

# **Camera Model**

# **Linear Least Squares**

# **Triangulation**

David Arnon

# Camera Model

# Kitti Cameras

- Left Camera:  $\begin{bmatrix} 719 & 0 & 607 & 0 \\ 0 & 719 & 185 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$
- Right Camera:  $\begin{bmatrix} 719 & 0 & 607 & -386 \\ 0 & 719 & 185 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$
- What are the coordinate systems?

# Kitti Cameras

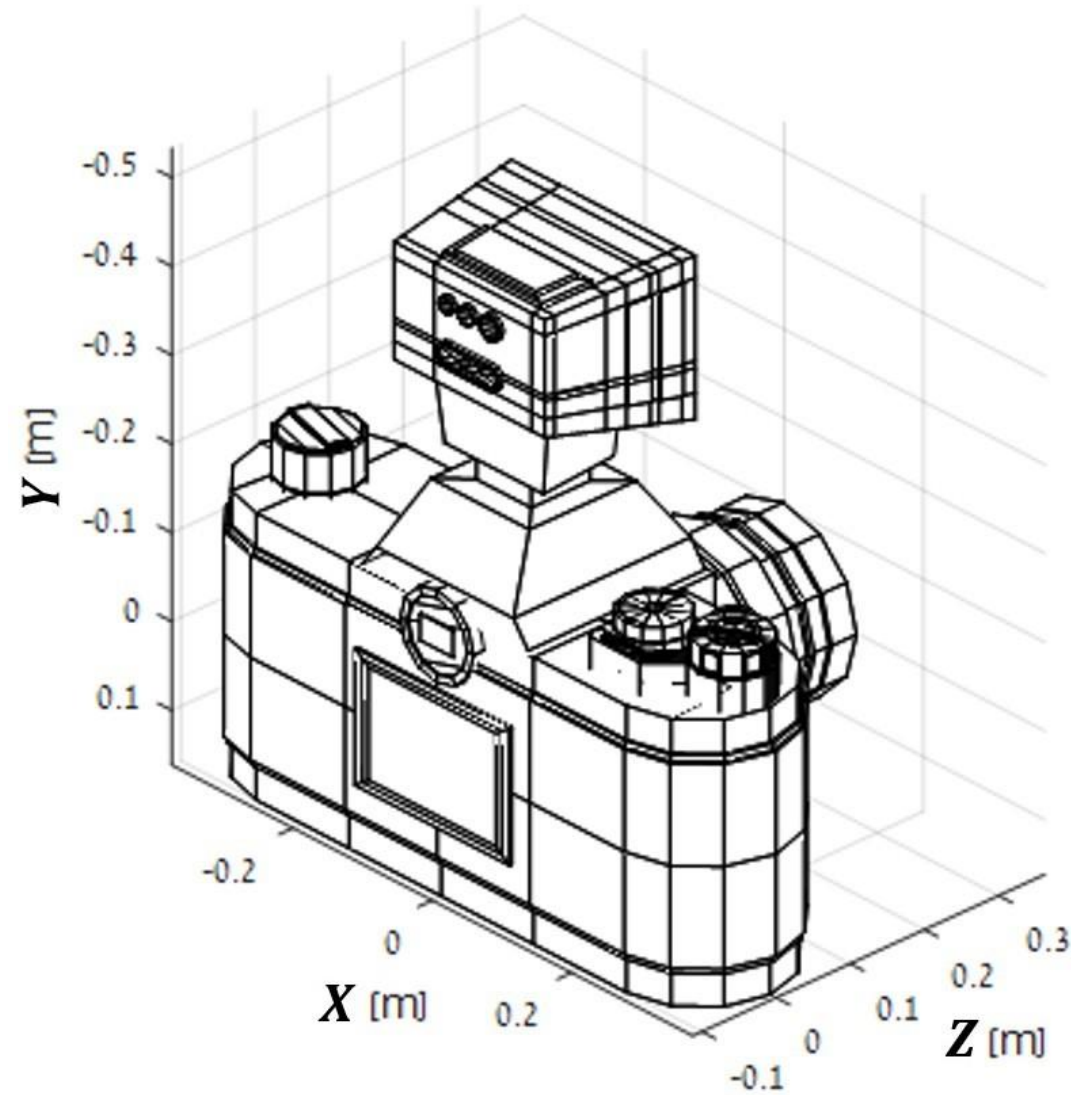


```
print('left cam keypoint:', kp1[0].pt)
print('right cam keypoint:', kp2[0].pt)
```

