3D Vision 1

Week 7

Image Formation

Camera Projection Matrix

Announcements

- Assignment 2 due in **one week** (11:59pm Friday 26 April)
  - This includes a one week extension that has already been applied
  - **Zero** marks if either report or code submitted late (unless extension)
    - Submit early; you can always resubmit an updated version later
    - Depending on your internet connection and load on the TurnItIn servers, uploading can sometimes be slow, so please factor this into your submission schedule
  - Submit your report (PDF) and code (ZIP file) **separately under the correct tab** in the submission box
  - Follow the instructions under Submission Requirements

Announcements

- Public holiday Thursday 25 April: Thursday lab rescheduled at **13:00-15:00 Tuesday Rm 109 CSIT Building**

Weekly Study Plan: Overview

Wk	Starting	Lecture	Lab	Assessment
1	19 Feb	Introduction	✗	
2	26 Feb	Low-level Vision 1	1	
3	4 Mar	Low-level Vision 2	1	
		Mid-level Vision 1		
4	11 Mar	Mid-level Vision 2	1	CLab1 report due Friday
		High-level Vision 1		
5	18 Mar	High-level Vision 2	2	
6	25 Mar	High-level Vision 3[1]	2	
	1 Apr	Teaching break	✗	
	8 Apr	Teaching break	✗	
7	15 Apr	3D Vision 1	2	CLab2 report due Friday
8	22 Apr	3D Vision 2	3	
9	29 Apr	3D Vision 3	3	
10	6 May	3D Vision 4	3	
		Mid-level Vision 3		
11	13 May	High-level Vision 4	✗	CLab3 report due Friday
12	20 May	Course Review	✗	

Weekly Study Plan: Part B

Wk	Starting	Lecture	By
7	15 Apr	3D vision: introduction, camera model, single-view geometry	Dylan
8	22 Apr	3D vision: camera calibration, two-view geometry (homography)	Dylan
9	29 Apr	3D vision: two-view geometry (epipolar geometry, triangulation, stereo)	Dylan
10	6 May	3D vision: multiple-view geometry	Weijian
		Mid-level vision: optical flow, shape-from-X	Dylan
11	13 May	High-level vision: self-supervised learning, detection, segmentation	Dylan
12	20 May	Course review	Dylan

Outline

1. Introduction to 3D vision
2. Model fitting (line fitting)
    1. Least squares
    2. M-estimation
    3. RANSAC
    4. Hough transform
3. Image formation (review): pinhole camera model
4. Camera projection matrix and single view geometry
5. Camera calibration
6. Resectioning and camera pose

Hough Transform

Fitting Multiple Lines Using a Hough Transform

- Given a binary edge image, find the lines (or curves) that explain the data points best in the parameter space
- This parameter space is called a Hough space

