Computer Vision

(Summer Semester 2020)

Lecture 6, Part 1

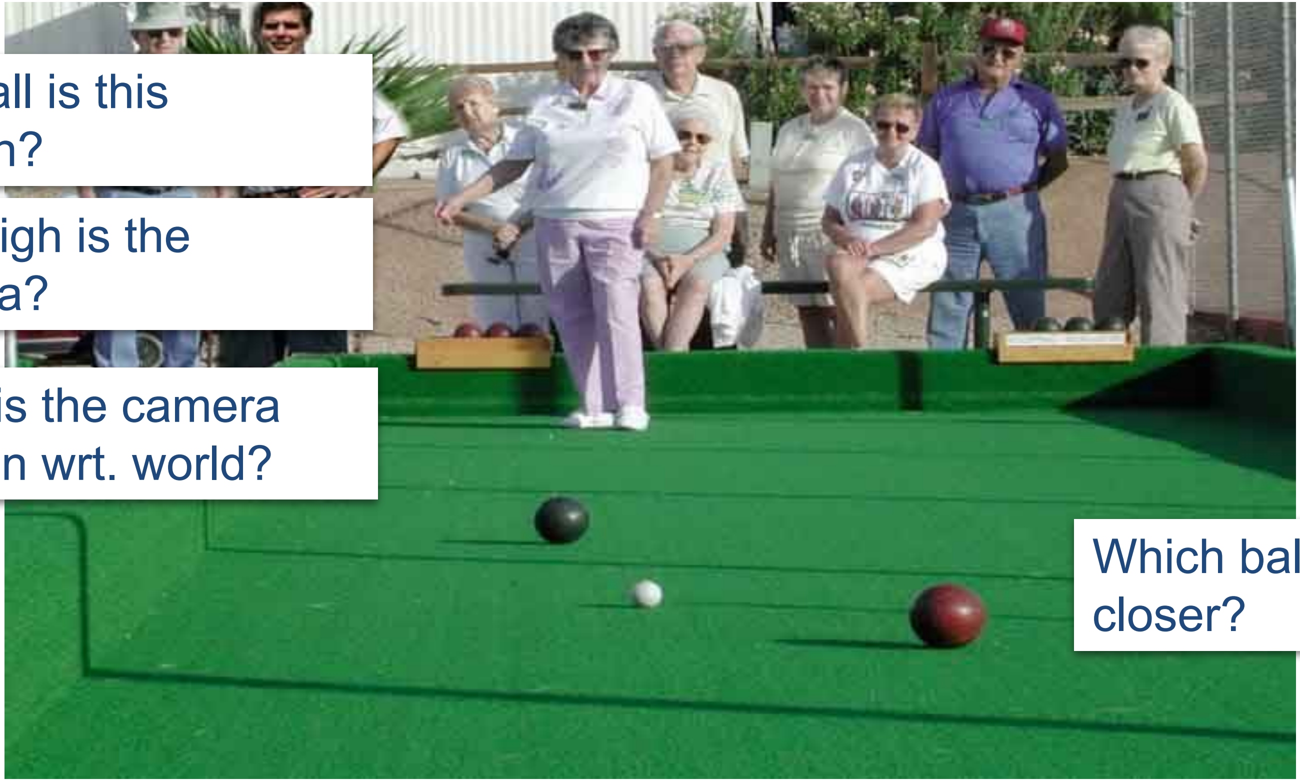
Camera Calibration

Camera Calibration

- **Transforms**
- **Least Squares Fitting**
- Least Squares Extrinsic and Intrinsic Parameter Fitting

- Note: The core of these slides stems from the class CSCI 1430: “Introduction to Computer Vision” by James Tompkin, Fall 2017, Brown University.

Cameras and World Geometry



How tall is this woman?

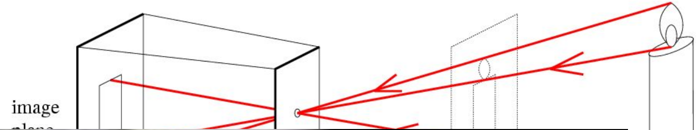
How high is the camera?

What is the camera rotation wrt. world?

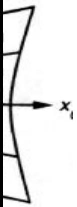
Which ball is closer?

J.Hays

Things to remember

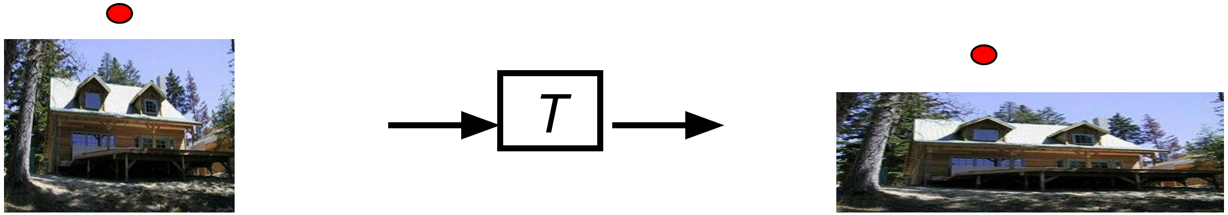


- Pinhole camera model and camera



- Aperture

Parametric (global) transformations



p = (x,y)

p’ = (x’,y’)

Transformation T is a coordinate-changing machine:

p’ = *T*(p)

What does it mean that *T* is global?

- *T* is the same for any point p

T can be described by just a few numbers (parameters)

For linear transformations, we can represent T as a matrix

p’ = **T**p

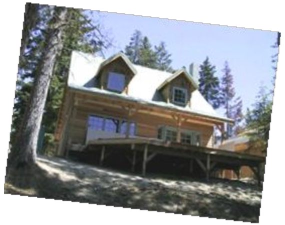
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \mathbf{T} \begin{bmatrix} x \\ y \end{bmatrix}$$

Common transformations

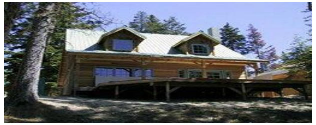
Original

Transformed

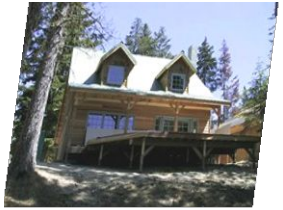
Translation



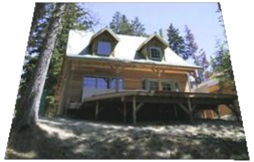
Rotation



Scaling



Affine



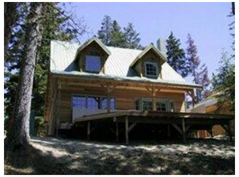
Perspective

Slide credit (next few slides):
A. Efros and/or S. Seitz

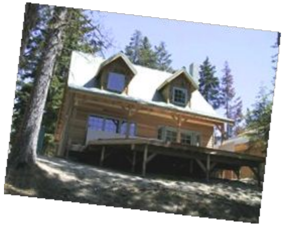
Common transformations

Original

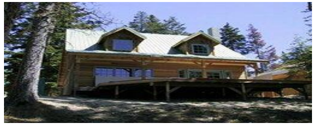
Transformed



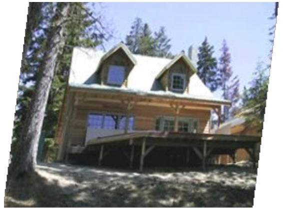
(a)



