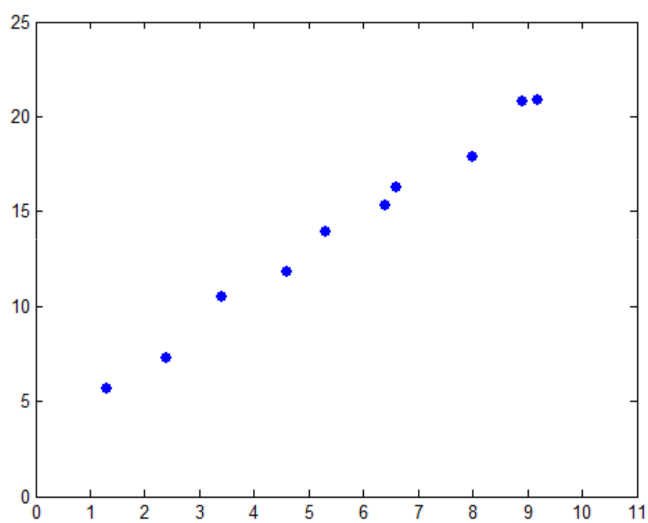


- This is an optimization problem… Find the 'best' set of parameters, given the input data

Parameter Optimization: Least Squared Error Solutions

- Let us first consider the 'simpler' problem of fitting a line to a set of data points…

x	y
1.3	5.7
2.4	7.3
3.4	10.5
4.6	11.8
5.3	13.9
6.6	16.3
6.4	15.3
8.0	17.9
8.9	20.8
9.2	20.9



- Equation of best fit line ?

Line Fitting: Least Squared Error Solution

- Step 1: Identify the model
  - Equation of line: $y = mx + c$
- Step 2: Set up an error term which will give the goodness of every point with respect to the (unknown) model
  - Error induced by i^{th} point:
  - $e_i = mx_i + c - y_i$
  - Error for whole data: $E = \Sigma_i e_i^2$
  - $E = \Sigma_i (mx_i + c - y_i)^2$
- Step 3: Differentiate Error w.r.t. parameters, put equal to zero and solve for minimum point

Line Fitting: Least Squared Error Solution

$$E = \sum_i (mx_i + c - y_i)^2$$

$$\frac{\partial E}{\partial m} = \sum_i (mx_i + c - y_i) x_i = 0$$

$$\frac{\partial E}{\partial c} = \sum_i (mx_i + c - y_i) = 0$$

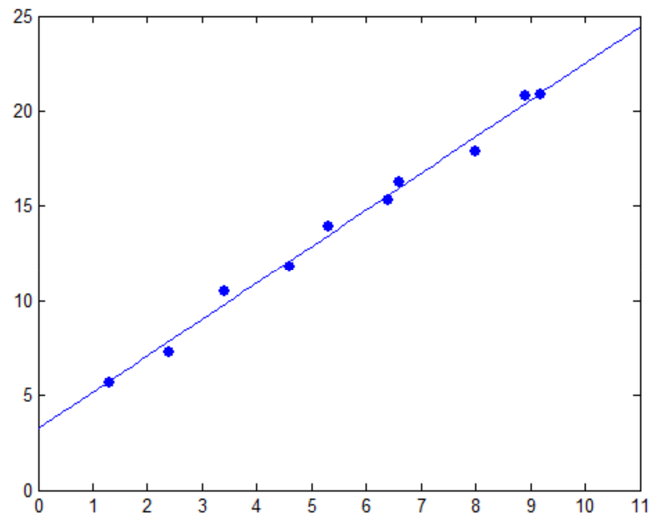
$$\begin{bmatrix} \sum_i x_i^2 & \sum_i x_i \\ \sum_i x_i & \sum_i 1 \end{bmatrix} \begin{bmatrix} m \\ c \end{bmatrix} = \begin{bmatrix} \sum_i x_i y_i \\ \sum_i y_i \end{bmatrix}$$

x	y
1.3	5.7
2.4	7.3
3.4	10.5
4.6	11.8
5.3	13.9
6.6	16.3
6.4	15.3
8.0	17.9
8.9	20.8
9.2	20.9

$$\begin{pmatrix} 380.63 & 56.1 \\ 56.1 & 10 \end{pmatrix} \begin{pmatrix} m \\ c \end{pmatrix} = \begin{pmatrix} 914.68 \\ 140.4 \end{pmatrix}$$