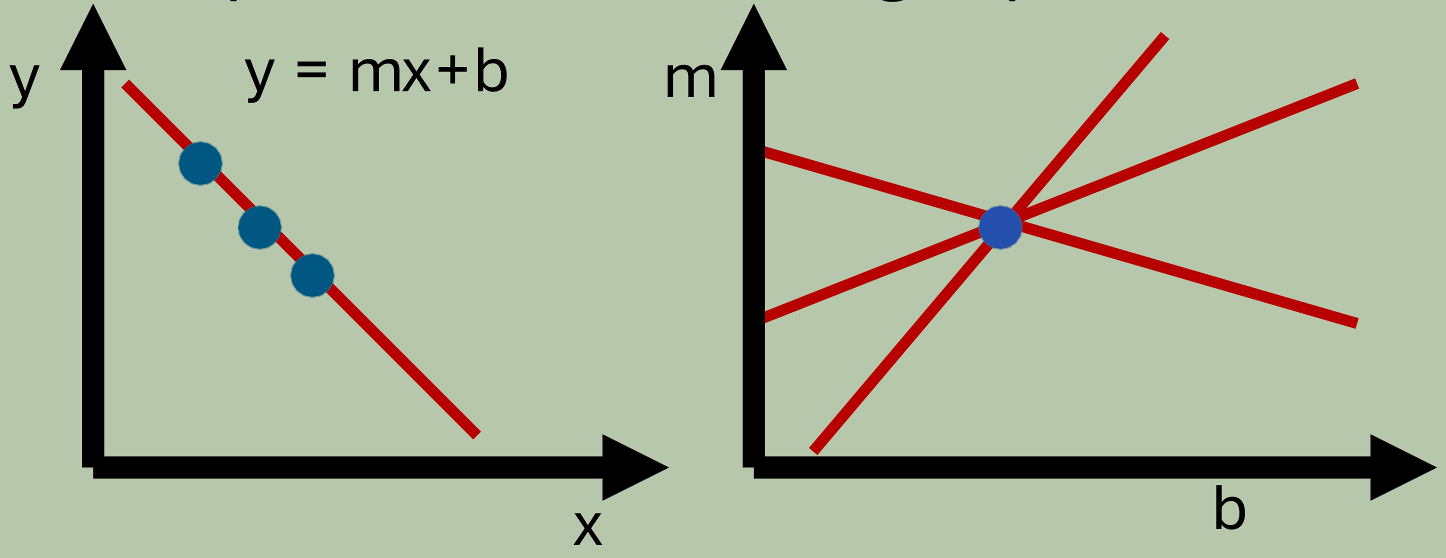


Image space   Parameter (Hough) space

P.V.C. Hough, *Machine Analysis of Bubble Chamber Pictures,* Proc. Int. Conf. High Energy Accelerators and Instrumentation, 1959. (Patented)

A point (x_1, y_1) is mapped to a line in Hough space

$y_1 = m x_1 + b$ $\quad$ $b = -x_1 m + y_1$ (rewrite the equation in terms of m and b)