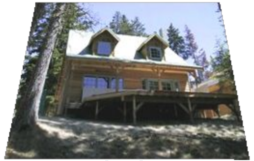
(b)



(c)



(d)

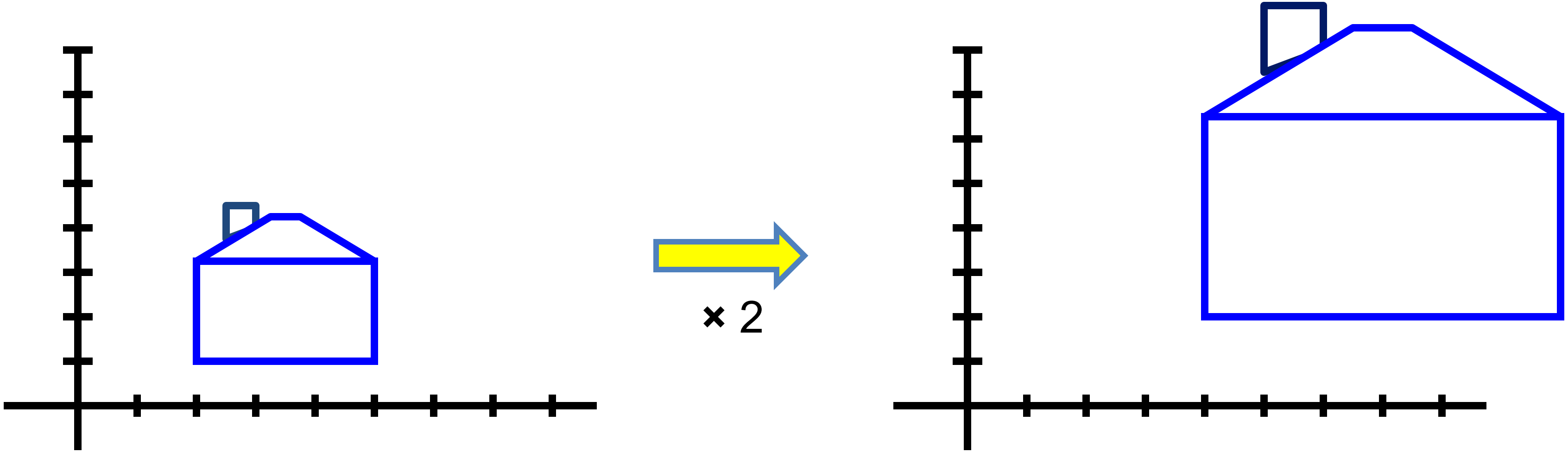


(e)

Slide credit (next few slides):
A. Efros and/or S. Seitz

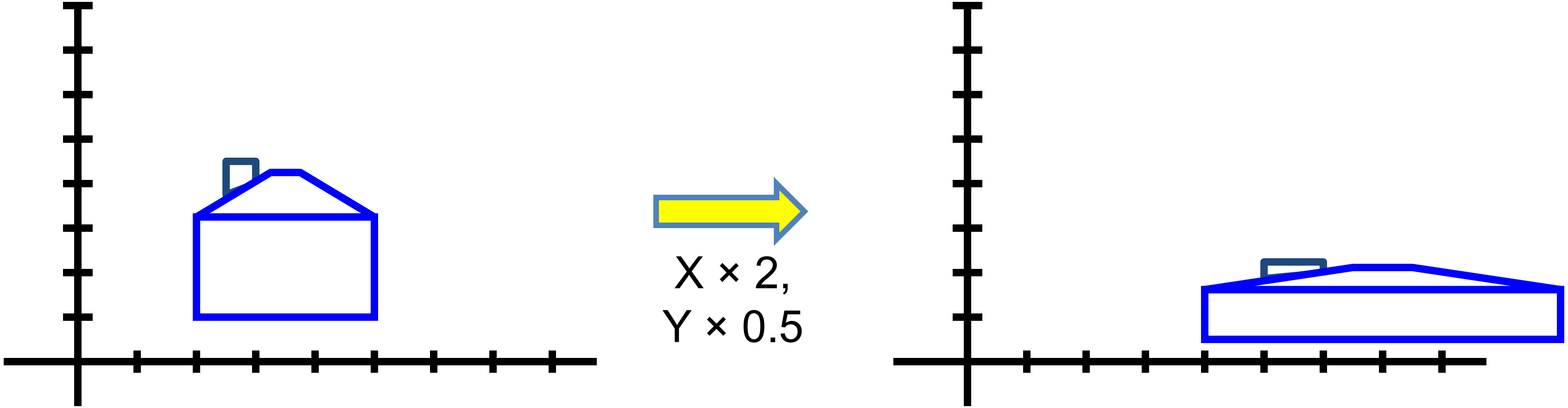
Scaling

- *Scaling* a coordinate means multiplying each of its components by a scalar
- *Uniform scaling* means this scalar is the same for all components:



Scaling

- *Non-uniform scaling*: different scalars per component:



Scaling

- Scaling operation:

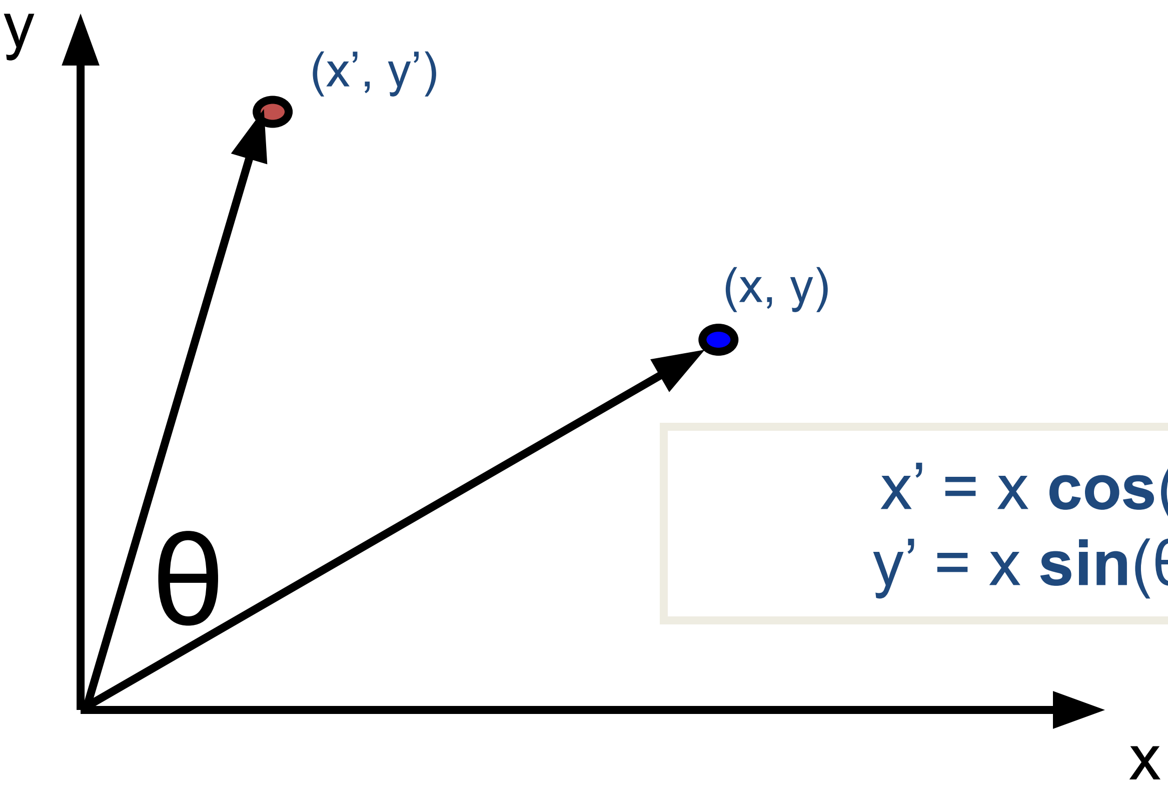
$$x' = ax$$

$$y' = by$$

- Or, in matrix form:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \underbrace{\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}}_{\textit{scaling matrix } S} \begin{bmatrix} x \\ y \end{bmatrix}$$

2-D Rotation



$$x' = x\,\mathbf{cos}(\theta) - y\,\mathbf{sin}(\theta)$$
$$y' = x\,\mathbf{sin}(\theta) + y\,\mathbf{cos}(\theta)$$

Basic 2D transformations

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Scale

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & \alpha_x \\ \alpha_y & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Shear

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\Theta & -\sin\Theta \\ \sin\Theta & \cos\Theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Rotate

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

Translate

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Affine

Affine is any combination of translation, scale, rotation, and shear

Affine Transformations

Affine transformations are combinations of

- Linear transformations, and
- Translations

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

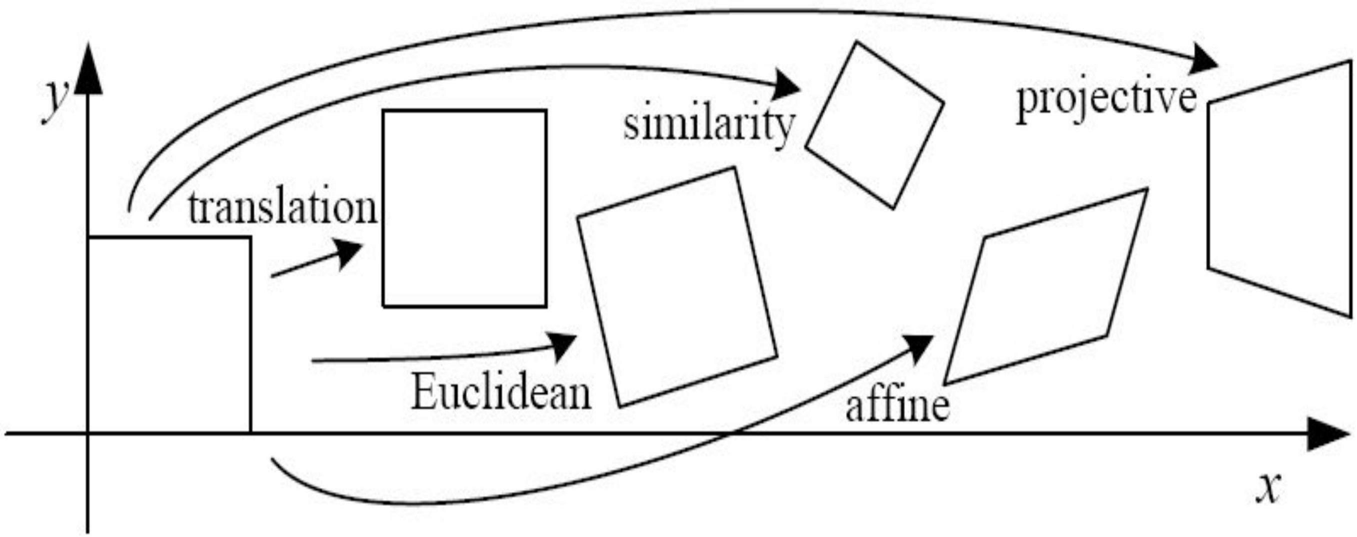
or

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Properties of affine transformations:

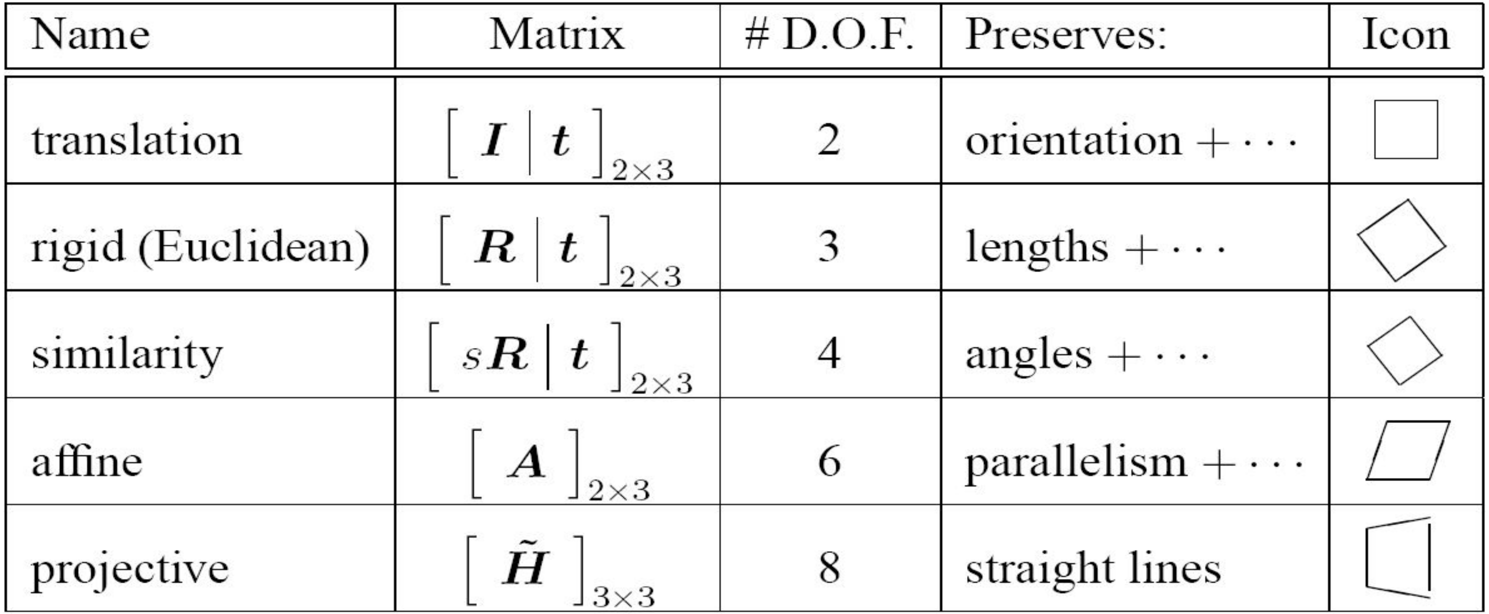
- Lines map to lines
- Parallel lines remain parallel
- Ratios are preserved
- Closed under composition

