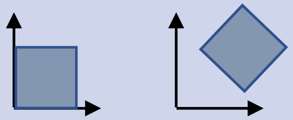
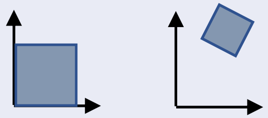
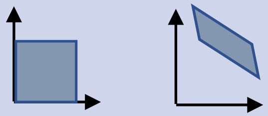
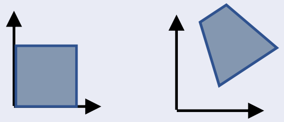
Camera Model
Linear Least Squares
Triangulation

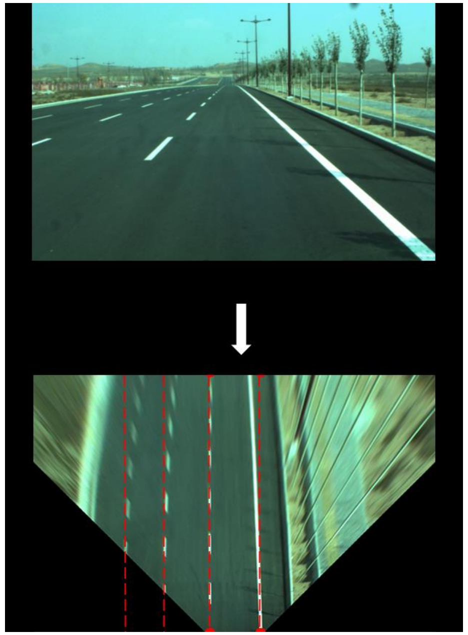
David Arnon

Projective Geometry

- Every {camera,plane} has a different horizon

Homogeneous Coordinates

Transformation		d.o.f	H
Rigid **Isometry** **Motion**		3	$\begin{bmatrix} R & t \\ 0^T & 1 \end{bmatrix}$
Similarity		4	$\begin{bmatrix} sR & t \\ 0^T & 1 \end{bmatrix}$
Affine		6	$\begin{bmatrix} a & b & t_1 \\ c & d & t_2 \\ 0 & 0 & 1 \end{bmatrix}$
Homography **Projectivity** **Planar**		8	$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix}$



Courtesy of line.17qq.com

Camera Model

Kitti Cameras

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

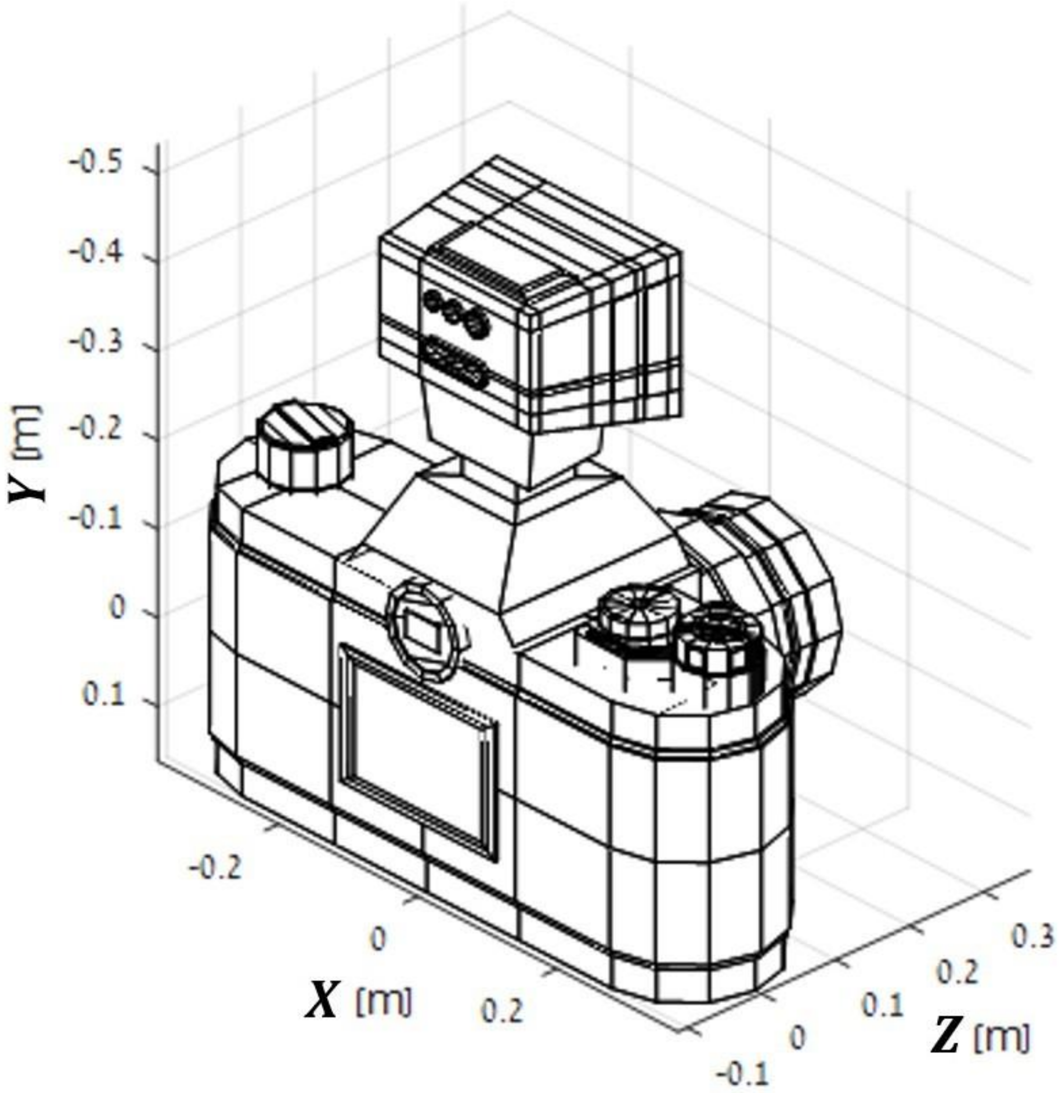
- Right Camera: $$\begin{bmatrix} 707 & 0 & 602 & -380 \\ 0 & 707 & 183 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Kitti Cameras