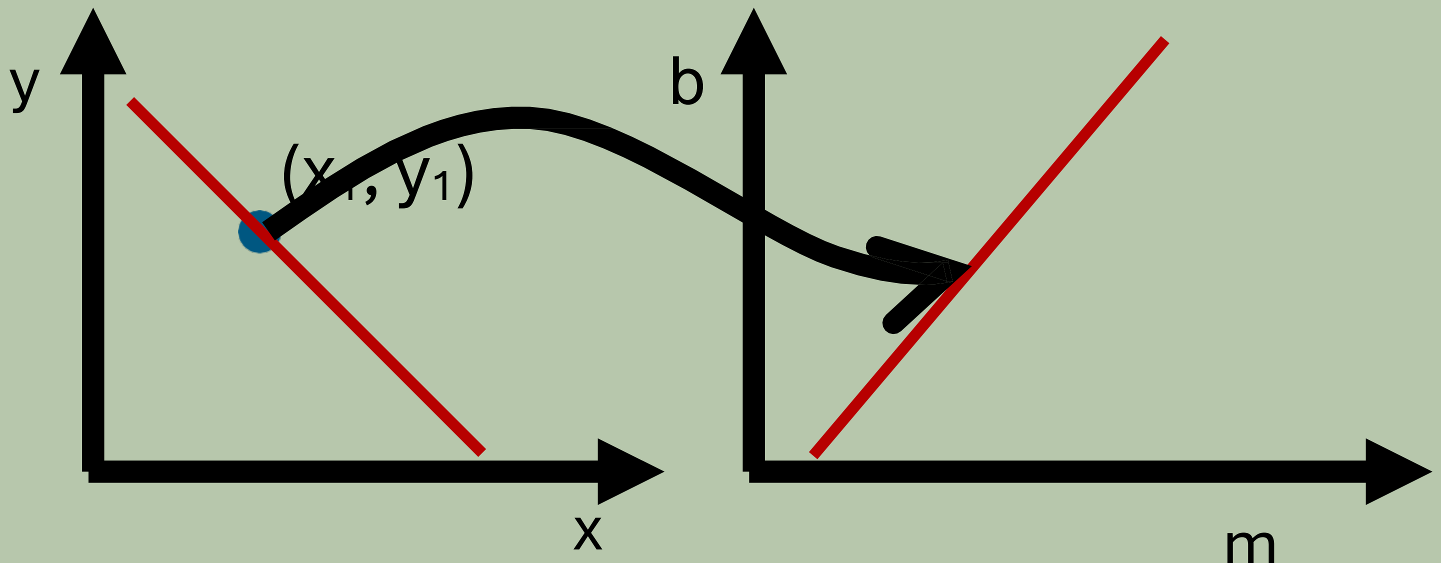


Image space

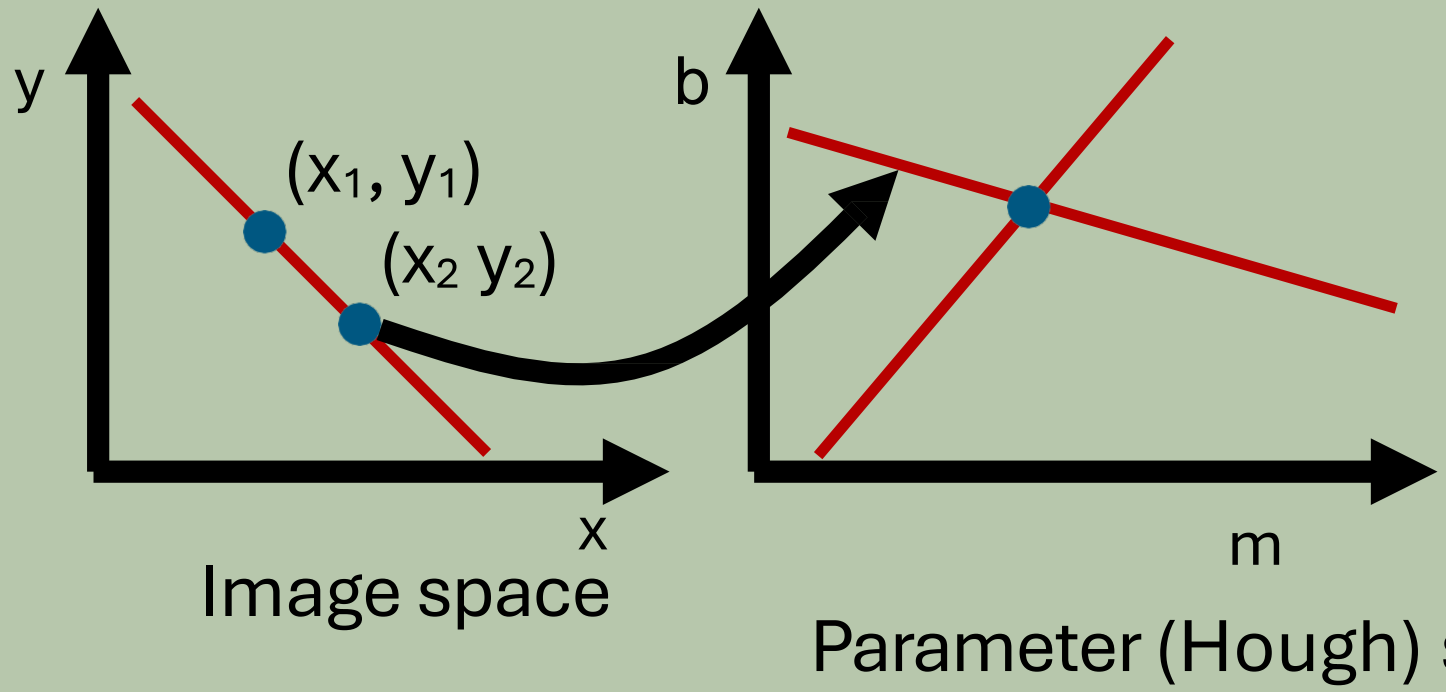
Parameter (Hough) space

A point (x_2, y_2) is mapped to a line in Hough space

$y_2 = m x_2 + b$

$b = -x_2 m + y_2$

The intersection point satisfies both (x_1, y_1) and (x_2, y_2), thus the line