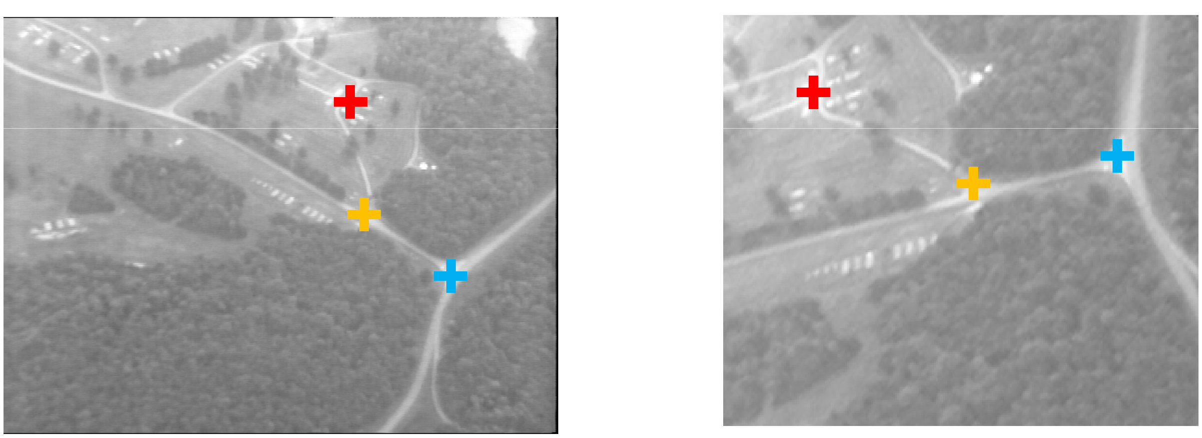
Solution: $m = 1.9274 \quad c = 3.227$

Line Fitting: Least Squared Error Solution



Recovering Best Affine Transformation

- Input: Set of correspondences
  - Image 1: (x_i, y_i) Image 2: (x_i', y_i')



Recovering Best Affine Transformation

- Least Squares Error Solution
  - Is the solution (i.e. set of parameters a_1 … a_6) such that the sum of the square of error in each corresponding point is as minimum as possible
  - No other set of parameters exists that may have a lower error (in the squared error sense)

Recovering Best Affine Transformation

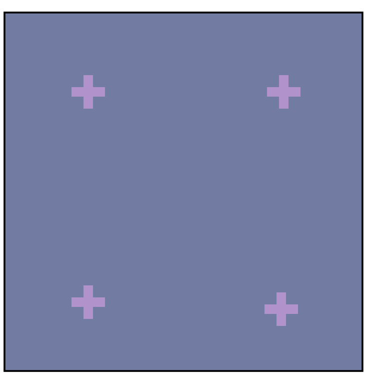


Image 1

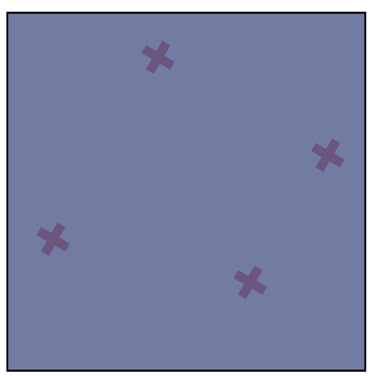
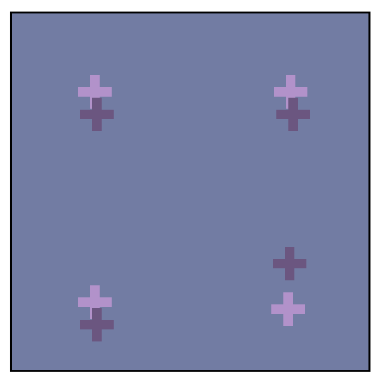