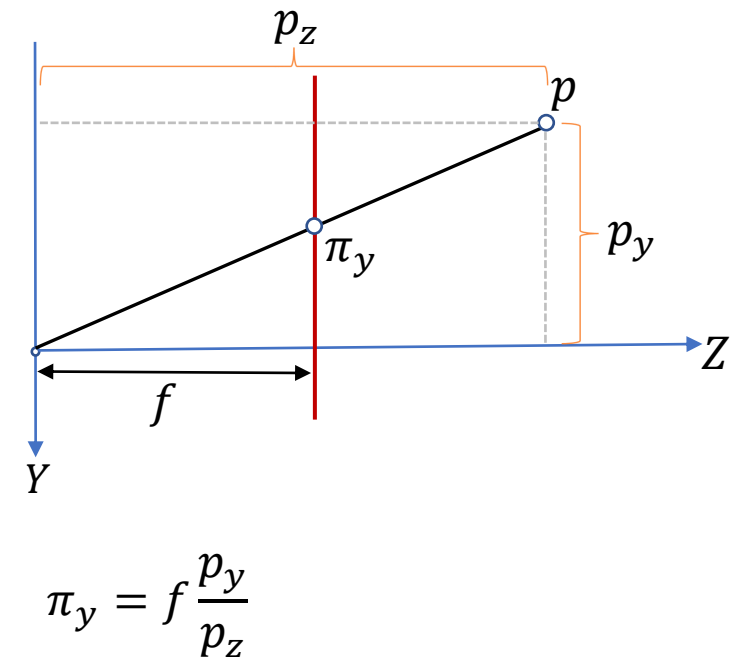
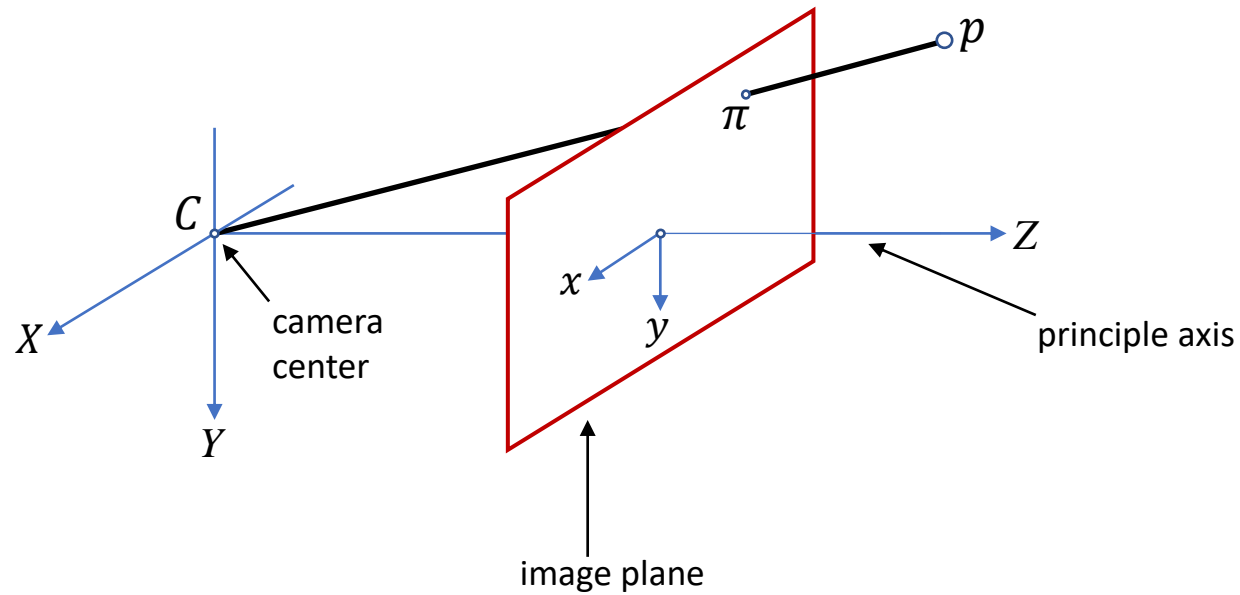
```
k, m1, m2 = read_cameras(img_dir + 'calib.txt')
p4d = cv2.triangulatePoints(k@m1, k@m2, kp1[0].pt, kp2[0].pt)
p3d = p4d[:3] / p4d[3]
print('3D point:', p3d.T)
```

```
left cam keypoint: (922.7208251953125, 30.694913864135742)
right cam keypoint: (907.992919921875, 29.992298126220703)
3D point: [[11.7 -5.57 25.79]]
```

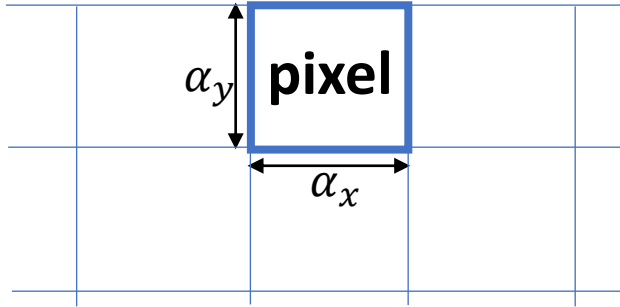
# Camera Coordinates



# Camera Coordinates

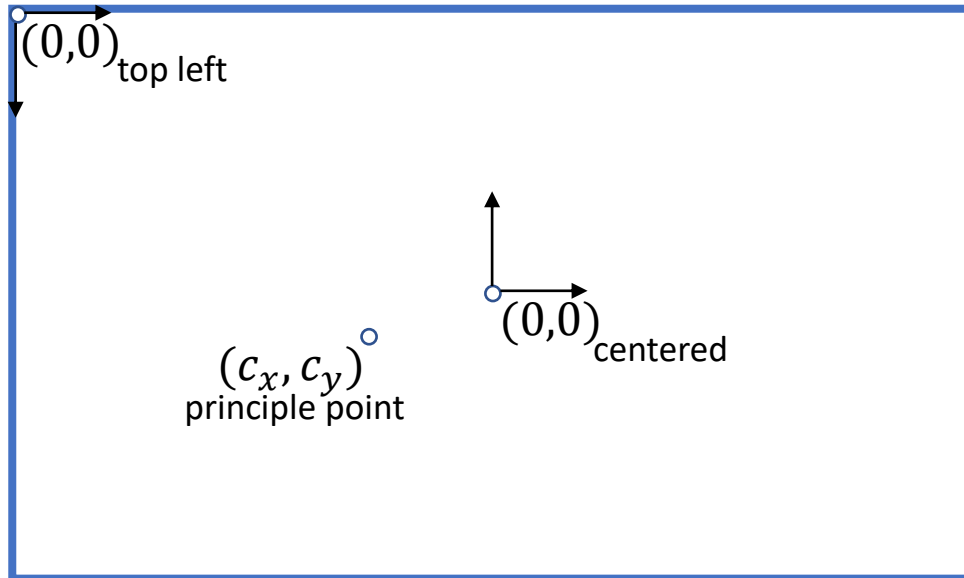


# Camera Model



$$f_x = \frac{f}{\alpha_x}$$

$$f_y = \frac{f}{\alpha_y}$$



$$\pi_x = f_x \frac{x}{z} + c_x$$

$$\pi_y = f_y \frac{y}{z} + c_y$$

# Camera Model

- Projection: 
$$\underbrace{\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}}_K \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} f_x x + c_x z \\ f_y y + c_y z \\ z \end{bmatrix} \propto \begin{bmatrix} f_x \frac{x}{z} + c_x \\ f_y \frac{y}{z} + c_y \\ 1 \end{bmatrix}$$



# Euler's theorem (1776)

**Theorema.** Quomodocunque sphaera circa centrum suum conuertatur, semper assignari potest diameter, cuius directio in situ translato conueniat cum situ initiali.