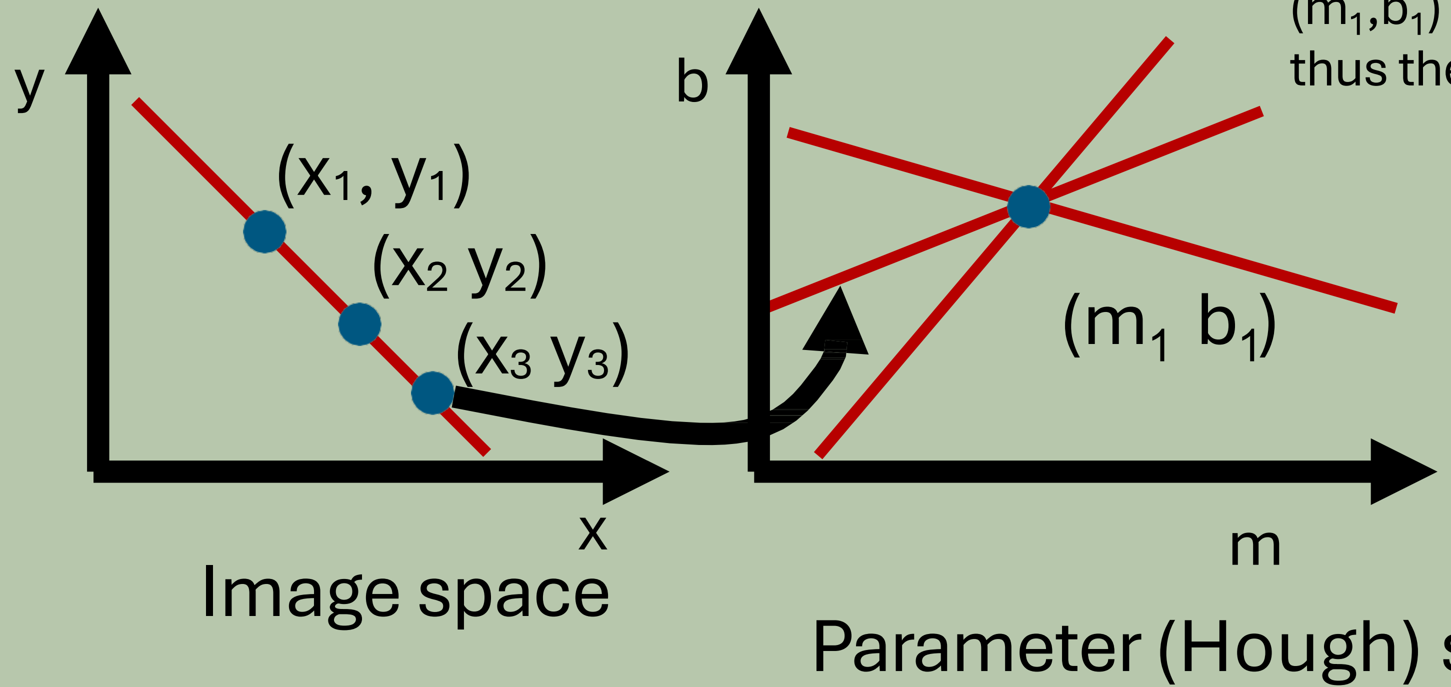


A point (x_3, y_3) is mapped to a line in Hough space

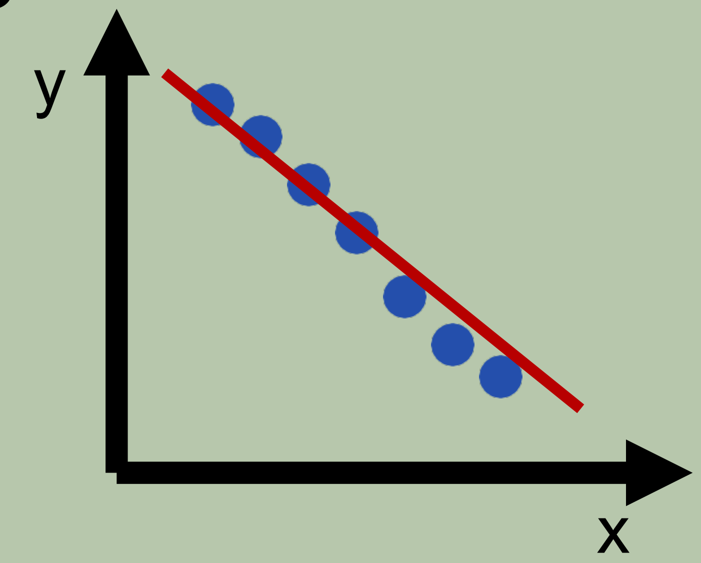
$y_3 = m x_3 + b$

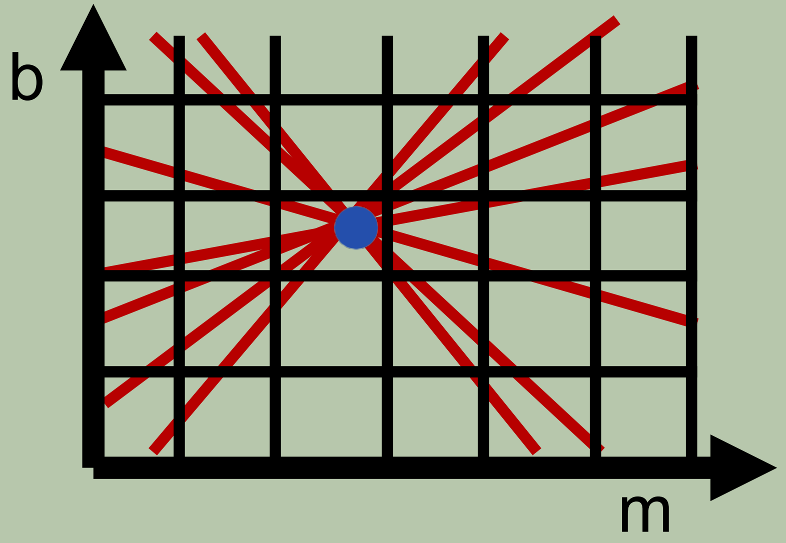
$b = -x_3 m + y_3$

The intersection point (m_1,b_1) satisfies 3 points, thus the line