Image 2

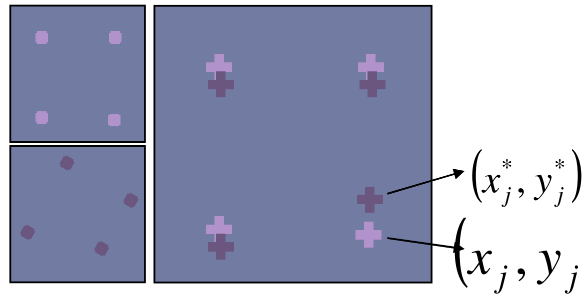


Overlap of points after recovering the transformation

We can try to find the set of parameters in which the **error** is minimum

Least Squares Error Solution

$$\begin{bmatrix} x_j^* \\ y_j^* \\ 1 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x'_j \\ y'_j \\ 1 \end{bmatrix}$$



$$\boxed{E(a_1,a_2,a_3,a_4,a_5,a_6) = \sum_{j=1}^{n} (x_j^* - x_j)^2 + (y_j^* - y_j)^2}$$

$$E(\mathbf{a}) = \sum_{j=1}^{n} \left((a_1 x'_j + a_2 y'_j + a_3 - x_j)^2 + (a_4 x'_j + a_5 y'_j + a_6 - y_j)^2 \right)$$

Least Squares Error Solution

$$E(\mathbf{a}) = \sum_{j=1}^{n} \left((a_1 x_j + a_2 y_j + a_3 - x'_j)^2 + (a_4 x_j + a_5 y_j + a_6 - y'_j)^2 \right)$$

- Minimize E w.r.t. **a**
- Compute $\partial E / \partial a_i$, put equal to zero, solve simultaneously

$$\begin{bmatrix} \sum_j x_j^2 & \sum_j x_j y_j & \sum_j x_j & 0 & 0 & 0 \\ \sum_j x_j y_j & \sum_j y_j^2 & \sum_j y_j & 0 & 0 & 0 \\ \sum_j x_j & \sum_j y_j & \sum_j 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sum_j x_j^2 & \sum_j x_j y_j & \sum_j x_j \\ 0 & 0 & 0 & \sum_j x_j y_j & \sum_j y_j^2 & \sum_j y_j \\ 0 & 0 & 0 & \sum_j x_j & \sum_j y_j & \sum_j 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{bmatrix} = \begin{bmatrix} \sum_j x_j x'_j \\ \sum_j y_j x'_j \\ \sum_j x'_j \\ \sum_j x_j y'_j \\ \sum_j y_j y'_j \\ \sum_j y'_j \end{bmatrix}$$

Recovering Best Affine Transformation (alternate way)

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{bmatrix} = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

Recovering Best Affine Transformation

- Given three pairs of corresponding points, we get 6 equations

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_2 & y_2 & 1 \\ x_3 & y_3 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_3 & y_3 & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{bmatrix} = \begin{bmatrix} x_1' \\ y_1' \\ x_2' \\ y_2' \\ x_3' \\ y_3' \end{bmatrix}$$

$$\mathbf{Ax=B} \qquad \mathbf{x=A^{-1}B}$$

Recovering Best Affine Transformation

- What if we knew four corresponding points?
- We should be able to utilize the additional information

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_2 & y_2 & 1 \\ x_3 & y_3 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_3 & y_3 & 1 \\ x_4 & y_4 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_4 & y_4 & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{bmatrix} = \begin{bmatrix} x_1' \\ y_1' \\ x_2' \\ y_2' \\ x_3' \\ y_3' \\ x_4' \\ y_4' \end{bmatrix}$$

Recovering Best Affine Transformation

- **Ax = B**
- Cannot take inverse directly
- Also, 4 correspondences may not be exactly represented by an affine transformation [Why ?]

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_2 & y_2 & 1 \\ x_3 & y_3 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_3 & y_3 & 1 \\ x_4 & y_4 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_4 & y_4 & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{bmatrix} = \begin{bmatrix} x_1' \\ y_1' \\ x_2' \\ y_2' \\ x_3' \\ y_3' \\ x_4' \\ y_4' \end{bmatrix}$$

Pseudo inverse

For an over-constrained linear system

$\mathbf{Ax = B}$

$\mathbf{A}$ has more rows than columns

Multiply by $\mathbf{A^T}$ on both sides

$\mathbf{A^T A x = A^T B}$

$\mathbf{A^T A}$ is a square matrix of as many rows as $\mathbf{x}$

We can take its inverse

$\mathbf{x = (A^T A)^{-1} A^T B}$

Pseudo-inverse gives the least squares error solution! [Proof?]