- Two Euclidean coordinate systems differ by rotation and translation.

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \phi & -\sin \phi \\ 0 & \sin \phi & \cos \phi \end{bmatrix}$$

$$R_y(\theta) = \begin{bmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{bmatrix}$$

$$R_z(\psi) = \begin{bmatrix} \cos \psi & -\sin \psi & 0 \\ \sin \psi & \cos \psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- What is the inverse?

$$R = R_z(\psi)R_y(\theta)R_x(\phi)$$



# Camera Model

- Coordinate change:  $\begin{bmatrix} x \\ y \\ z \end{bmatrix} = [R|t] \begin{bmatrix} w \\ 1 \end{bmatrix} = Rw + t, \quad w = \begin{bmatrix} w_x \\ w_y \\ w_z \end{bmatrix}$

- Projection:  $K[R|t] \begin{bmatrix} w \\ 1 \end{bmatrix} = K \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \underbrace{\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}}_K \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

# Kitti Cameras

- Left Camera: 
$$\begin{bmatrix} 707 & 0 & 602 \\ 0 & 707 & 183 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

- Right Camera: 
$$\begin{bmatrix} 707 & 0 & 602 \\ 0 & 707 & 183 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -0.54 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

- Find Location:

$$0 = [I|t] \begin{bmatrix} c \\ 1 \end{bmatrix}$$

$$0 = c + t$$

$$c = -t = \begin{bmatrix} 0.54 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} 707 & 0 & 602 & 0 \\ 0 & 707 & 183 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 707 & 0 & 602 & -380 \\ 0 & 707 & 183 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

# Homography

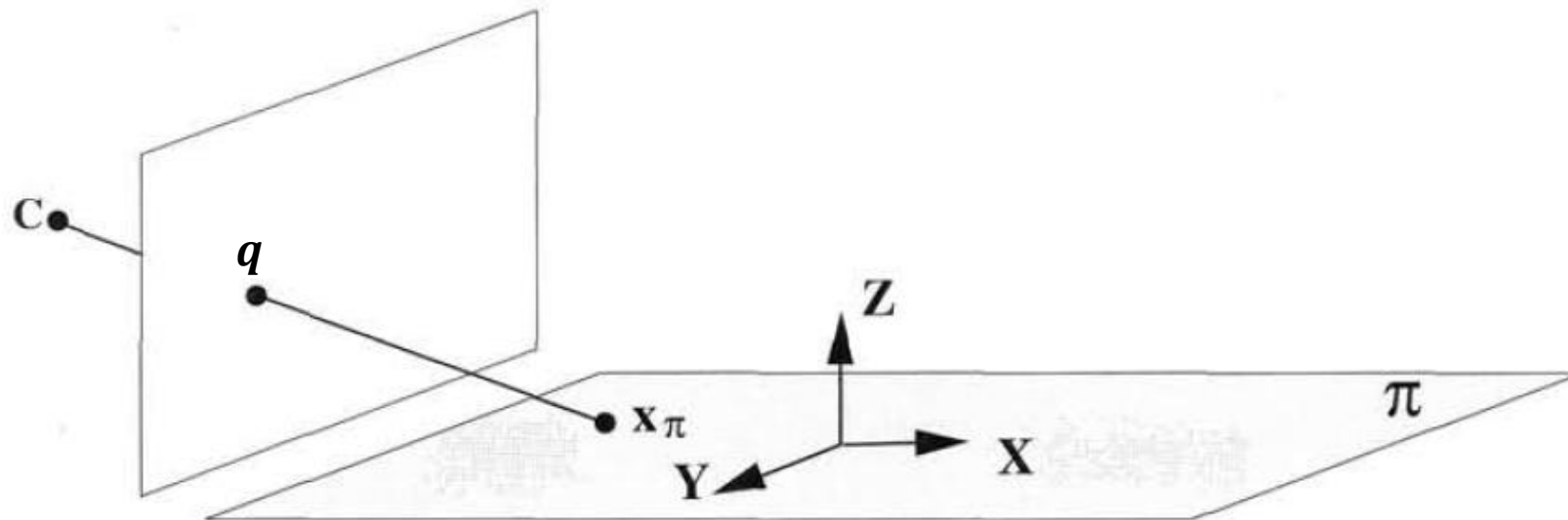
- Projection of a plane is an Homography:



# Homography

- Projection of a plane is an Homography:

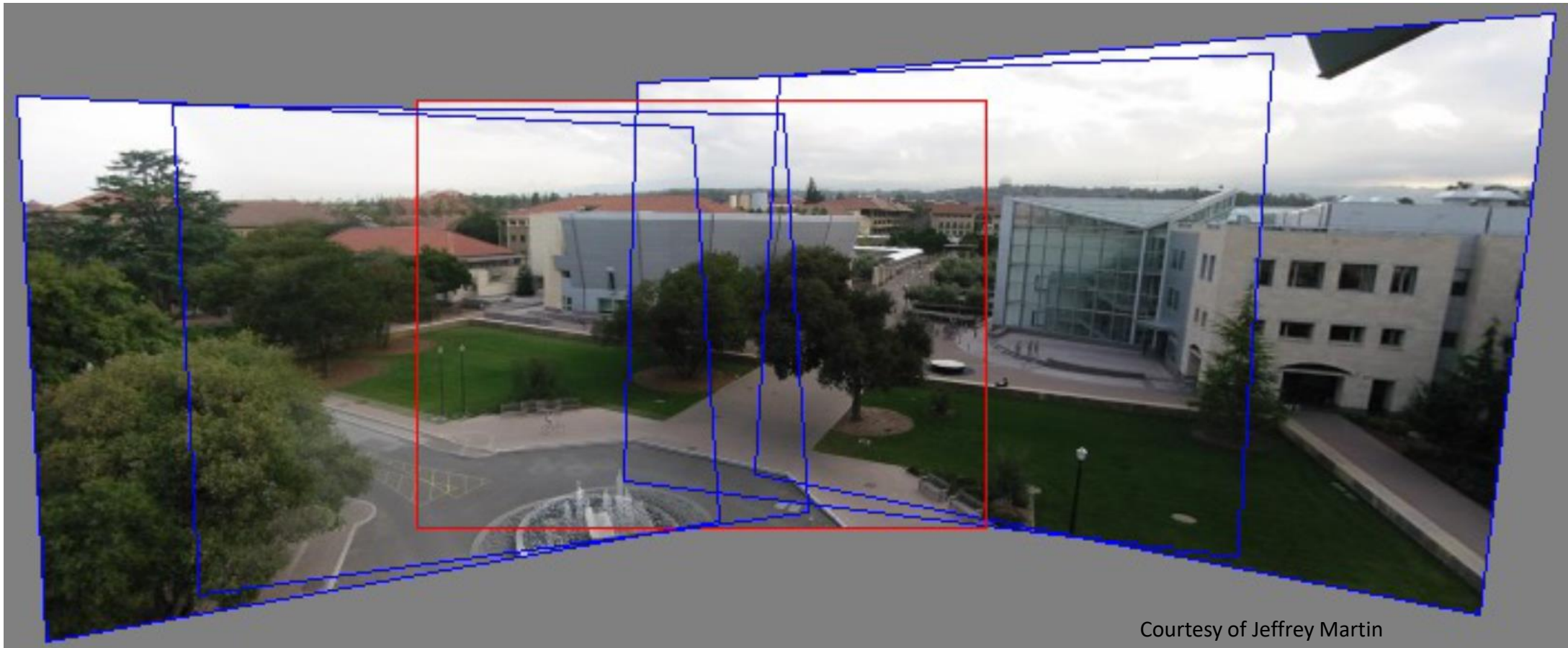
$$q \propto P\mathbf{X} = \begin{bmatrix} | & | & | & | \\ p_1 & p_2 & p_3 & p_4 \\ | & | & | & | \end{bmatrix} \begin{bmatrix} x \\ y \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} | & | & | \\ p_1 & p_2 & p_4 \\ | & | & | \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$





# Homography

- Rotated cameras are related by an Homography:



Courtesy of Jeffrey Martin

# Homography

- Rotated cameras are related by an Homography:

$$\text{w.l.o.g.} \quad C_1 = K_1[I|0], \quad C_2 = K_2[R|0].$$

$$q_1 \propto C_1 \begin{bmatrix} X \\ 1 \end{bmatrix} = K_1 X \quad \Rightarrow \quad X \propto K_1^{-1} q_1$$

$$q_2 \propto C_2 \begin{bmatrix} X \\ 1 \end{bmatrix} = K_2 R X \propto K_2 R K_1^{-1} q_1$$

# Horizon

- The line at infinity is projected to a straight line:

$$q \propto \begin{bmatrix} | & | & | & | \\ p_1 & p_2 & p_3 & p_4 \\ | & | & | & | \end{bmatrix} \begin{bmatrix} x \\ y \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} | & | \\ p_1 & p_2 \\ | & | \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = xp_1 + yp_2$$

- $l = p_1 \times p_2$





# Line

- A line is projected to a line:

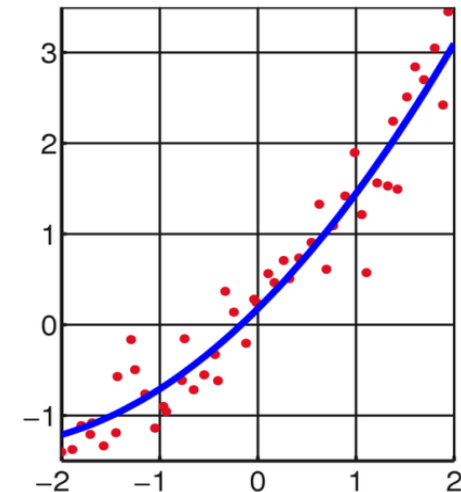
$$P \begin{bmatrix} a + td \\ 1 \end{bmatrix} = P \begin{bmatrix} a \\ 1 \end{bmatrix} + tP \begin{bmatrix} d \\ 0 \end{bmatrix}$$

- $l = P \begin{bmatrix} a \\ 1 \end{bmatrix} \times P \begin{bmatrix} d \\ 0 \end{bmatrix}$

# Linear Least Squares

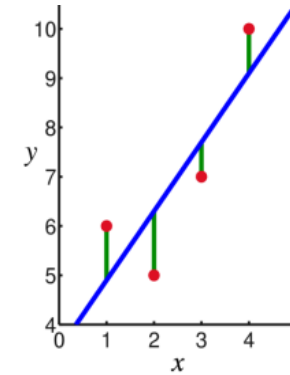
# Least Squares

- First developed by Gauss in 1795
- Standard approach to the approximate solution of overdetermined systems
- Used regularly for data fitting



# Least Squares

- Minimizes the sum of squares of the errors made in solving every equation
  - $L_2$  norm
- Same as maximum likelihood if the errors have a normal distribution
- Non-linear least squares is usually solved by iterative refinement and requires an initial solution
- Linear least squares has a closed-form solution! 😊