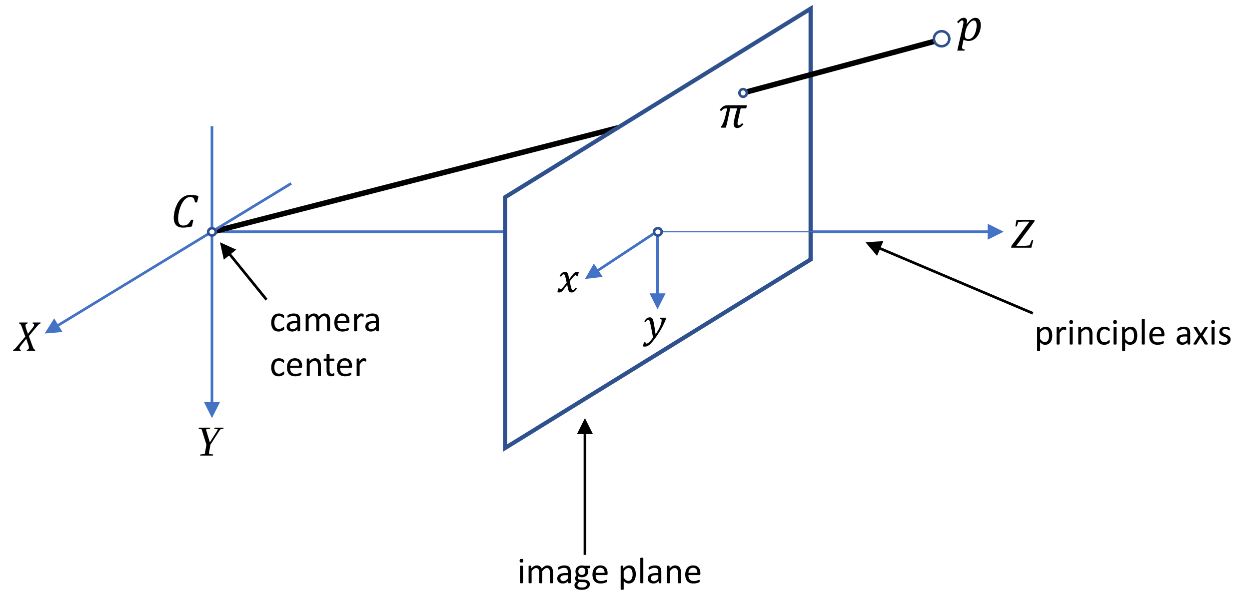

```
print('left  cam keypoint:', kp1[0].pt)
print('right cam keypoint:', kp2[0].pt)

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)
```

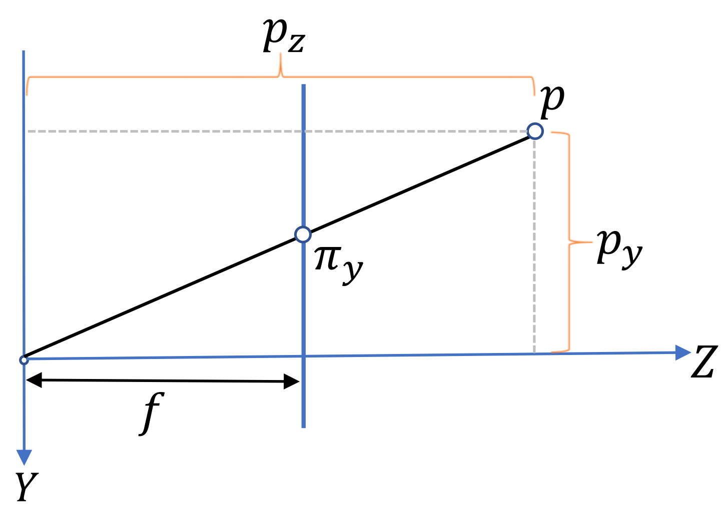
```
left  cam keypoint: (922.7208251953125, 30.694913864135742)
right cam keypoint: (907.992919921875, 29.992298126220703)
3D point: [[11.7  -5.57 25.79]]
```