Recovering Best Affine Transformation

- In general, we may be given *n* correspondences
- Concatenate *n* correspondences in **A** and **B**
- **A** is $2n \times 6$
- **B** is $2n \times 1$
- Solve using Least Squares
- $\mathbf{x = (A^T A)^{-1} A^T B}$

2D Displacement Models

- **Translation:**

$$x' = x + b_1$$
$$y' = y + b_2$$

- **Rigid:**

$$x' = x\cos\theta - y\sin\theta + b_1$$
$$y' = x\sin\theta + y\cos\theta + b_2$$

- **Affine:**

$$x' = a_1 x + a_2 y + b_1$$
$$y' = a_3 x + a_4 y + b_2$$

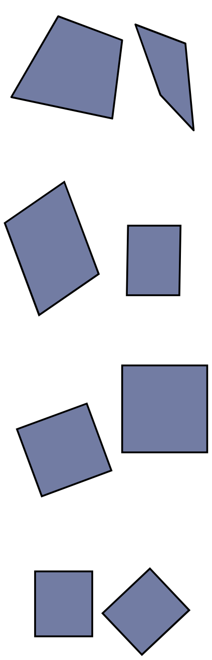
- **Projective:**

$$x' = \frac{a_1 x + a_2 y + b_1}{c_1 x + c_2 y + 1}$$
$$y' = \frac{a_3 x + a_4 y + b_2}{c_1 x + c_2 y + 1}$$

Difference b/w affine and rigid?

Affine does not have the orthonormality constraint

Stratification



Projective
8dof

Concurrency, collinearity, order of contact (intersection, tangency, inflection, etc.), cross ratio

Affine
6dof

Parallellism, ratio of areas, ratio of lengths on parallel lines (e.g midpoints), linear combinations of vectors (centroids).

Similarity
4dof

Ratios of lengths, angles.

Euclidean
3dof

lengths, areas.

Ref: Marc Pollefeys

Projective Displacement Model

- 8 unknowns
- Need homogeneous coordinates to express in matrix form
- Physical interpretation: plane + camera

Other Displacement Models

- Biquadratic:

$$x' = a_1 + a_2 x + a_3 y + a_4 x^2 + a_5 y^2 + a_6 xy$$

$$y' = a_7 + a_8 x + a_9 y + a_{10} x^2 + a_{11} y^2 + a_{12} xy$$

- Bilinear:

$$x' = a_1 + a_2 x + a_3 y + a_4 xy$$

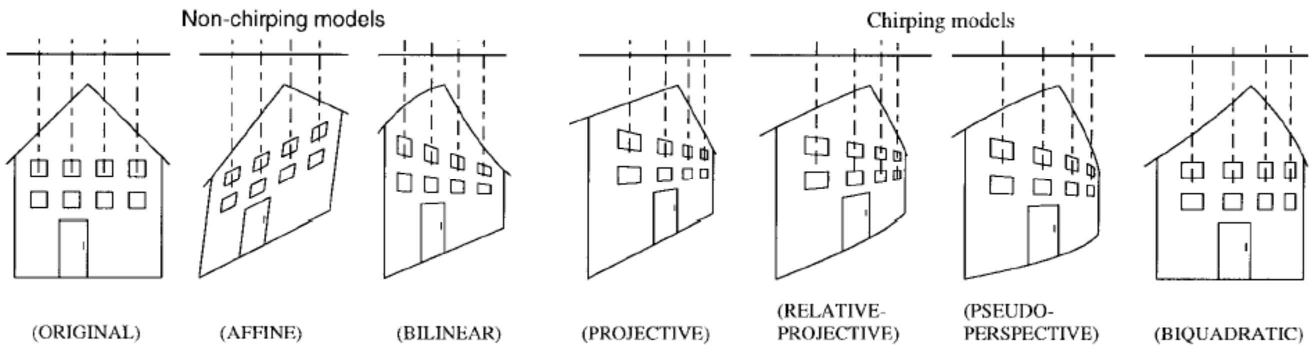
$$y' = a_5 + a_6 x + a_7 y + a_8 xy$$

- Pseudo-perspective

$$x' = a_1 + a_2 x + a_3 y + a_4 x^2 + a_5 xy$$

$$y' = a_6 + a_7 x + a_8 y + a_4 y^2 + a_5 xy$$

Displacement Models



- Ref: Steve Mann & Rosalind W. Picard, "Video Orbits of the Projective Group: A simple approach to featureless estimation of parameters", IEEE Trans. on Image Processing, Vol. 6, No. 9, September 1997