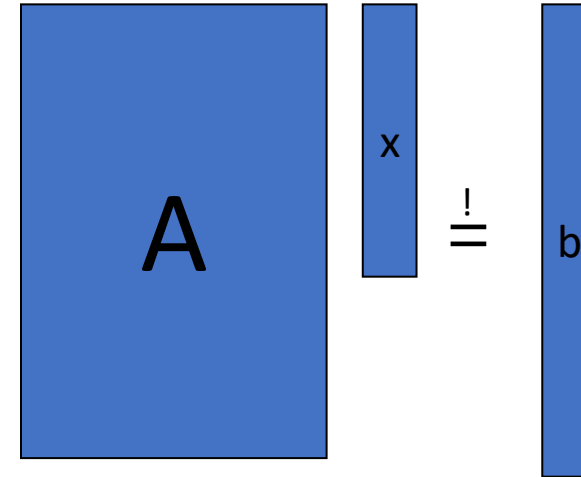
# Linear Least Squares

## Problem Statement

- $\operatorname{argmin}_x \|Ax - b\|_2$

- $A \in M_{m \times n} \quad m \geq n$

- $x \in M_{n \times 1}$



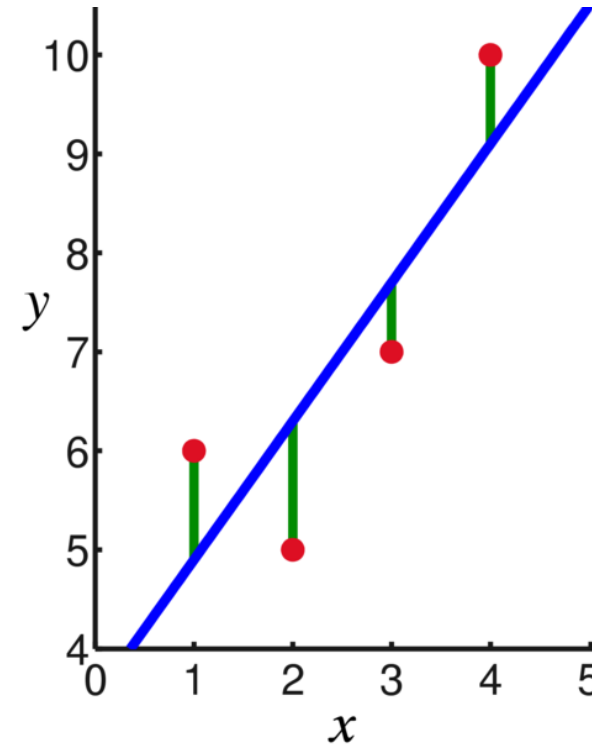
- $\operatorname{argmin}_x \|Ax\|_2 \quad s.t. \quad \|x\|_2 = 1$

# Linear Least Squares

## Example - Line

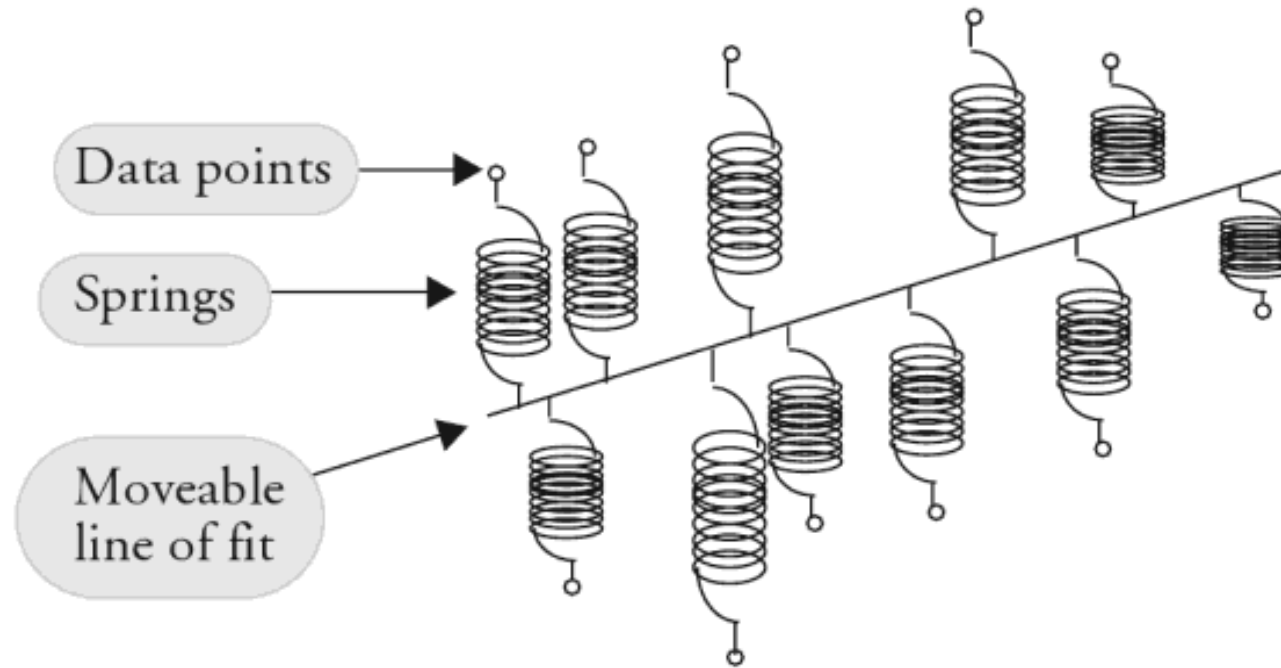
$$\begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ x_3 & 1 \\ x_4 & 1 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 \\ 2 & 1 \\ 3 & 1 \\ 4 & 1 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} 6 \\ 5 \\ 7 \\ 10 \end{bmatrix}$$



# Linear Least Squares

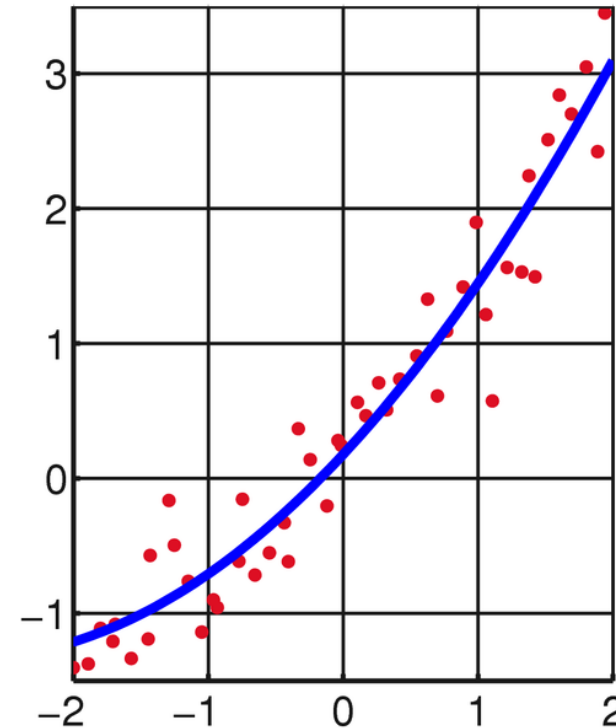
## Example - Line



# Linear Least Squares

## Example – Quadratic Function

$$\begin{bmatrix} x_1^2 & x_1 & 1 \\ x_2^2 & x_2 & 1 \\ x_3^2 & x_3 & 1 \\ \vdots & \vdots & \vdots \\ x_n^2 & x_n & 1 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix}$$