Camera Coordinates



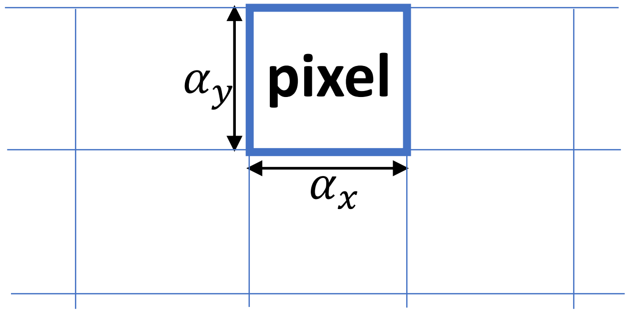
Camera Coordinates





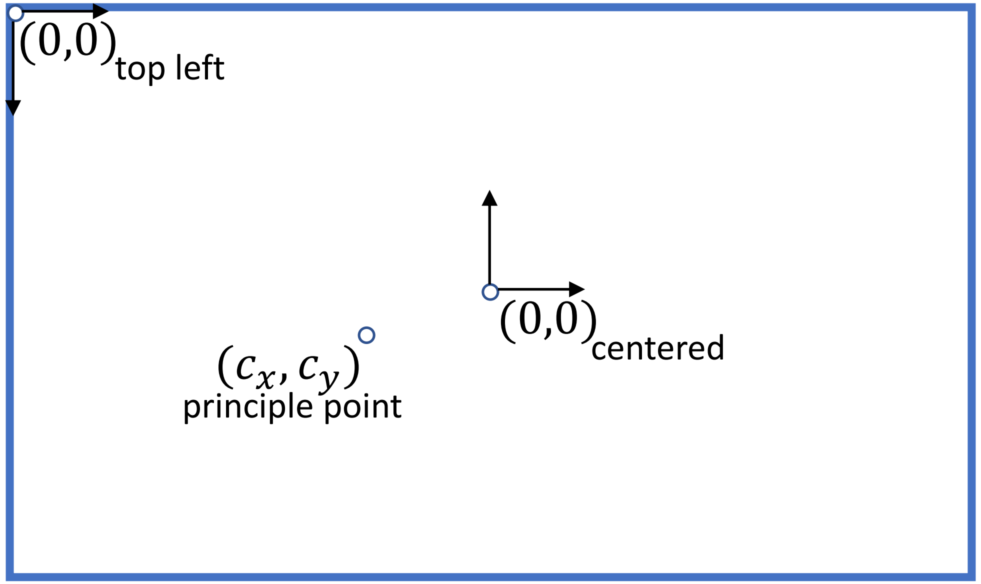
$$\pi_y = f\frac{p_y}{p_z}$$

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: $$\underbrace{\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}}_{K} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} f_x x + c_x z \\ f_y y + c_y z \\ z \end{bmatrix} \propto \begin{bmatrix} f_x \dfrac{x}{z} + c_x \\ f_y \dfrac{y}{z} + c_y \\ 1 \end{bmatrix}$$

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.

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

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

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

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

Camera Model

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

- 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

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

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

- 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:

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

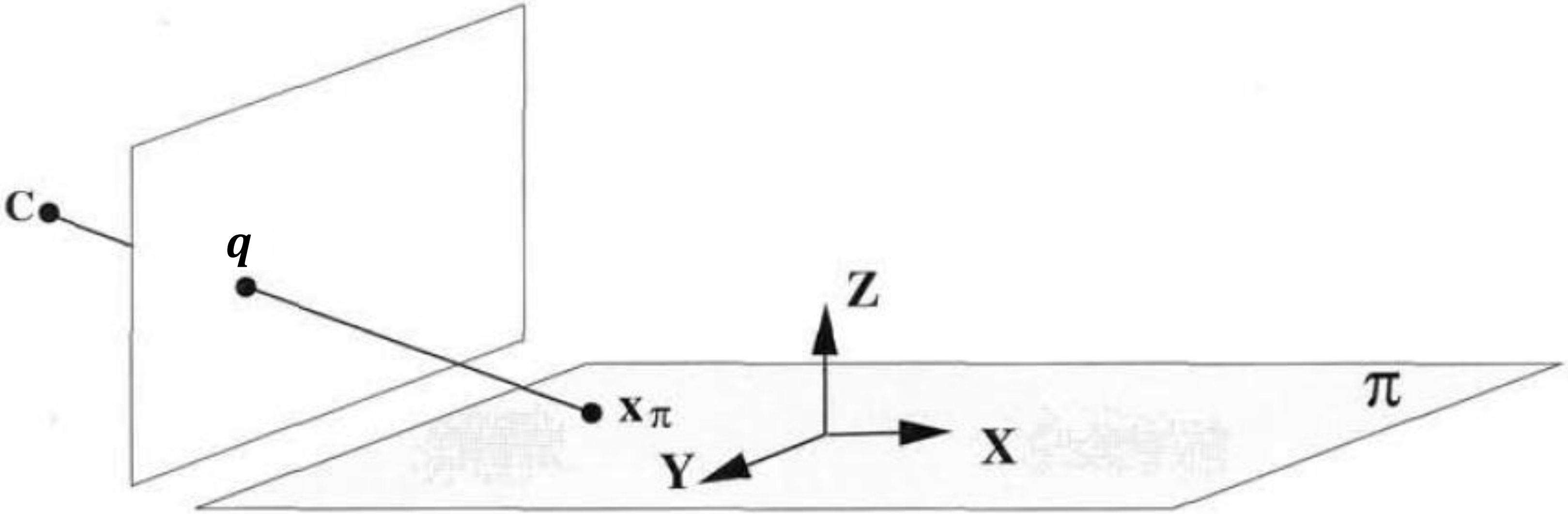
$$0 = c + t$$

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

Homography

- Projection of a plane is an Homography:

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



Homography

- Rotated cameras are related by an Homography:

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

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

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

Horizon

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

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

- $l = p_1 \times p_2$

Line

- A line is projected to a line:

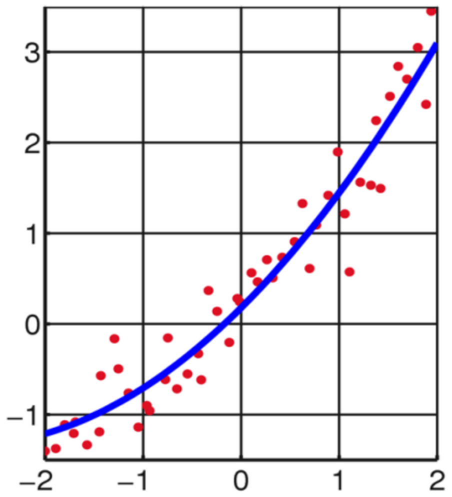
$$P\begin{bmatrix} a + td \\ 1 \end{bmatrix} = P\left(\begin{bmatrix} a \\ 1 \end{bmatrix} + \begin{bmatrix} td \\ 0 \end{bmatrix}\right) = P\begin{bmatrix} a \\ 1 \end{bmatrix} + tP\begin{bmatrix} d \\ 0 \end{bmatrix}$$

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

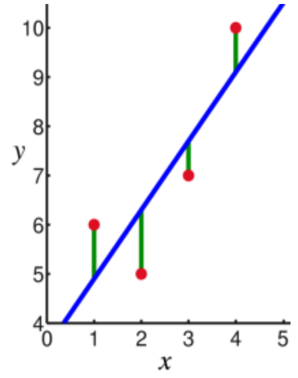
Linear Least Squares

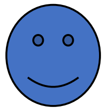
Least Squares

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



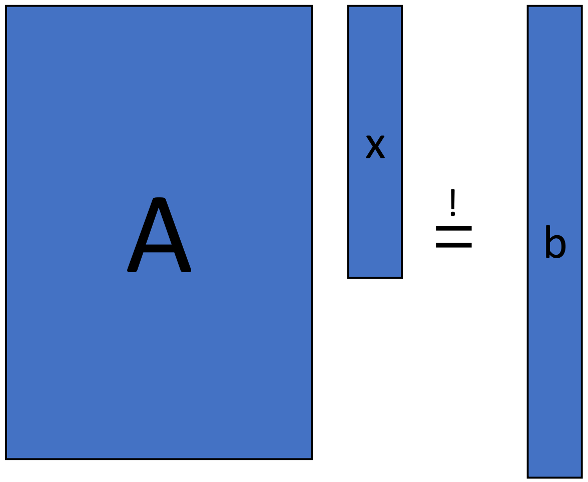
Least Squares



- 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

- $\boldsymbol{argmin_x \|Ax - b\|_2}$
  - $A \in M_{m\times n} \quad m \geq n$
  - $x \in M_{n\times 1}$



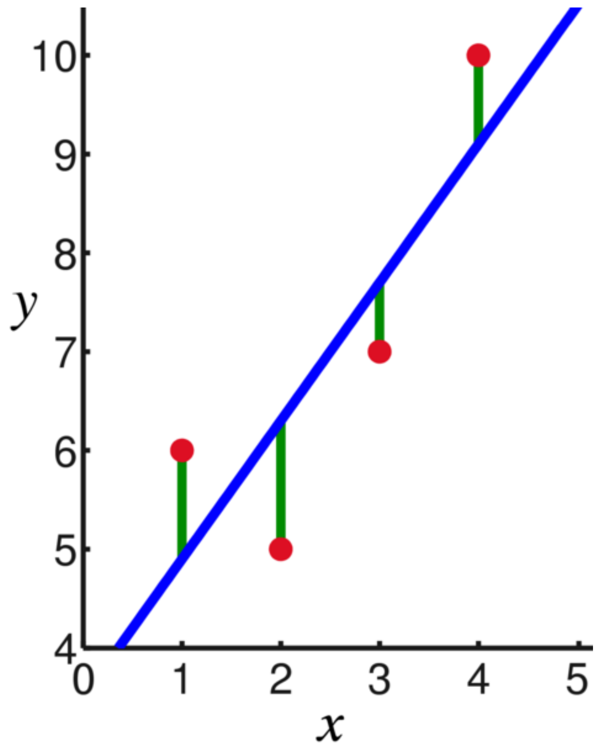
- $\boldsymbol{argmin_x \|Ax\|_2} \qquad s.t.\ \|x\|_2 = 1$