2D image transformations (reference table)



Szeliski

2D image transformations (reference table)



Name	Matrix	# D.O.F.	Preserves:	Icon
translation	$\begin{bmatrix} \boldsymbol{I} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	2	orientation $+\cdots$	□
rigid (Euclidean)	$\begin{bmatrix} \boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	3	lengths $+\cdots$	◇
similarity	$\begin{bmatrix} s\boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	4	angles $+\cdots$	◇
affine	$\begin{bmatrix} \boldsymbol{A} \end{bmatrix}_{2\times 3}$	6	parallelism $+\cdots$	▱
projective	$\begin{bmatrix} \tilde{\boldsymbol{H}} \end{bmatrix}_{3\times 3}$	8	straight lines	⏢

Szeliski

Projective Transformations

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

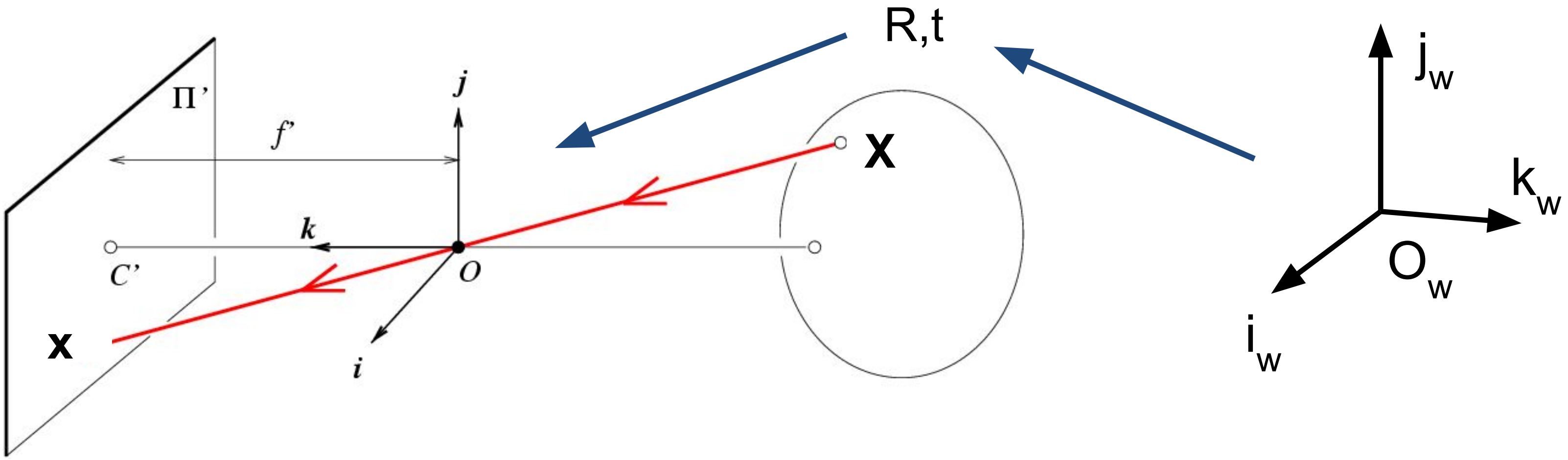
Projective transformations are combos of

- Affine transformations, and
- Projective warps

Properties of projective transformations:

- Lines map to lines
- Parallel lines do not necessarily remain parallel
- Ratios are not preserved
- Closed under composition
- Models change of basis
- Projective matrix is defined up to a scale (8 DOF)

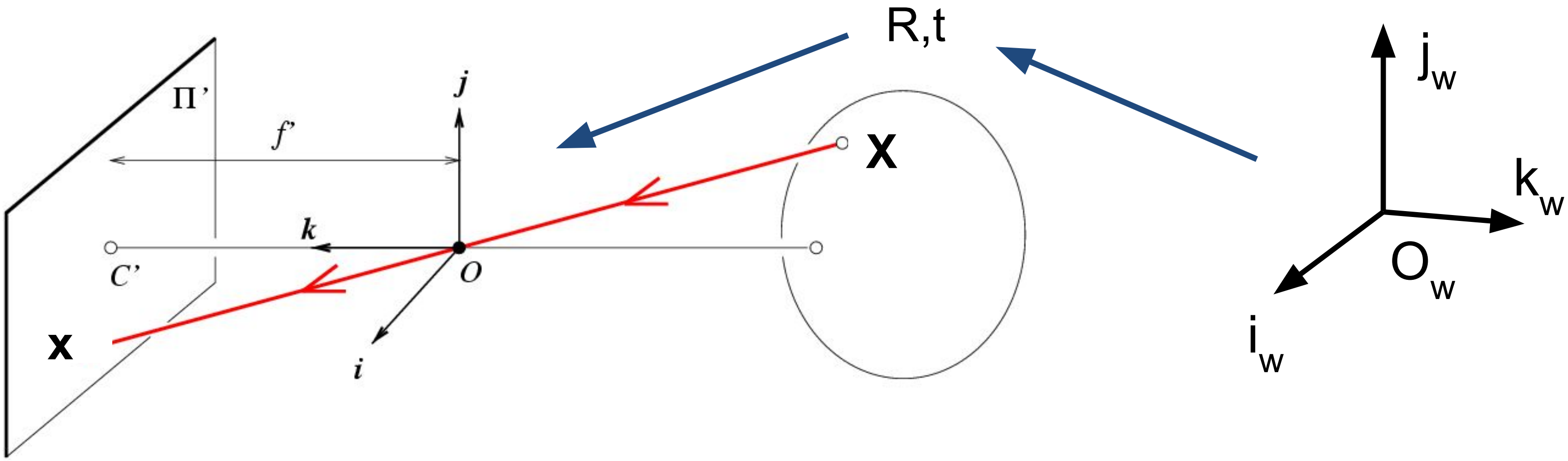
Camera (projection) matrix



$$\mathbf{x} = \mathbf{K}\underbrace{\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix}}_{\text{Extrinsic Matrix}}\mathbf{X}$$

x: Image Coordinates: (u,v,1)
K: Intrinsic Matrix (3x3)
R: Rotation (3x3)
t: Translation (3x1)
X: World Coordinates: (X,Y,Z,1)

Camera (projection) matrix



$$\mathbf{x} = \mathbf{K}\underbrace{\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix}}_{\text{Extrinsic Matrix}}\mathbf{X}$$

$$w\begin{bmatrix}u\\v\\1\end{bmatrix} = \begin{bmatrix}f_x & s & u_0\\0 & f_y & v_0\\0 & 0 & 1\end{bmatrix}\begin{bmatrix}r_{11} & r_{12} & r_{13} & t_x\\r_{21} & r_{22} & r_{23} & t_y\\r_{31} & r_{32} & r_{33} & t_z\end{bmatrix}\begin{bmatrix}x\\y\\z\\1\end{bmatrix}$$