http://wearcam.org/orbits/gallery.html

2D Affine Warping

Warping

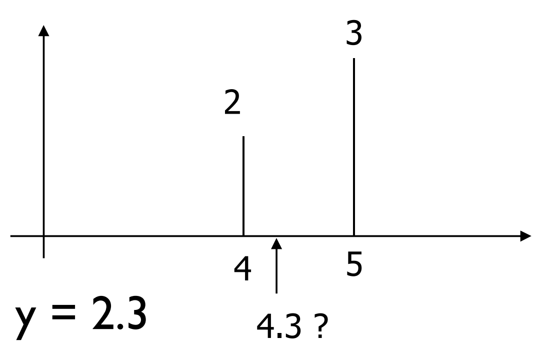
- Inputs:
  - Image X
  - Affine Transformation A = $[a_1\ a_2\ b_1\ a_3\ a_4\ b_2]^T$
- Output:
  - Generate X' such that X' = AX
- Obvious Process:
  - For each pixel in X
  - Apply transformation
  - At that location in X', put the same color as at the original location in X
- Problems?

Warping

- This will leave holes…
  - Because every pixel does not map to an integer location!
- Reverse Transformation
- For each integer location in X'
- Apply inverse mapping
  - Problem?
- Will not result in answers at integer locations, in general
- Bilinearly interpolate from 4 neighbors

Interpolation…

- In 1D



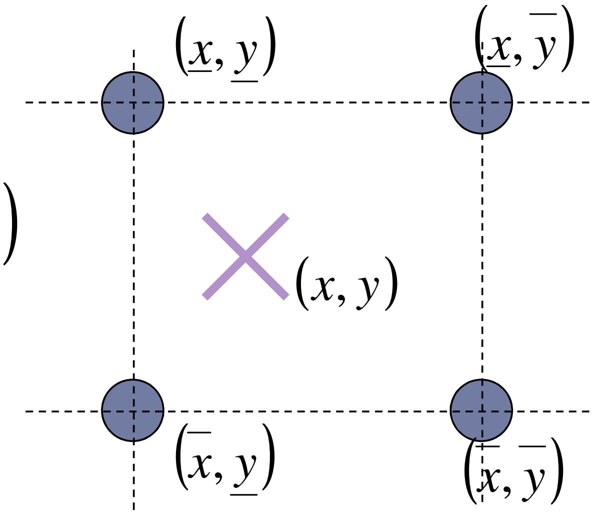
- Use y=mx+c
  - m = 1, c = -2
- Substitute x=4.3, => y = 2.3

2D Bilinear Interpolation

- Four nearest points of (x, y)

$$\left(\underline{x}, \underline{y}\right), \left(\underline{x}, \overline{y}\right), \left(\overline{x}, \underline{y}\right), \left(\overline{x}, \overline{y}\right)$$

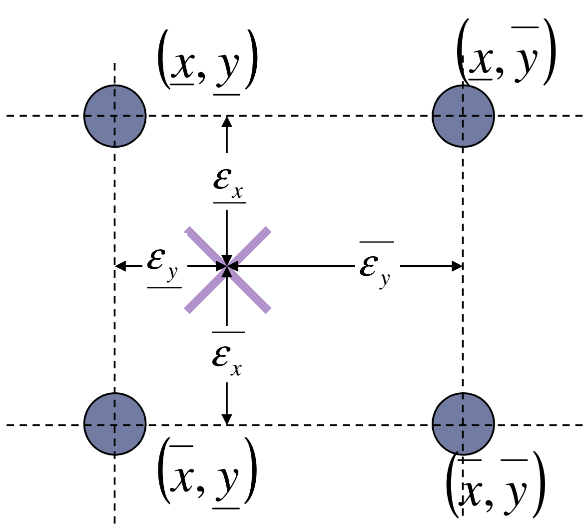
where
$$\underline{x} = \operatorname{int}(x)$$
$$\underline{y} = \operatorname{int}(y)$$
$$\overline{x} = \underline{x} + 1$$
$$\overline{y} = \underline{y} + 1$$