Linear Least Squares

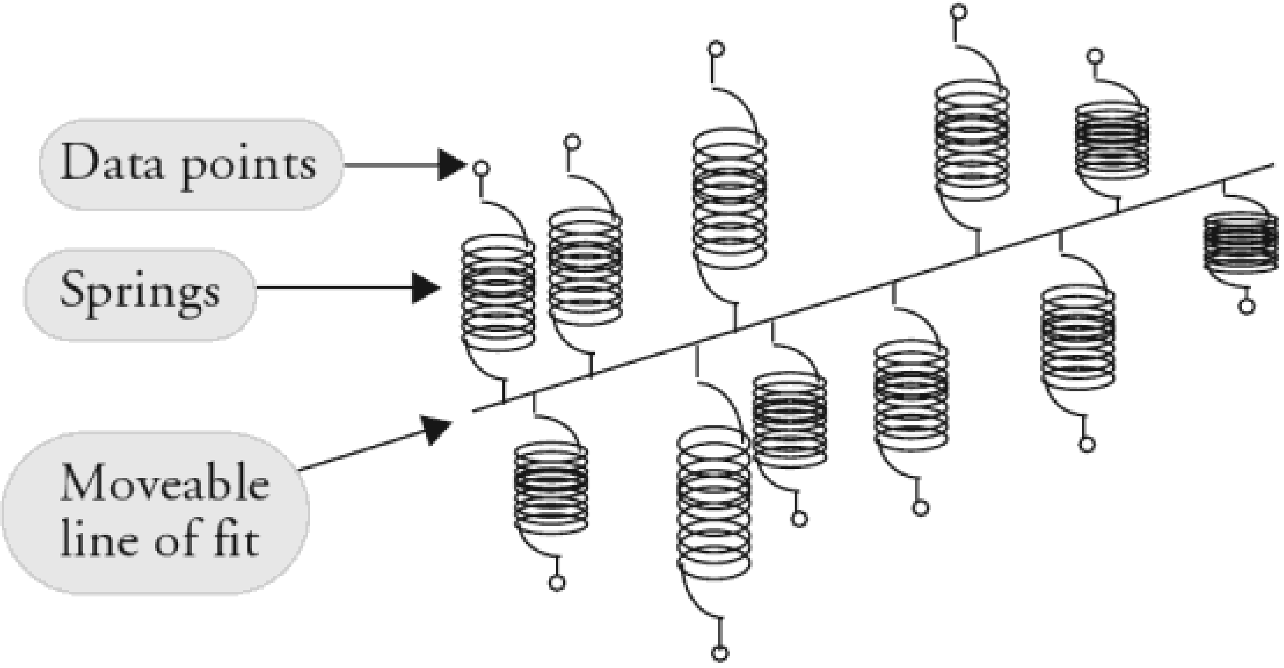
Example - Line

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

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



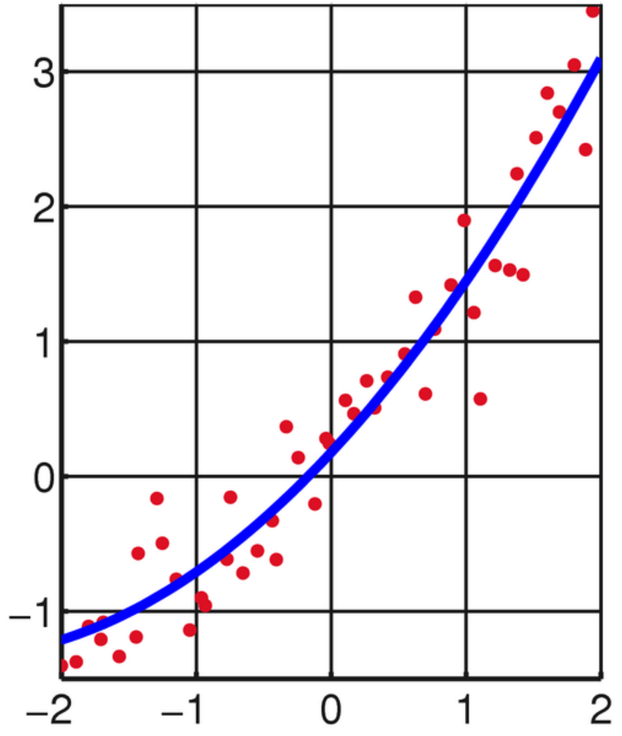
Linear Least Squares
Example - Line



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

- $\boldsymbol{argmin_x \|Ax - b\|_2}$

- The solution is $\boldsymbol{x = A^+ b}$
- $A^+ = (A^T A)^{-1} A^T$
    - A^+ is the pseudo-inverse matrix of A

- For large problems we can solve $A^T A x = A^T b$ instead of inverting $A^T A$. (Cholesky decomposition)

Linear Least Squares

Pseudo-Inverse Proof

- $\text{argmin}_x \ \|A \cdot x - b\|_2 =$
  $\text{argmin}_x \ (Ax - b)^T \cdot (Ax - b) =$
  $\text{argmin}_x \ (x^T A^T - b^T) \cdot (Ax - b) =$
  $\text{argmin}_x \ (x^T A^T A x - b^T A x - x^T A^T b + b^T b)$

- Find zero derivative:

  $2A^T A x - 2A^T b = 0$
  $x = (A^T A)^{-1} A^T b$

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
  - Matlab: x = A\b (backslash operator / mldivide)