Homogeneous coordinates

- Allows matrix form for 3D transformations

- Take rotation

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

- Now add translation

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix}$$

Homogeneous coordinates

- Take rotation

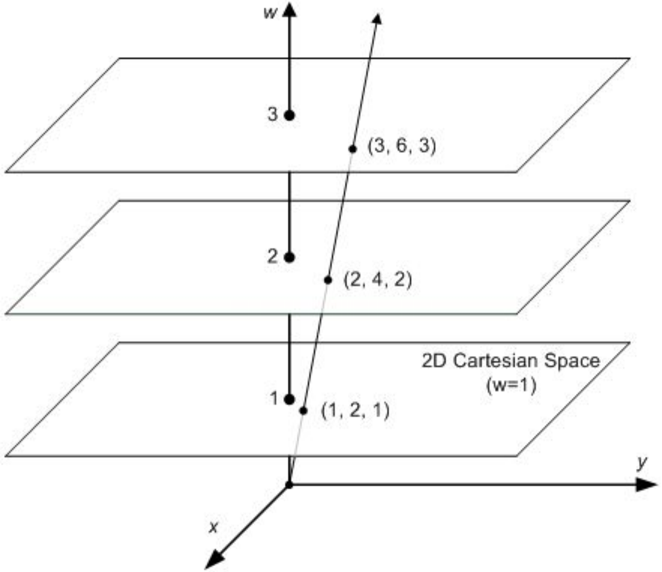
$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

- Now add translation with homogeneous coordinate

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Homogeneous coordinates

- Projective
- Point becomes a line



Song Ho Ahn

To homogeneous

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

From homogeneous

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w)$$

How to calibrate the camera? (also called “camera resectioning”)

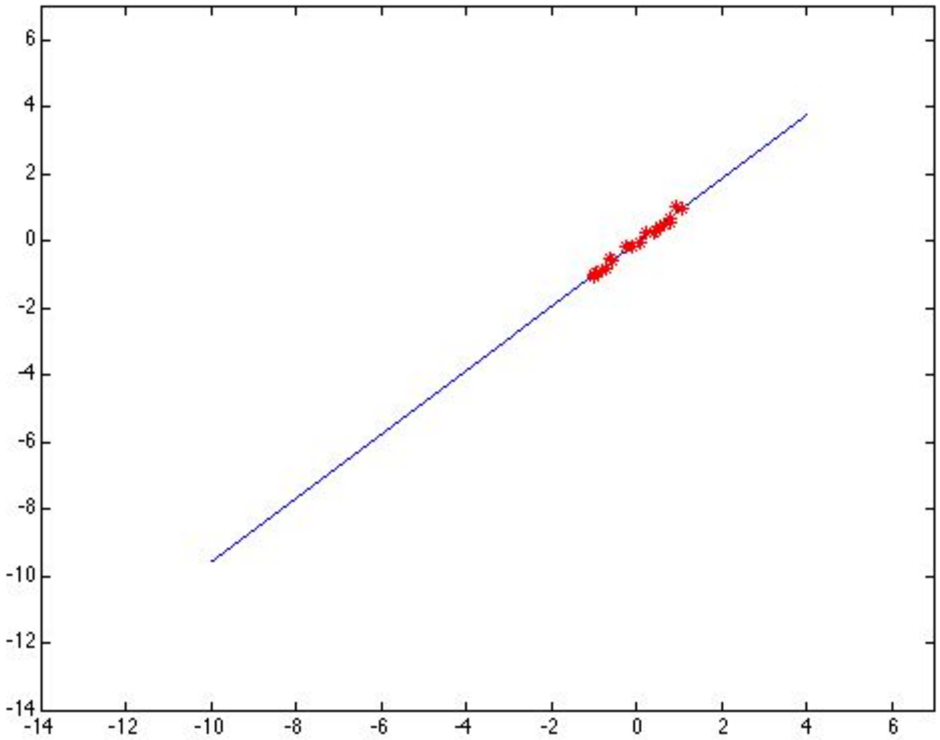
$$\mathbf{x} = \mathbf{K}\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix}\mathbf{X}$$

$$\mathbf{x} = \mathbf{M}\mathbf{X}$$

$$\begin{bmatrix} wu \\ wv \\ w \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Linear least-squares regression!

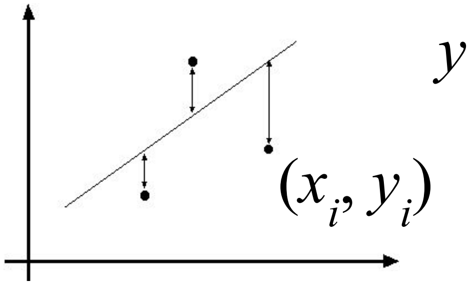
Simple example: Fitting a line



Least squares line fitting

- Data: $(x_1, y_1), \ldots, (x_n, y_n)$
- Line equation: $y_i = m\,x_i + b$
- Find (m, b) to minimize

$$\boxed{E = \sum_{i=1}^{n} (y_i - mx_i - b)^2}$$



$y{=}mx{+}b$

$$E = \sum_{i=1}^{n}\left(\begin{bmatrix} x_i & 1 \end{bmatrix}\begin{bmatrix} m \\ b \end{bmatrix} - y_i\right)^2 = \left\|\begin{bmatrix} x_1 & 1 \\ \square & \square \\ x_n & 1 \end{bmatrix}\begin{bmatrix} m \\ b \end{bmatrix} - \begin{bmatrix} y_1 \\ \square \\ y_n \end{bmatrix}\right\|^2 = \|\mathbf{Ap} - \mathbf{y}\|^2$$

$$= \mathbf{y}^T\mathbf{y} - 2(\mathbf{Ap})^T\mathbf{y} + (\mathbf{Ap})^T(\mathbf{Ap})$$