Bilinear Interpolation

$$f'(x, y) = \overline{\varepsilon_x}\,\overline{\varepsilon_y} f(\underline{x}, \underline{y}) + \underline{\varepsilon_x}\,\overline{\varepsilon_y} f(\overline{x}, \underline{y}) + \overline{\varepsilon_x}\,\underline{\varepsilon_y} f(\underline{x}, \overline{y}) + \underline{\varepsilon_x}\,\underline{\varepsilon_y} f(\overline{x}, \overline{y})$$

$$\overline{\varepsilon_x} = \overline{x} - x$$
$$\overline{\varepsilon_y} = \overline{y} - y$$
$$\underline{\varepsilon_x} = x - \underline{x}$$
$$\underline{\varepsilon_y} = y - \underline{y}$$



3D Transformations

3D Translation

- Point in 3D given by (X_1 Y_1 Z_1)
- Translated by (*dx dy dz*)

$X_2 = X_1 + dx$

$Y_2 = Y_1 + dy$

$Z_2 = Z_1 + dz$

Translation

- In matrix form

$$\begin{bmatrix} X_2 \\ Y_2 \\ Z_2 \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & dx \\ 0 & 1 & 0 & dy \\ 0 & 0 & 1 & dz \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\mathbf{T}} \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \\ 1 \end{bmatrix}$$

Inverse Transformation

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 & dx \\ 0 & 1 & 0 & dy \\ 0 & 0 & 1 & dz \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{T}^{-1} = \begin{bmatrix} 1 & 0 & 0 & -dx \\ 0 & 1 & 0 & -dy \\ 0 & 0 & 1 & -dz \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{T}\,\mathbf{T}^{-1} = \mathbf{I}$$

Scaling

- Point in 3D given by (X_1 Y_1 Z_1)
- Scaled by (*Sx Sy Sz*)

$X_2 = X_1 * Sx$

$Y_2 = Y_1 * Sy$

$Z_2 = Z_1 * Sz$

Scaling

- In matrix form

$$\begin{bmatrix} X_2 \\ Y_2 \\ Z_2 \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} Sx & 0 & 0 & 0 \\ 0 & Sy & 0 & 0 \\ 0 & 0 & Sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\mathbf{S}} \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \\ 1 \end{bmatrix}$$

Inverse Transformation

$$\mathbf{S} = \begin{bmatrix} Sx & 0 & 0 & 0 \\ 0 & Sy & 0 & 0 \\ 0 & 0 & Sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{S}^{-1} = \begin{bmatrix} \frac{1}{Sx} & 0 & 0 & 0 \\ 0 & \frac{1}{Sy} & 0 & 0 \\ 0 & 0 & \frac{1}{Sz} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{S}\,\mathbf{S}^{-1} = \boldsymbol{I}$$

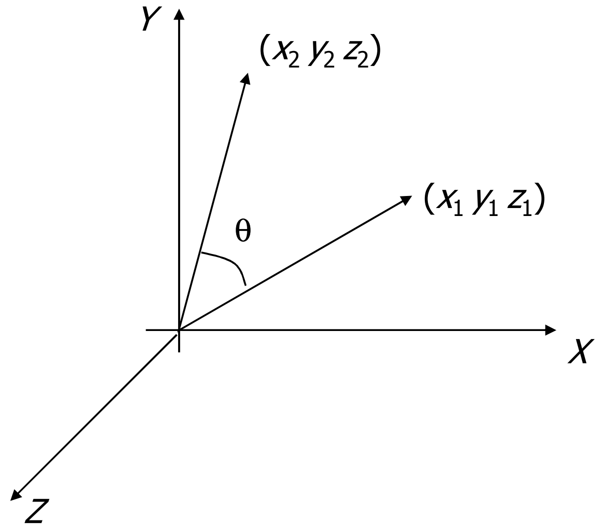
Shearing

- What will these do?

$$\begin{bmatrix} 1 & e & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & e & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & d & e & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & d & 0 & 0 \\ e & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3D Rotation