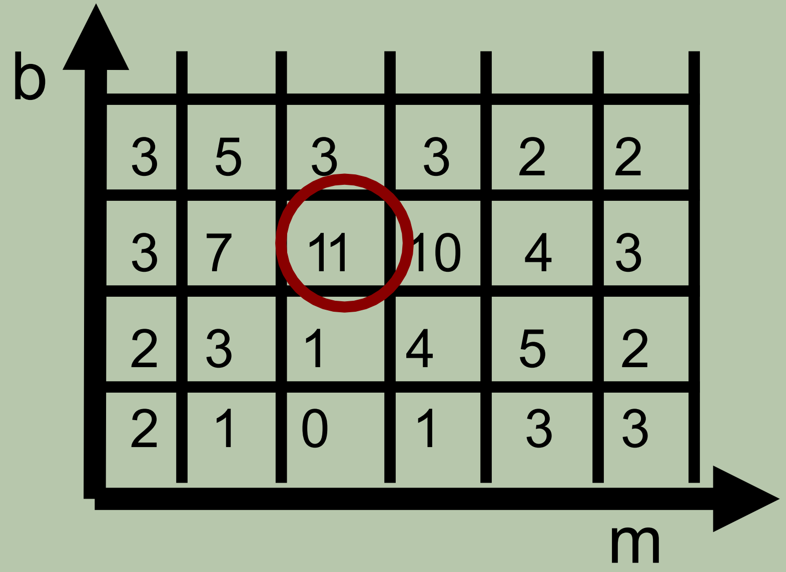


Hough Transform: Accumulator





- For each pixel, draw a line in the discretised Hough space, assigning a value of one to all the discretised positions it passes through
- Process all image pixels