# Linear Least Squares Solution

- $\operatorname{argmin}_x \|Ax - b\|_2$
- The solution is  $x^* = A^\dagger b$
- $A^\dagger \doteq (A^T A)^{-1} A^T$ 
  - $A^\dagger$  is the pseudo-inverse matrix of  $A$
- For large problems we can solve  $A^T Ax = A^T b$  instead of inverting  $A^T A$ . (Cholesky decomposition)

# Linear Least Squares




## Pseudo-Inverse Proof

- $\operatorname{argmin}_x \|A \cdot x - b\|_2 =$   
 $\operatorname{argmin}_x (Ax - b)^\top \cdot (Ax - b) =$   
 $\operatorname{argmin}_x (x^\top A^\top - b^\top) \cdot (Ax - b) =$   
 $\operatorname{argmin}_x (x^\top A^\top A x - b^\top A x - x^\top A^\top b + b^\top b)$
- Find zero derivative:

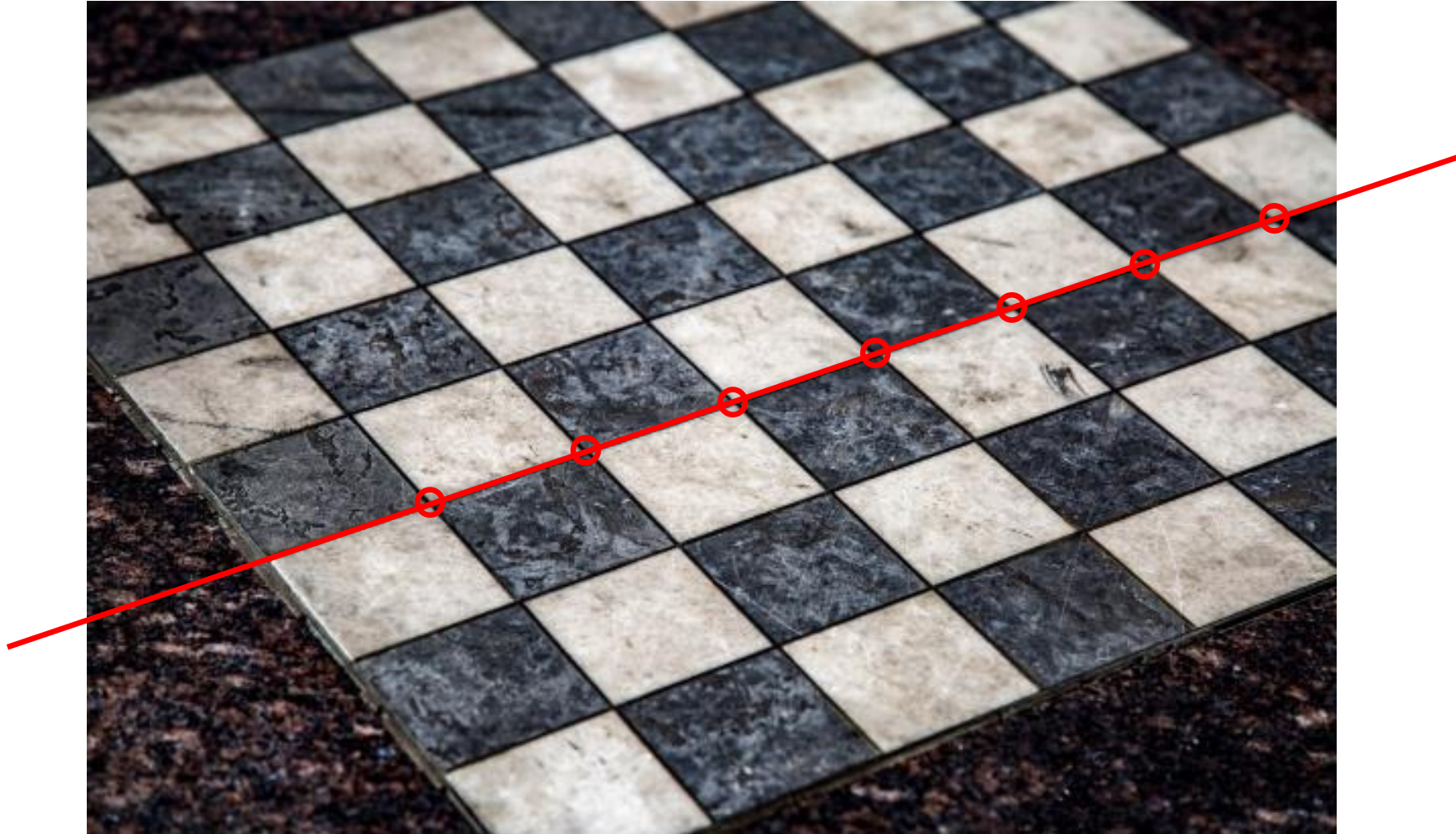
$$2A^\top A x - 2A^\top b = 0$$

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

# Linear Least Squares Solution

- We can also calculate the pseudo-inverse matrix by using SVD or QR decomposition.
  - more numerically stable 
  - works when **A** is rank deficient 
  - more computationally expensive 
    - $O(mn^2)$
  - Matlab:  $x = A \backslash b$  (backslash operator / mldivide)

# Least Squares Line





# Least Squares Line



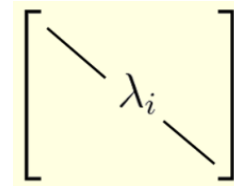
# Linear Least Squares Solution

- $\operatorname{argmin}_x \|Ax\|_2 \quad s.t. \quad \|x\|_2 = 1$
- Calculate SVD of A:  
 $A = UDV^T$
- The solution is the last column of V.
  - (unit) singular vector of A with the least singular value.
  - (unit) eigenvector of  $A^T A$  with the least eigenvalue