$$\frac{dE}{dp} = 2\mathbf{A}^T\mathbf{Ap} - 2\mathbf{A}^T\mathbf{y} = 0$$

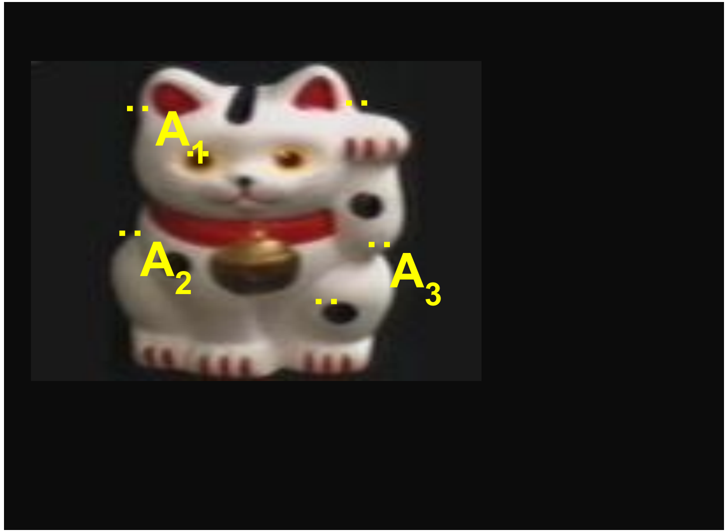
Matlab: `p = A \ y;`

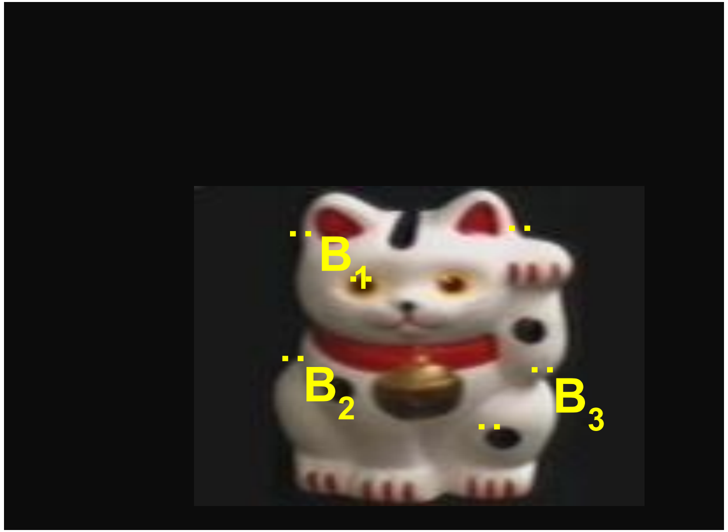
$$\mathbf{A}^T\mathbf{Ap} = \mathbf{A}^T\mathbf{y} \Rightarrow \mathbf{p} = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{y}$$

(Closed form solution)

S. Lazebnik

Example: solving for translation



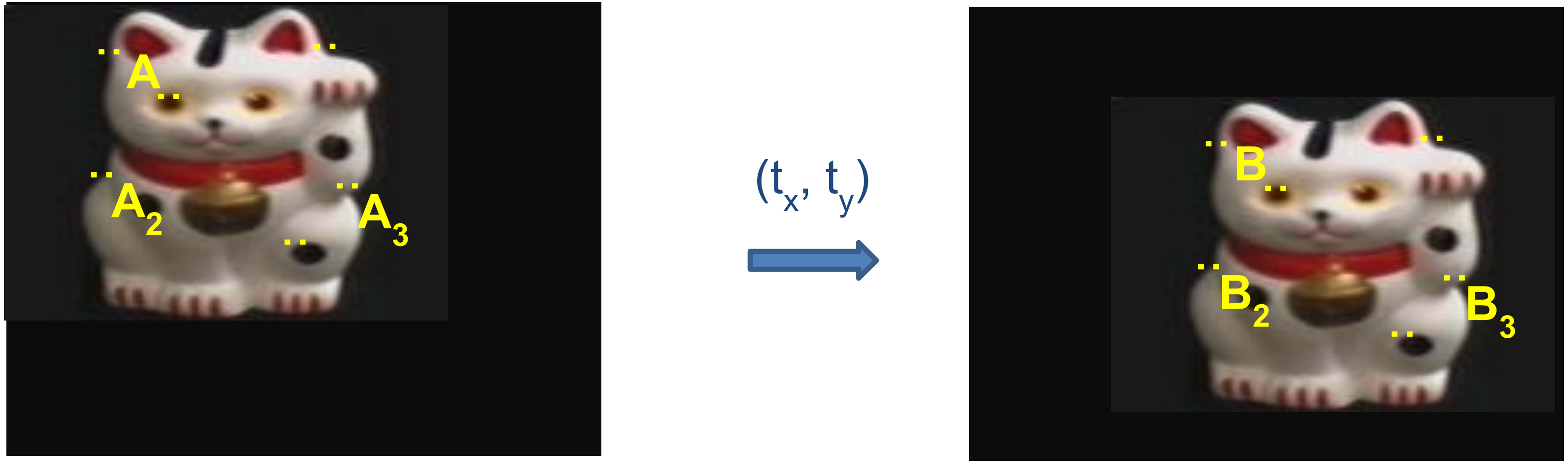


Given matched points in {A} and {B}, estimate the translation of the object

$$\begin{bmatrix} x_i^B \\ y_i^B \end{bmatrix} = \begin{bmatrix} x_i^A \\ y_i^A \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

Example: solving for translation

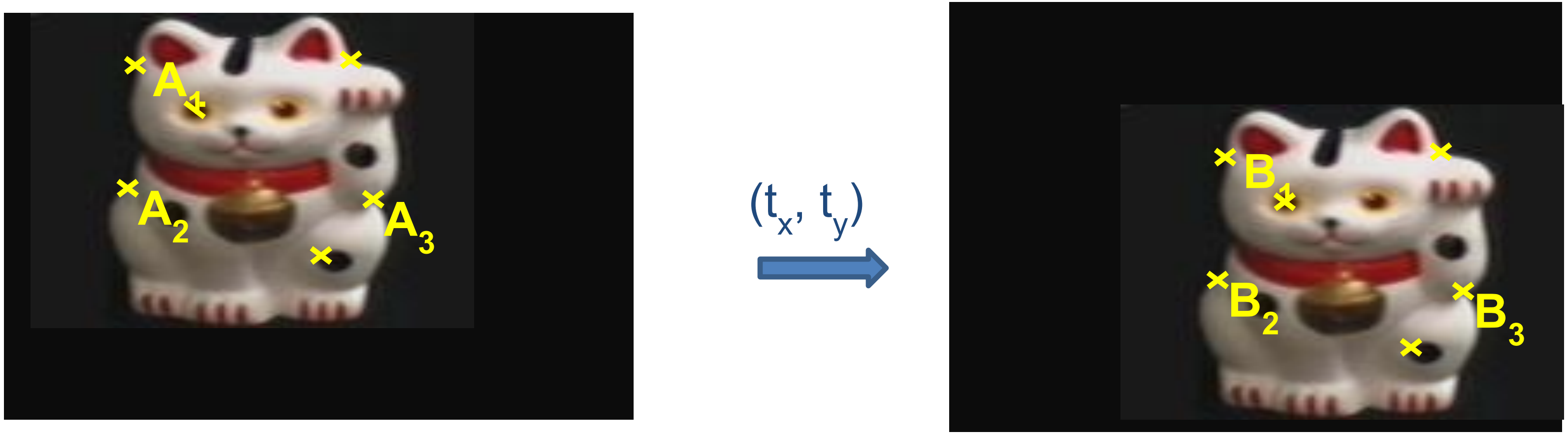
Least squares setup



$$\begin{bmatrix} x_i^B \\ y_i^B \end{bmatrix} = \begin{bmatrix} x_i^A \\ y_i^A \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ \vdots & \vdots \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} t_x \\ t_y \end{bmatrix} = \begin{bmatrix} x_1^B - x_1^A \\ y_1^B - y_1^A \\ \vdots \\ x_n^B - x_n^A \\ y_n^B - y_n^A \end{bmatrix}$$