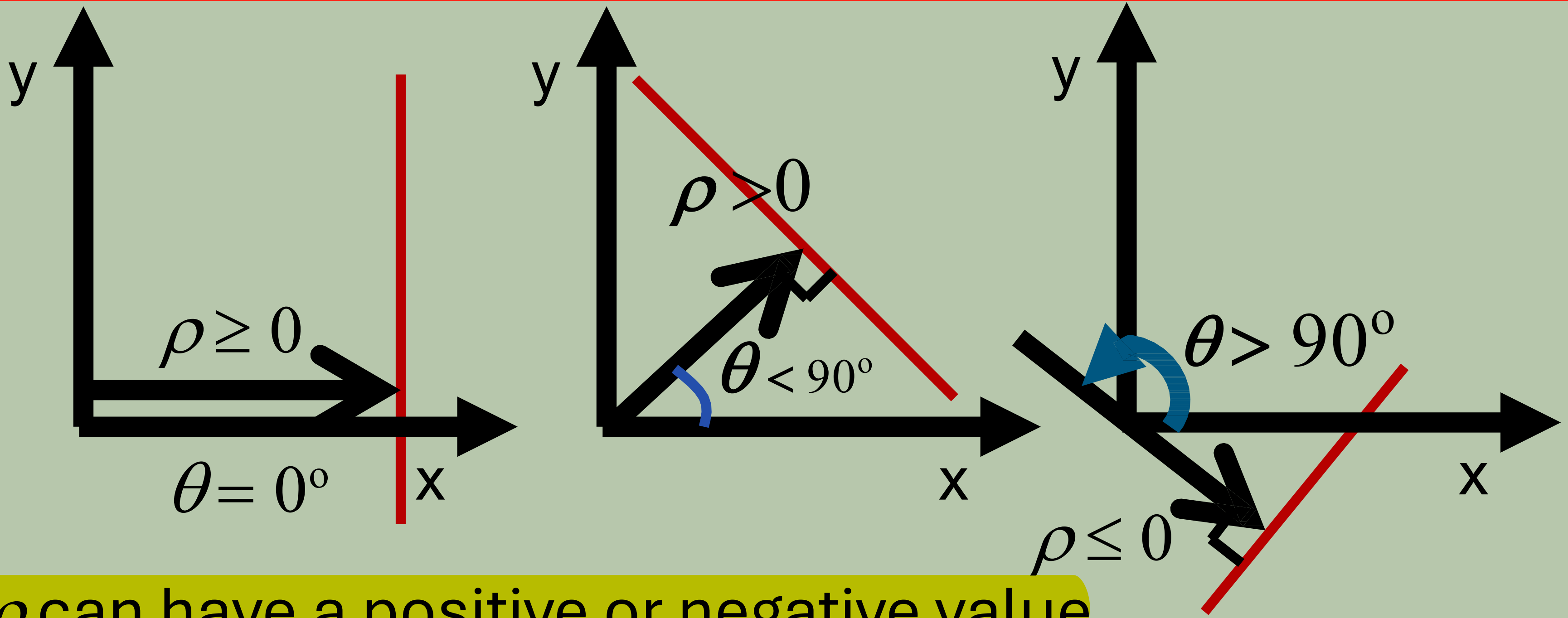


Hough Transform

- Can detect multiple lines in an image
  - from multiple local maxima
- Can easily be extended for circles and ellipses
- Computationally efficient
- Problem: *(m, b)* are unbounded
  - E.g., the slope parameter m can have an infinite value

Hough Transform: Polar Form

- Use a polar representation for the parameter space
- $x\cos\theta + y\sin\theta = \rho \rightarrow y = -\frac{\cos\theta}{\sin\theta}x + \frac{\rho}{\sin\theta}$ $\quad (0 \le \theta \le 180^o)$



- Note: ρ can have a positive or negative value

A point (x_1, y_1) is mapped to a sinusoid in the polar parameter space

$$x_1 \cos\theta + y_1 \sin\theta = \rho$$

$$\begin{aligned}\rho &= x_1 \cos\theta + y_1 \sin\theta \\ &= \alpha_1 \sin(\theta + \beta_1) \\ &= \alpha_1 \sin\beta_1 \cos\theta + \alpha_1 \cos\beta_1 \sin\theta\end{aligned}$$