# SVD

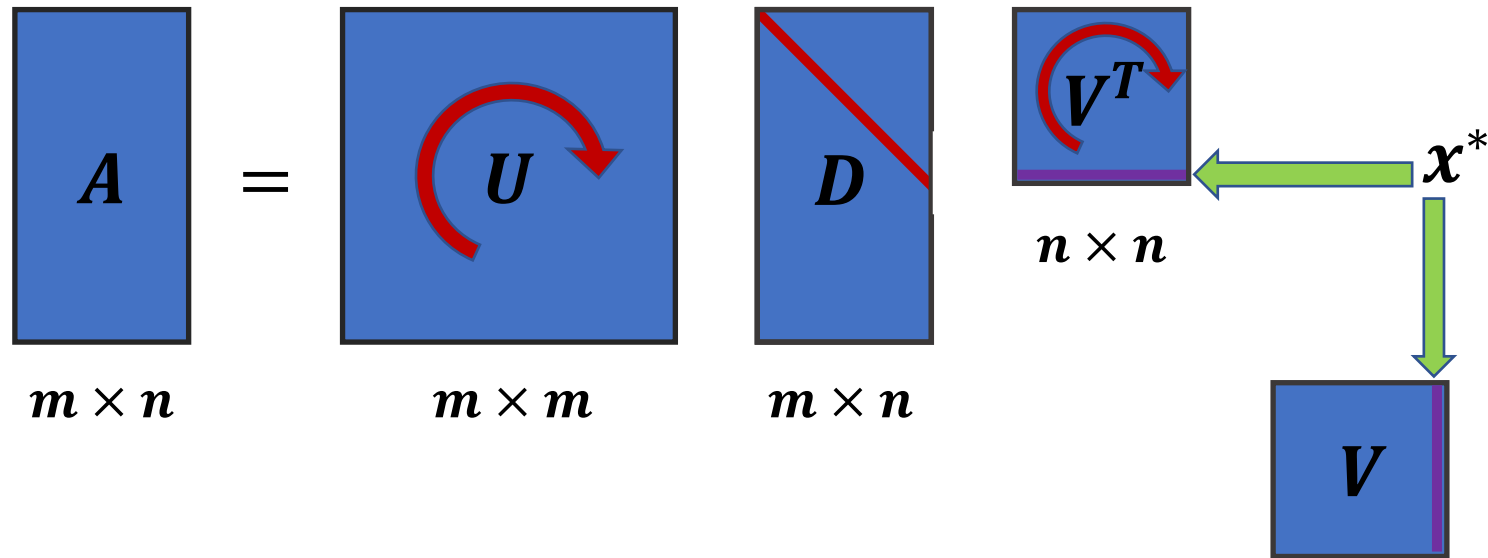
## Singular Value Decomposition

- $A = UDV^T$  is the SVD of  $A$  if:
  - $U \in M_{m \times m}$  Orthonormal ( $U^T U = I_{m \times m}$ )
  - $V \in M_{n \times n}$  Orthonormal ( $V^T V = I_{n \times n}$ )
  - $D \in M_{m \times n}$  Diagonal with non-negative entries ordered in descending order.
- $D$  diagonal entries are:
  - called **singular values** of  $A$
  - square root of the **eigenvalues** of  $A^T A$
- $V$  columns are the **eigenvectors** of  $A^T A$ .
  - $A^T A v_i = V D^T U^T U D V^T v_i = V D^T D V^T v_i = V D^T D e_i = V \lambda_i^2 e_i = \lambda_i^2 v_i$