Linear Least Squares
Solution

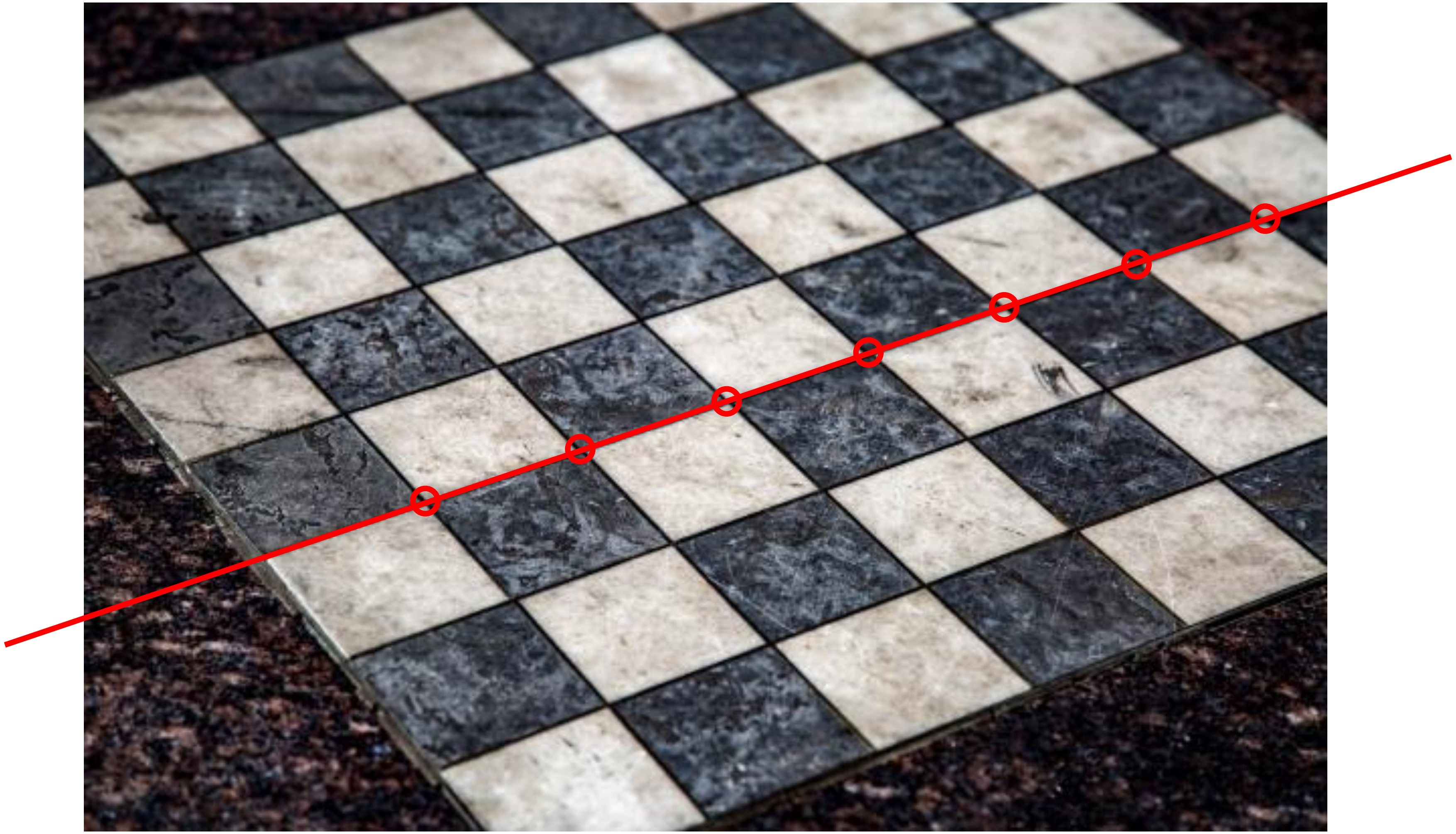
- $\boldsymbol{argmin_x \|Ax\|_2} \quad s.t. \;\; \|x\|_2 = 1$

- Calculate SVD of A:
  $A = UDV^T$

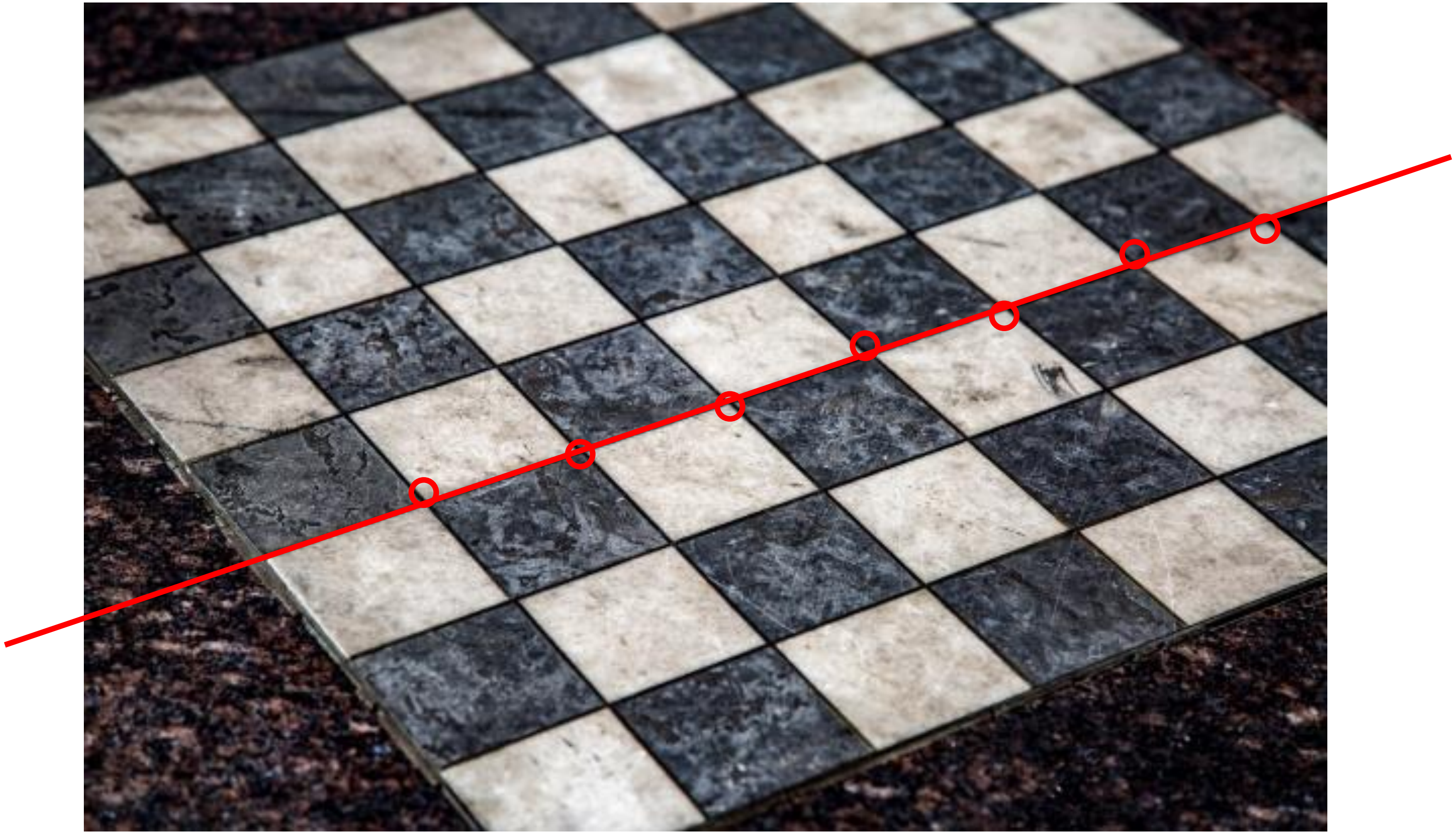
- The solution is the last column of V.
  - (unit) singular vector of A with the least singular value.
  - (unit) eigenvector of A^TA with the least eigenvalue