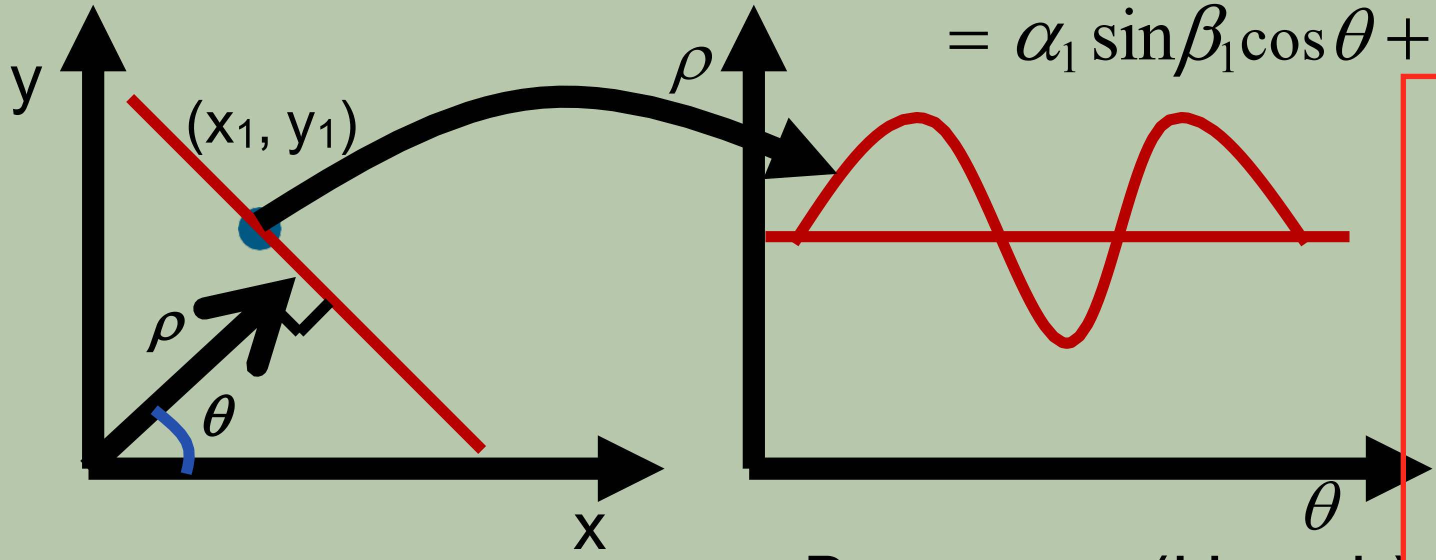


Image space

Parameter (Hough) space

$$0 \le \theta \le 180^o$$

$$\alpha_1 = \sqrt{x_1^2 + y_1^2}$$

$$\sin\beta_1 = \frac{x_1}{\sqrt{x_1^2 + y_1^2}}$$

$$\cos\beta_1 = \frac{y_1}{\sqrt{x_1^2 + y_1^2}}$$

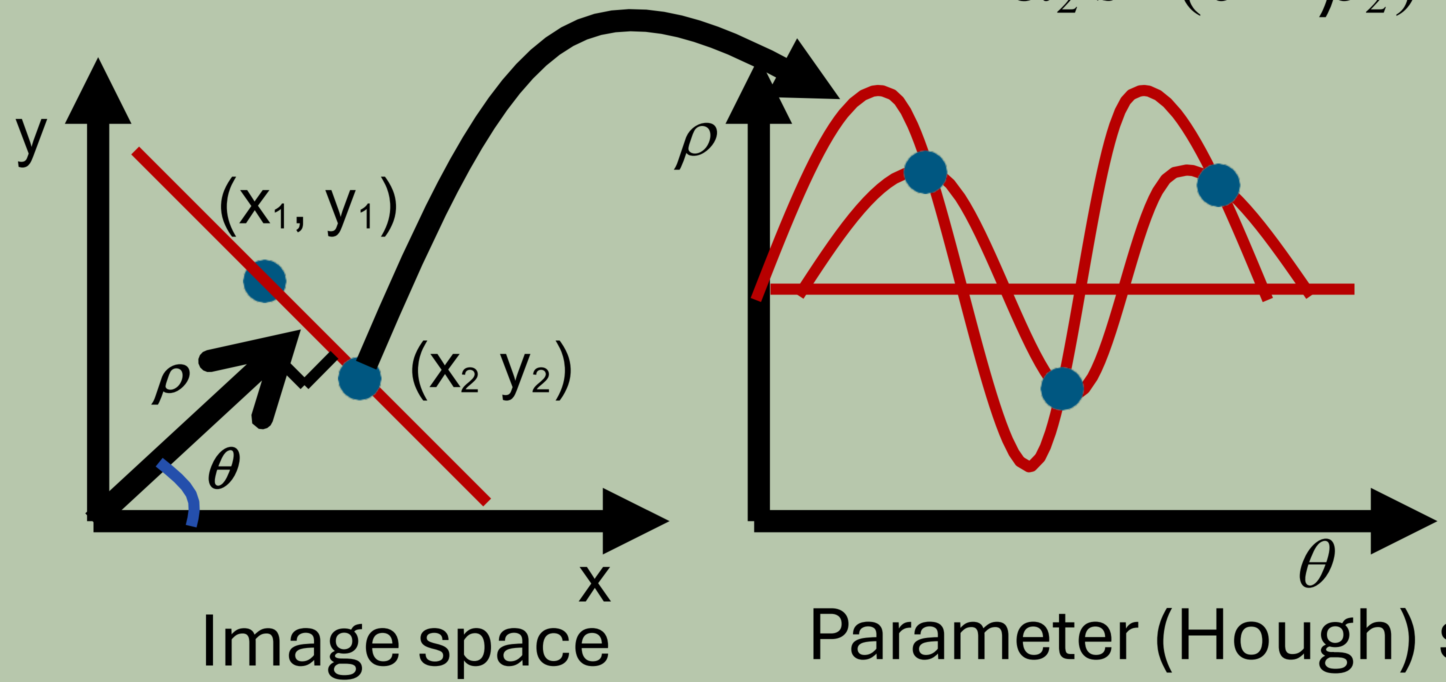
$$\beta_1 = \tan^{-1}\frac{x_1}{y_1}$$

How to map: just divide the right-hand side by $\sqrt{x_1^2 + y_1^2}$ and multiply by the same