# SVD

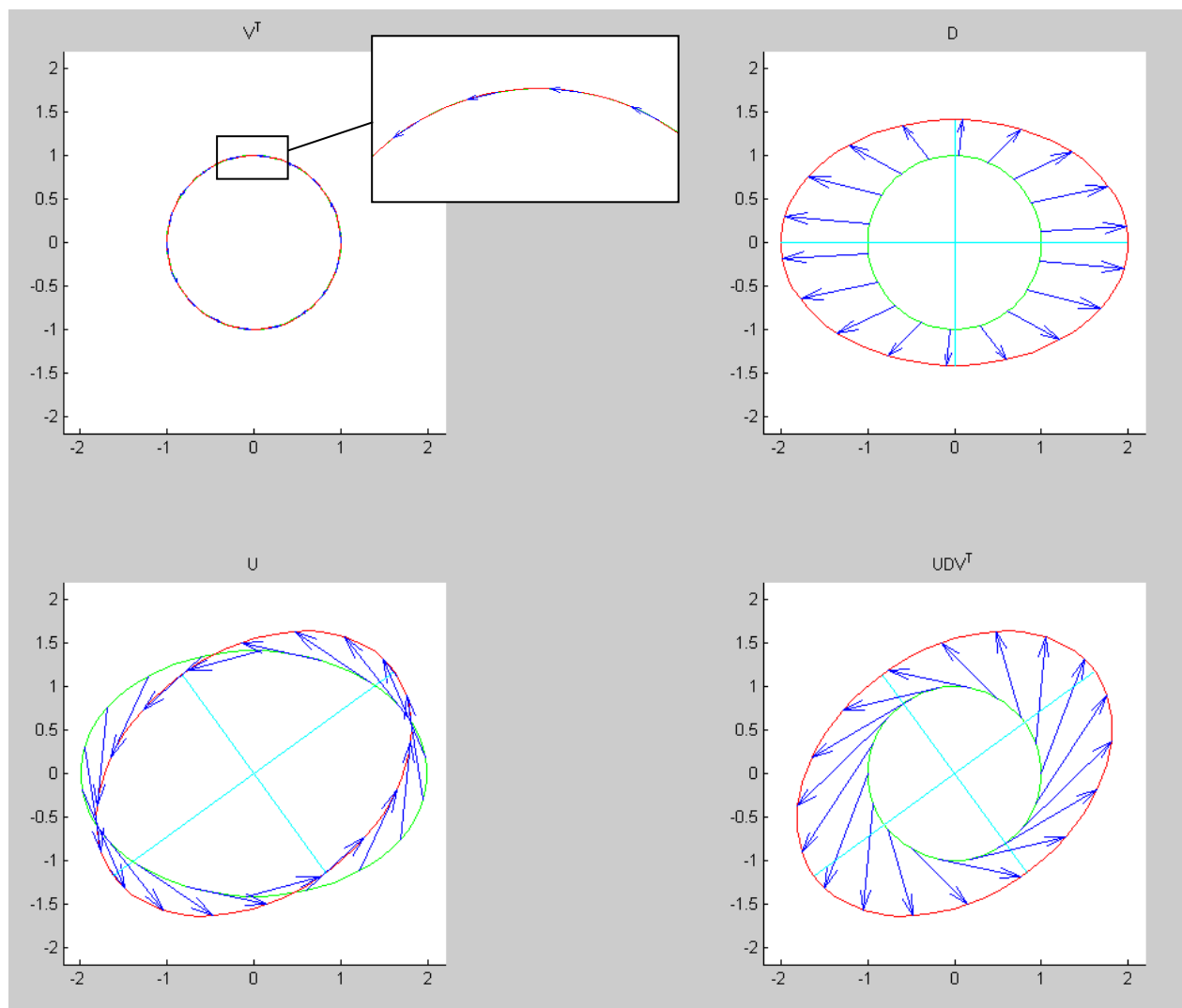
## Singular Value Decomposition





# SVD:

$$A = UDV^T$$



# Linear Least Squares Solution

- $\operatorname{argmin}_x \|Ax\|_2 \quad s.t. \quad \|x\|_2 = 1$
- Calculate SVD of A:  
 $A = UDV^\top$
- The solution is the last column of V.
  - (unit) singular vector of A with the least singular value.
  - (unit) eigenvector of  $A^\top A$  with the least eigenvalue

# Least Squares

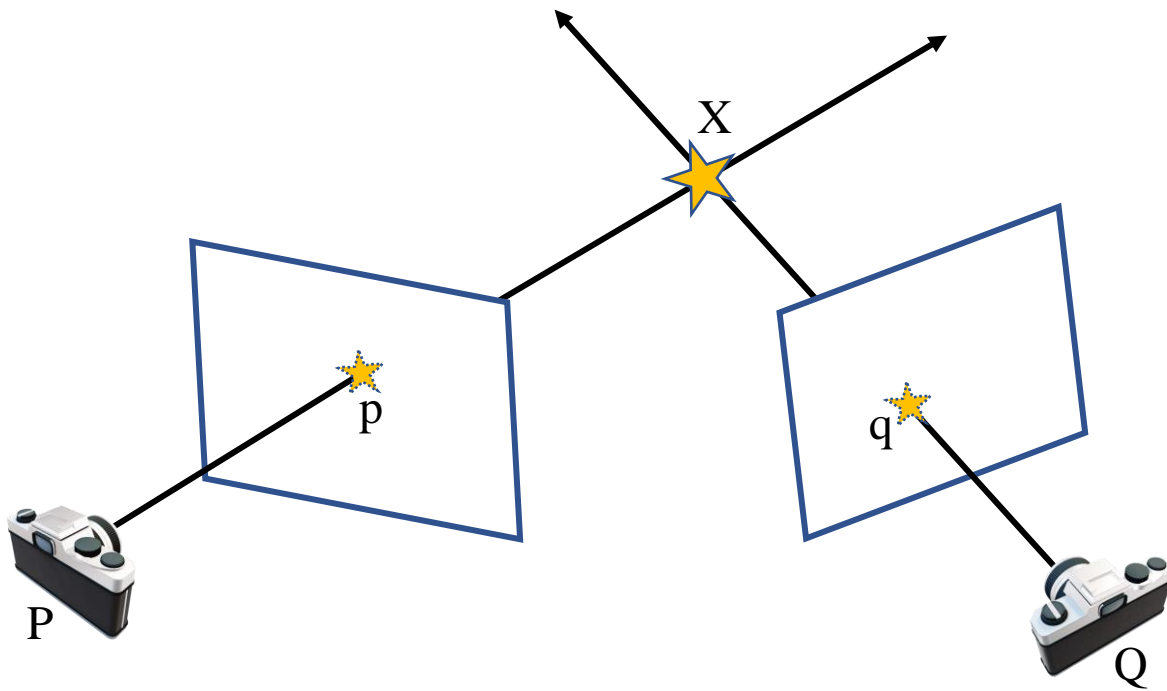
## Usage

- When to use least squares?
  - Global solution
  - Outliers can be removed
  - The noise is Gaussian
    - or is uncorrelated, has zero mean and equal variance
- Linear least squares is much easier
  - When The data can be arranged in a linear model
  - Or can be linearly approximated

# Triangulation

# Triangulation

- Calibration  $(P, Q)$ , correspondences  $(p, q)$



$$P = \begin{bmatrix} \text{---} p_1^\top \text{---} \\ \text{---} p_2^\top \text{---} \\ \text{---} p_3^\top \text{---} \end{bmatrix}$$

$$Q = \begin{bmatrix} \text{---} q_1^\top \text{---} \\ \text{---} q_2^\top \text{---} \\ \text{---} q_3^\top \text{---} \end{bmatrix}$$

$$p = \begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix} \quad q = \begin{bmatrix} q_x \\ q_y \\ 1 \end{bmatrix}$$

# Triangulation

- We look for  $X = \tilde{\lambda} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$  s.t.  $\lambda p = PX$   
 $\hat{\lambda} q = QX$

$$\begin{bmatrix} \lambda p_x \\ \lambda p_y \\ \lambda \end{bmatrix} = \begin{bmatrix} \text{---} P_1^\top \text{---} \\ \text{---} P_2^\top \text{---} \\ \text{---} P_3^\top \text{---} \end{bmatrix} X$$

$$\begin{bmatrix} \hat{\lambda} q_x \\ \hat{\lambda} q_y \\ \hat{\lambda} \end{bmatrix} = \begin{bmatrix} \text{---} Q_1^\top \text{---} \\ \text{---} Q_2^\top \text{---} \\ \text{---} Q_3^\top \text{---} \end{bmatrix} X$$

$$\begin{bmatrix} P_3^\top p_x - P_1^\top p_x \\ P_3^\top p_y - P_2^\top p_y \\ Q_3^\top q_x - Q_1^\top q_x \\ Q_3^\top q_y - Q_2^\top q_y \end{bmatrix} X = 0$$