Least Squares

Line



Least Squares
Line



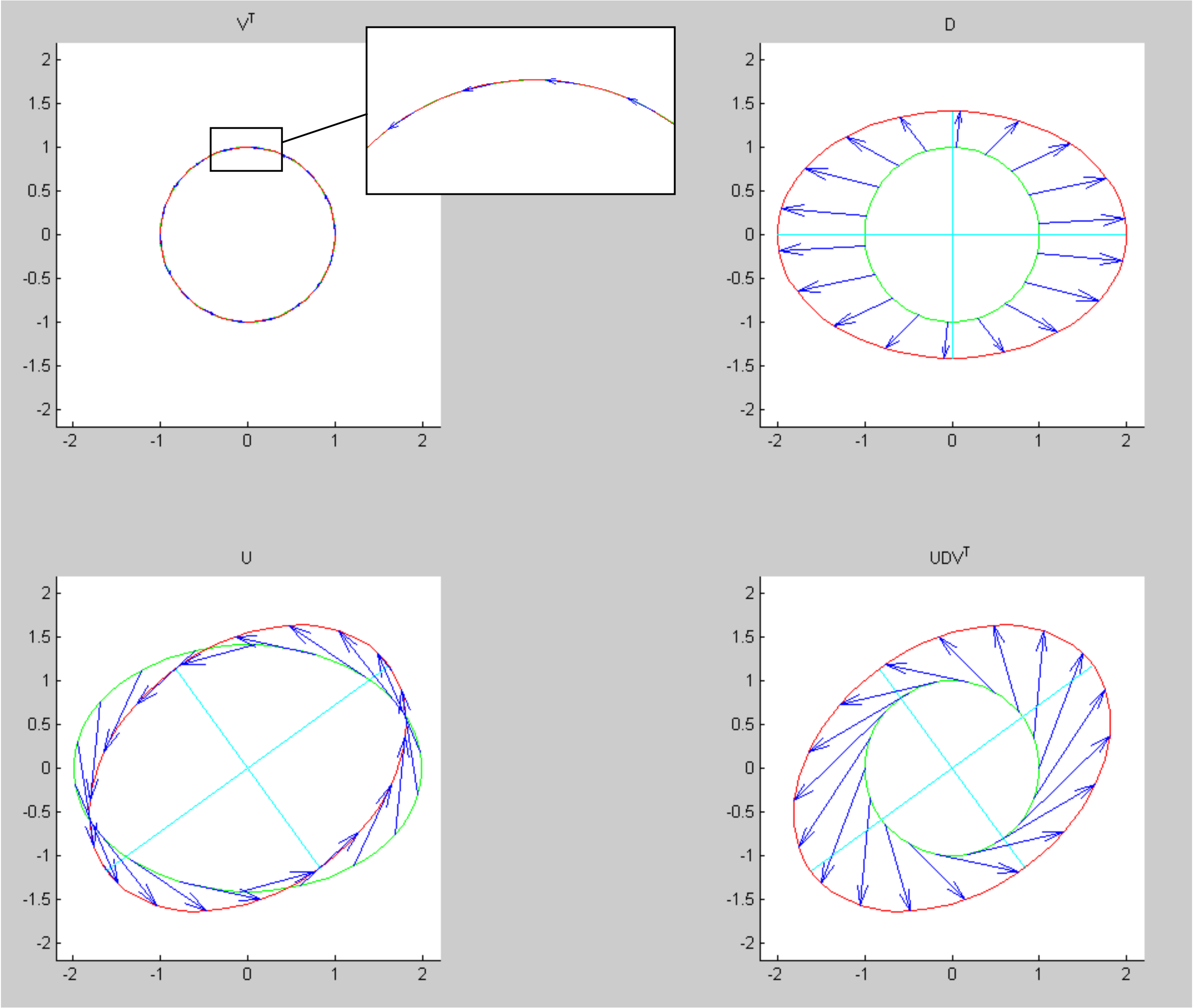
SVD
Singular Value Decomposition

- $A = UDV^{T}$ 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 $A^{T}A$

$$\begin{bmatrix} \ddots & & \\ & \lambda_i & \\ & & \ddots \end{bmatrix}$$

- V columns are the **eigenvectors** of $A^{T}A$.
  - $A^{T}Av_i = VD^{T}U^{T}UDV^{T}v_i = VD^{2}V^{T}v_i = VD^{2}e_i = V\lambda_i^{2}e_i = \lambda_i^{2}v_i$

SVD: $A = UDV^T$



Linear Least Squares

Solution

- $\boldsymbol{argmin_x} \|\boldsymbol{Ax}\|_{\mathbf{2}} \quad s.t. \ \|x\|_2 = 1$

- 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^TA with the least eigenvalue

Least Squares
Usage

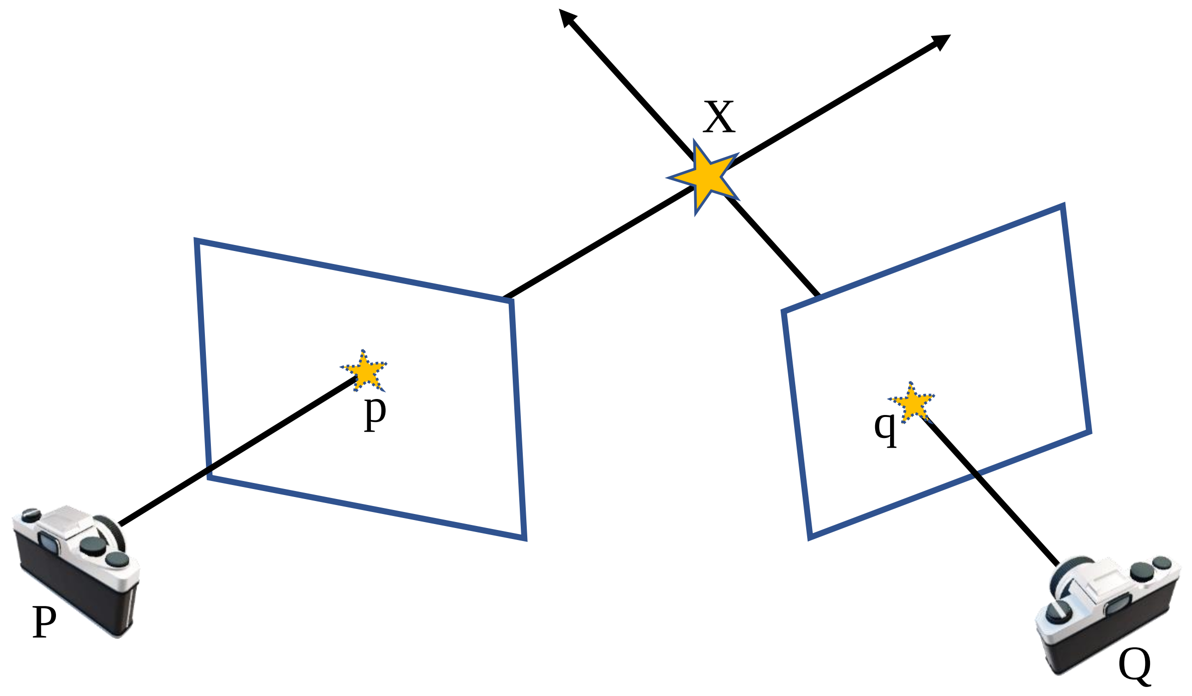
- When to use least squares?
  - Global solution
  - Outliers can be removed
  - The noise is Gaussian
    - or is uncorrelated, has zero mean and equal variance

  - 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)



$$P = \begin{bmatrix} — P_1 — \\ — P_2 — \\ — P_3 — \end{bmatrix}$$

$$Q = \begin{bmatrix} — Q_1 — \\ — Q_2 — \\ — Q_3 — \end{bmatrix}$$

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

Triangulation

- We look for $X = \tilde{\lambda}\begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ 1 \end{bmatrix}$ s.t. $\lambda p = PX$ and $\hat{\lambda} q = QX$

$$\begin{bmatrix} \lambda p_x \\ \lambda p_y \\ \lambda \end{bmatrix} = \begin{bmatrix} \text{---}\; P_1 \;\text{---} \\ \text{---}\; P_2 \;\text{---} \\ \text{---}\; P_3 \;\text{---} \end{bmatrix} X \qquad \begin{bmatrix} \hat{\lambda} q_x \\ \hat{\lambda} q_y \\ \hat{\lambda} \end{bmatrix} = \begin{bmatrix} \text{---}\; Q_1 \;\text{---} \\ \text{---}\; Q_2 \;\text{---} \\ \text{---}\; Q_3 \;\text{---} \end{bmatrix} X$$

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