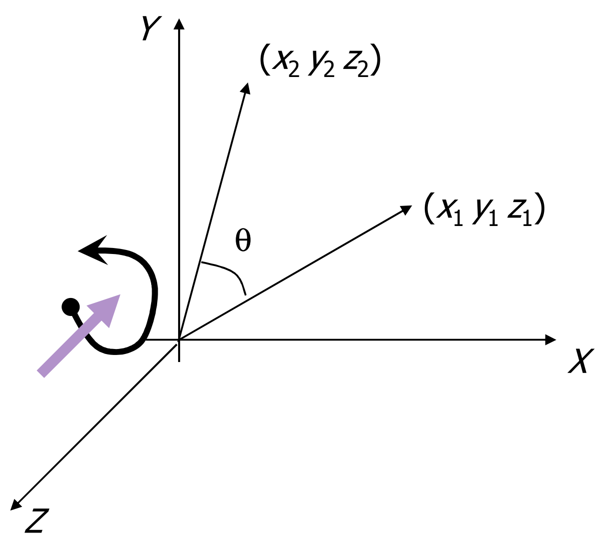
- Rotation about *Z*-axis
- Z-coordinate will not change
- Z' = Z
- If we ignore the Z-coordinate, it is 2-D transformation in XY plane



$$\begin{bmatrix} X' \\ Y' \\ Z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

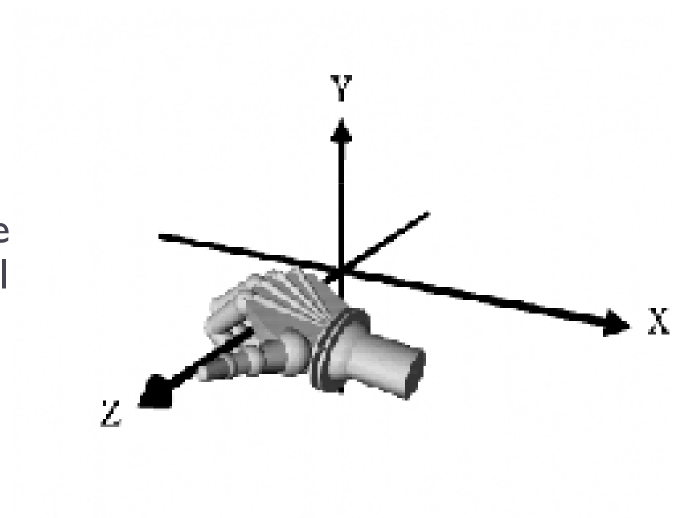
Rotation

- Positive rotation is counterclockwise when **looking down** the axis of rotation towards origin

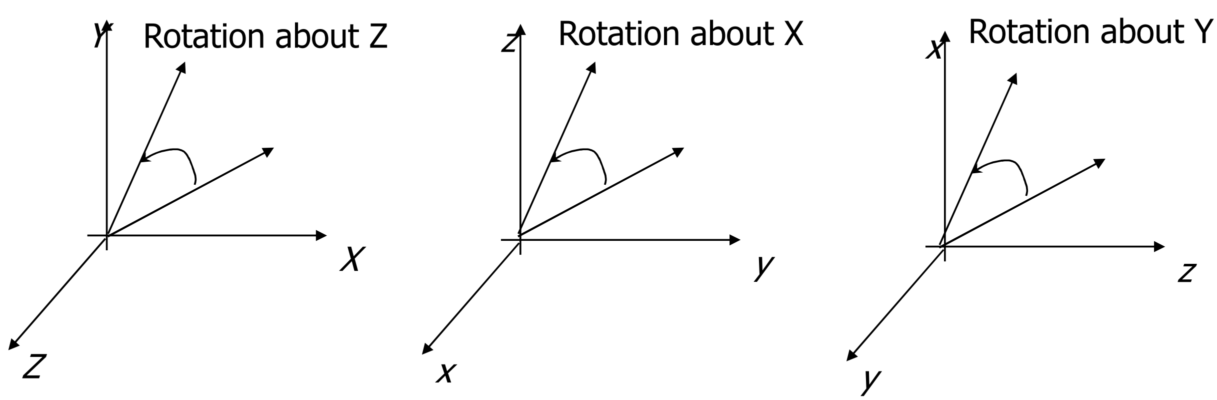


Rotation about Principal Axes

- Right hand rule
- XYZ is a right handed system if:
  - Place your right wrist at the origin, curl your finge from X to Y, thumb shoul point along +ve Z



Rotation about Principal Axes



$X' = X\cos\theta - Y\sin\theta$	$Y' = Y\cos\theta - Z\sin\theta$	$Z' = Z\cos\theta - X\sin\theta$
$Y' = X\sin\theta + Y\cos\theta$	$Z' = Y\sin\theta + Z\cos\theta$	$X' = Z\sin\theta + X\cos\theta$
$Z' = Z$	$X' = X$	$Y' = Y$