Example: solving for translation



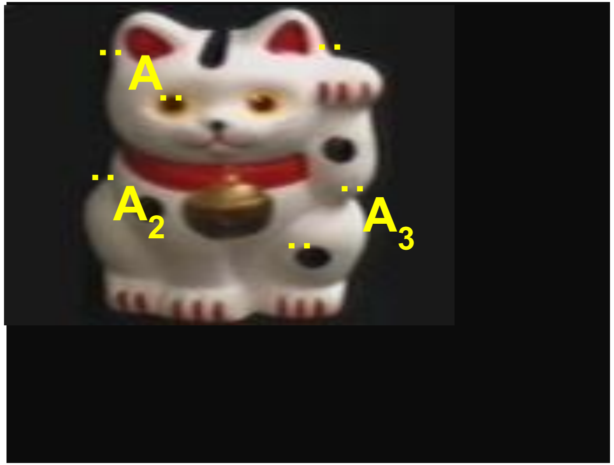
$$\begin{bmatrix} x_i^B \\ y_i^B \end{bmatrix} = \begin{bmatrix} x_i^A \\ y_i^A \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

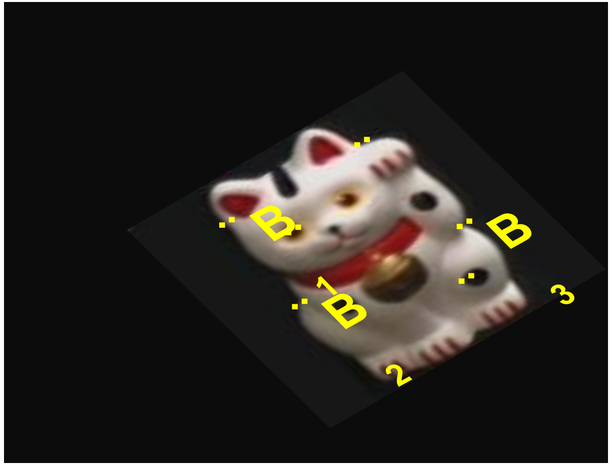
Least squares solution

1. Write down objective function
2. Derived solution
   a) Compute derivative
   b) Compute solution
3. Computational solution
   a) Write in form Ax=p
   b) Solve using closed-form solution

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ \square & \square \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} t_x \\ t_y \end{bmatrix} = \begin{bmatrix} x_1^B - x_1^A \\ y_1^B - y_1^A \\ \square \\ x_n^B - x_n^A \\ y_n^B - y_n^A \end{bmatrix}$$

Example: discovering rot/trans/scale

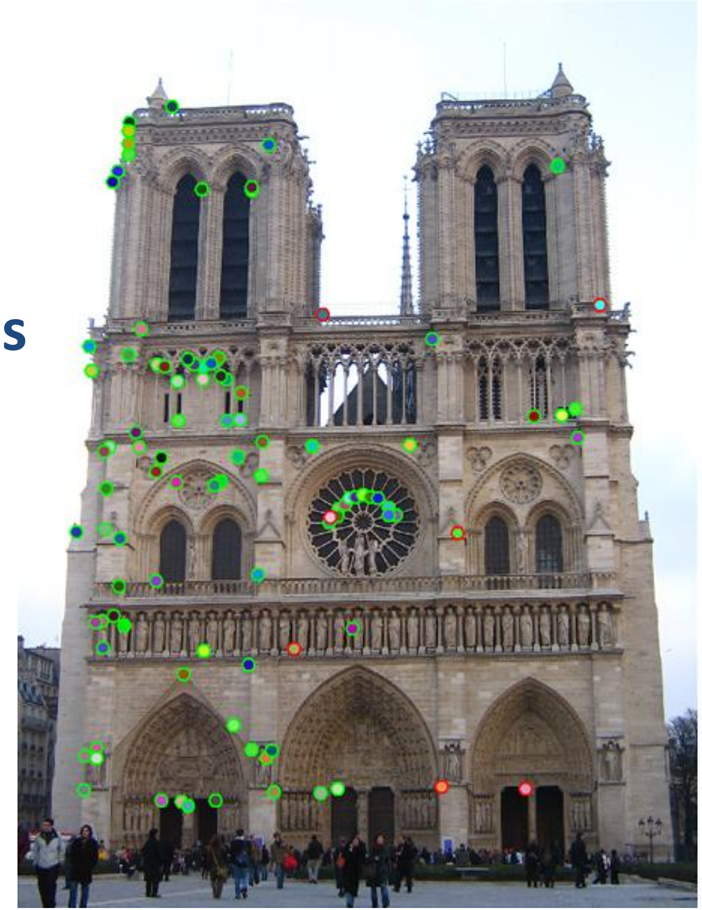


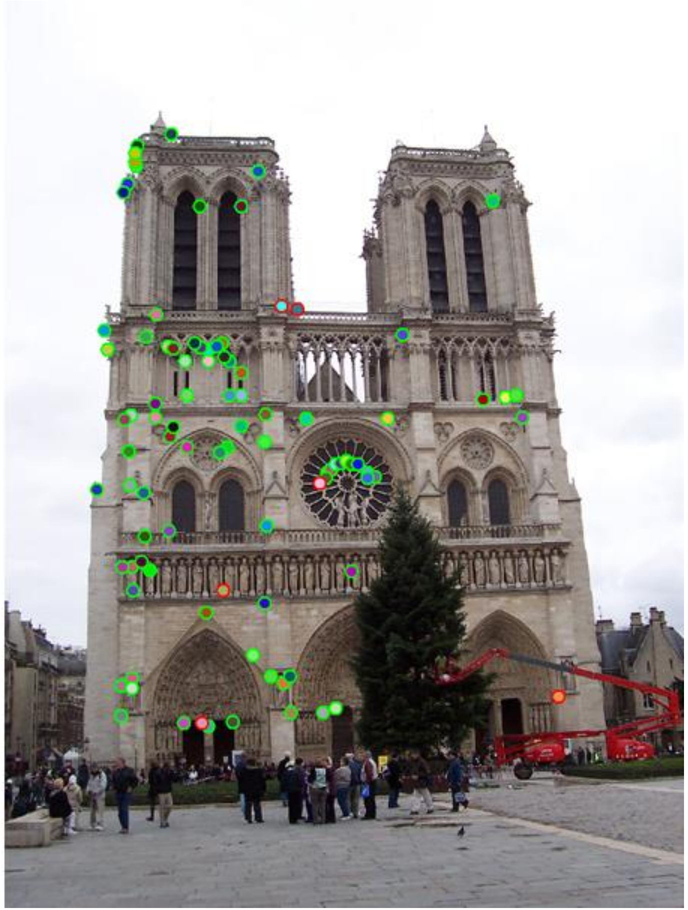


Given matched points in {A} and {B}, estimate the transformation matrix

$$\begin{bmatrix} x_i^B \\ y_i^B \end{bmatrix} = \mathrm{T} \begin{bmatrix} x_i^A \\ y_i^A \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} \qquad \mathrm{T} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

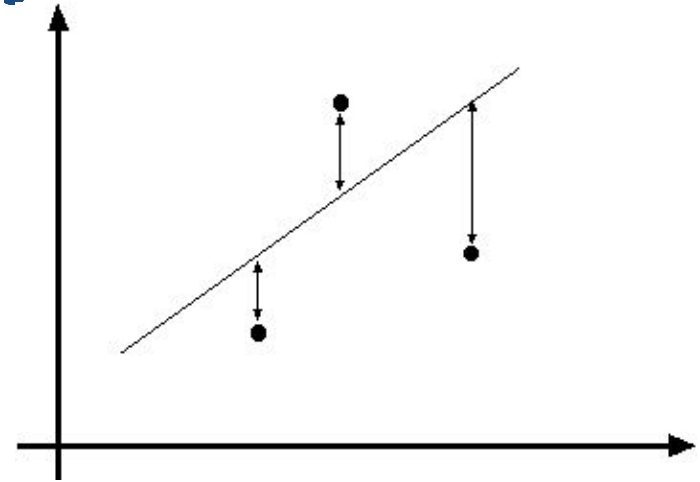
Are these transformations enough?