Rotation about Principal Axes

$$X' = X\cos\theta - Y\sin\theta$$
$$Y' = X\sin\theta + Y\cos\theta$$
$$Z' = Z$$

Rotation about Z

$$Y' = Y\cos\theta - Z\sin\theta$$
$$Z' = Y\sin\theta + Z\cos\theta$$
$$X' = X$$

Rotation about X

$$Z' = Z\cos\theta - X\sin\theta$$
$$X' = Z\sin\theta + X\cos\theta$$
$$Y' = Y$$

Rotation about Y

$$\begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma & 0 \\ 0 & \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} \cos\beta & 0 & \sin\beta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\beta & 0 & \cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Concatenation of Rotations

- Rotation around X by γ followed by rotation around Y by β followed by rotation around Z by α

$$R = R_\alpha^Z R_\beta^Y R_\gamma^X$$

$$R = \begin{bmatrix} \cos\alpha\cos\beta & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \sin\alpha\cos\beta & \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma \\ -\sin\beta & \cos\beta\sin\gamma & \cos\beta\cos\gamma \end{bmatrix}$$

Small Angle Approximation

$$R = R_Z^{\alpha} R_Y^{\beta} R_X^{\gamma} = \begin{bmatrix} \cos\alpha\cos\beta & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \sin\alpha\cos\beta & \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma \\ -\sin\beta & \cos\beta\sin\gamma & \cos\beta\cos\gamma \end{bmatrix}$$

Small angle approximation

$$R = \begin{bmatrix} 1 & -\alpha & \beta \\ \alpha & 1 & -\gamma \\ -\beta & \gamma & 1 \end{bmatrix}$$

Properties of Rotation Matrix

- Rotation Matrices are **orthonormal**

  i.e. $R R^T = R^T R = I$

- 3D rigid motion can be described by a rotation and translation

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix}$$

Properties of Rotation Matrix

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

- *R* has 9 unknowns, but orthonormality provides 6 constraints

$$\sum_{j=1}^{3} r_{ij} r_{kj} = \sum_{j=1}^{3} r_{ji} r_{jk} = \begin{cases} 0 & i \neq k \\ 1 & i = k \end{cases}$$

- Hence the number of degrees of freedom of a 3D rotation are 9 – 6 = 3

Properties of Rotation Matrices

- Any concatenation of rotation matrices also forms a rotation matrix i.e. the matrix remains orthonormal [Proof?]

Properties of Rotation Matrices

- A rotation matrix transforms its own rows onto the principal axes

$$\begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} r_{11} \\ r_{12} \\ r_{13} \end{bmatrix} = \ ? \qquad \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} r_{21} \\ r_{22} \\ r_{23} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} r_{31} \\ r_{32} \\ r_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

Rotation about Arbitrary Axis

- Any complex rotation can be described by a single rotation around an axis **n** by an angle θ [Proof?]
- Therefore, any complex rotation can be described by $[n_1, n_2, n_3]^T$ and θ
- Still 3 degrees of freedom as **n** can be taken to be a unit vector without any loss of generality

$$\sqrt{n_1^2 + n_2^2 + n_3^2} = 1$$

Rotation about Arbitrary Axis

- To rotate about an axis **n** by an angle θ

1. Set up rotations such that **n** rotates onto one of the principal axis [How?]
2. Rotate about that axis by θ
3. Undo the transformations in step 1

Rotation about Arbitrary Axis

- Question: Given an arbitrary 3D rotation matrix, how can we find out the axis **n** and the angle ϑ that represents this rotation?

$$\begin{bmatrix} -0.8256 & 0.40388 & -0.39404 \\ -0.20084 & -0.86294 & -0.46367 \\ -0.5273 & -0.30367 & 0.79356 \end{bmatrix}$$

Given R on the left, how can we tell n and θ?

Eigenvectors and Values of a Rotation Matrix

- 3D rotation matrix has eigenvalues of 1, $\cos\vartheta + i\sin\vartheta$ and $\cos\vartheta - i\sin\vartheta$ [Proof?]
- The eigenvector associated with the real eigenvalue represents the axis of rotation [proof?]