Problem with “vertical” least squares

- Not rotation-invariant
- Fails completely for vertical lines



S. Lazebnik

Total least squares

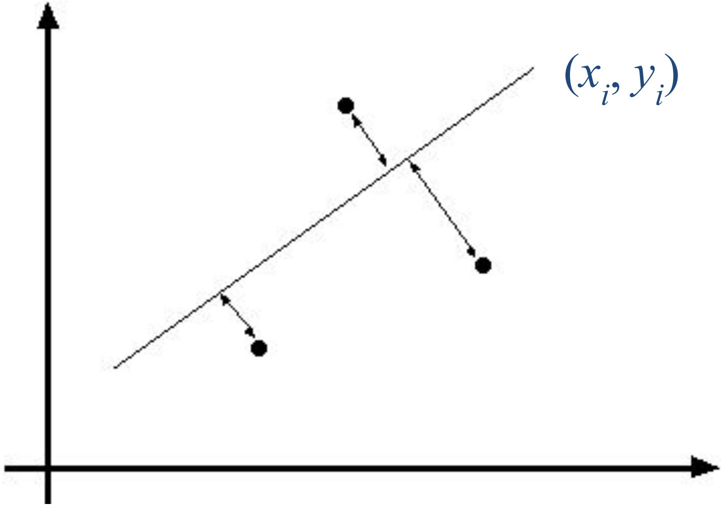
If (a^2+b^2=1) then

Distance between point (x_i, y_i) is

$$|ax_i + by_i + c|$$

$ax+by+c=0$

Unit normal: $N=(a, b)$



proof: Point-Line Distance--2-Dimensional -- from Wolfram MathWorld

S. Lazebnik

Total least squares

$ax+by+c=0$

Unit normal: $N=(a, b)$

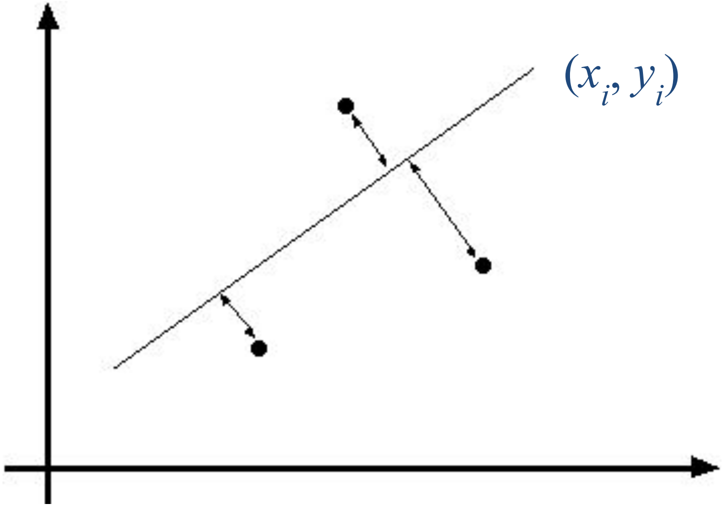
If ($a^2+b^2=1$) then

Distance between point (x_i, y_i) is

$$|ax_i + by_i + c|$$

Find (a, b, c) to minimize the sum of squared perpendicular distances

$$\boxed{E = \sum_{i=1}^{n}(ax_i + by_i + c)^2}$$



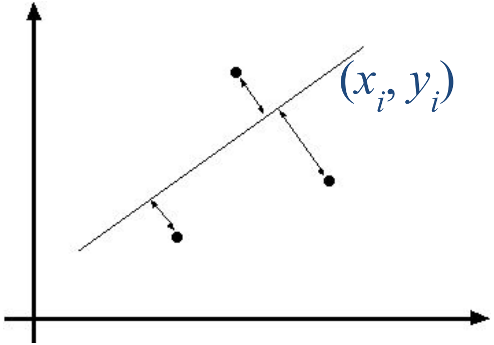
proof: Point-Line Distance--2-Dimensional -- from Wolfram MathWorld

Total least squares

Find (a, b, c) to minimize the sum of squared perpendicular distances

$ax+by+c=0$

Unit normal: $N=(a, b)$



$$\boxed{E=\sum_{i=1}^{n}(ax_i+by_i+c)^2}$$

$$c=-\frac{a}{n}\sum_{i=1}^{n}x_i-\frac{b}{n}\sum_{i=1}^{n}y_i=-a\bar{x}-b\bar{y}$$

$$\frac{\partial E}{\partial c}=\sum_{i=1}^{n}2(ax_i+by_i+c)=0$$

$$E=\sum_{i=1}^{n}(a(x_i-\bar{x})+b(y_i-\bar{y}))^2=\left\|\begin{bmatrix}x_1-\bar{x} & y_1-\bar{y}\\ \square & \square\\ x_n-\bar{x} & y_n-\bar{y}\end{bmatrix}\begin{bmatrix}a\\ b\end{bmatrix}\right\|^2=\mathbf{p}^T\mathbf{A}^T\mathbf{A}\mathbf{p}$$

$$\text{minimize}\,\mathbf{p}^T\mathbf{A}^T\mathbf{A}\mathbf{p}\quad \text{s.t. } \mathbf{p}^T\mathbf{p}=1 \quad\Rightarrow\quad \text{minimize}\frac{\mathbf{p}^T\mathbf{A}^T\mathbf{A}\mathbf{p}}{\mathbf{p}^T\mathbf{p}}$$

Solution is eigenvector corresponding to smallest eigenvalue of A^TA

See details on Raleigh Quotient: http://en.wikipedia.org/wiki/Rayleigh_quotient

Recap: Two Common Optimization Problems

Problem statement	Solution
$\text{minimize } \left\Vert \mathbf{Ax}-\mathbf{b}\right\Vert^2$	$\mathbf{x}=\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{b}$
$\text{least squares solution to } \mathbf{Ax}=\mathbf{b}$	$\mathbf{x}=\mathbf{A}\setminus\mathbf{b}$ (matlab)

Problem statement	Solution
$\text{minimize } \mathbf{x}^T\mathbf{A}^T\mathbf{A}\mathbf{x} \text{ s.t. } \mathbf{x}^T\mathbf{x}=1$	
$\text{minimize} \dfrac{\mathbf{x}^T\mathbf{A}^T\mathbf{A}\mathbf{x}}{\mathbf{x}^T\mathbf{x}}$	$[\mathbf{v},\lambda]=\text{eig}(\mathbf{A}^T\mathbf{A})$ $\lambda_1<\lambda_{2..n}:\mathbf{x}=\mathbf{v}_1$
$\text{non-trivial lsq solution to } \mathbf{Ax}=0$	