A point (x_2, y_2) is mapped to a sinusoid in the polar parameter space

$$x_2 \cos\theta + y_2 \sin\theta = \rho$$

$$\rho = x_2 \cos\theta + y_2 \sin\theta$$
$$= \alpha_2 \sin(\theta + \beta_2)$$

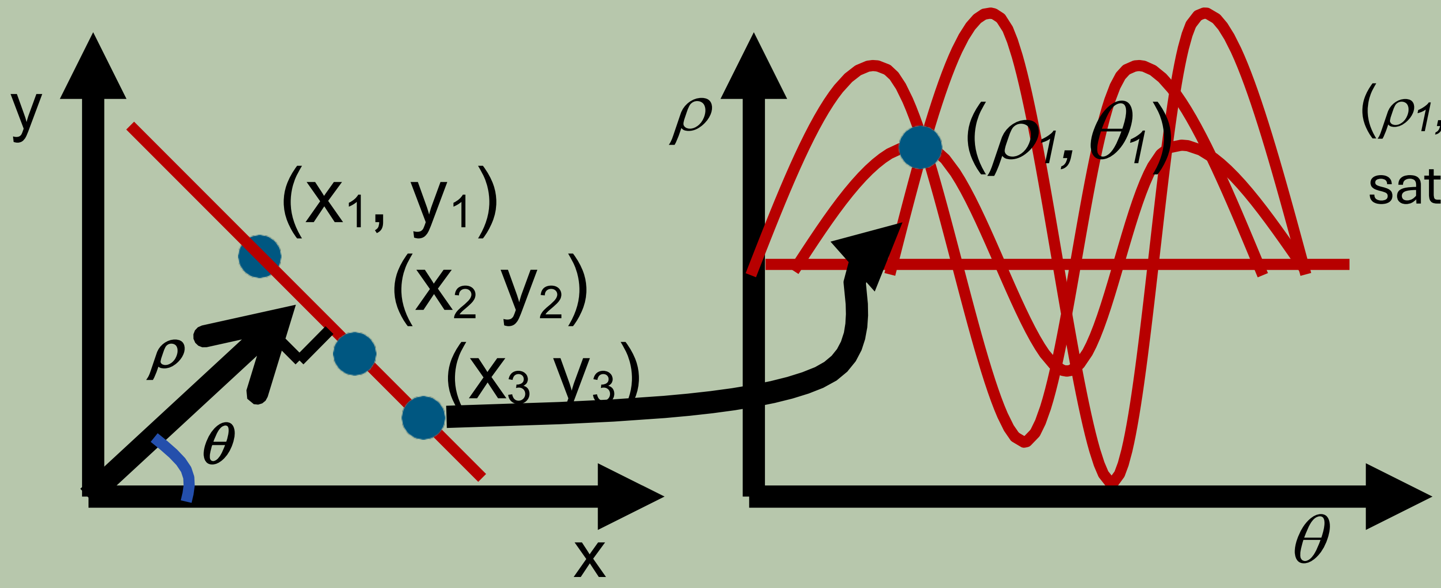


$$0 \le \theta \le 180^o$$

A point (x_3, y_3) is mapped to a sinusoid in the polar parameter space

$$x_3 \cos\theta + y_3 \sin\theta = \rho$$

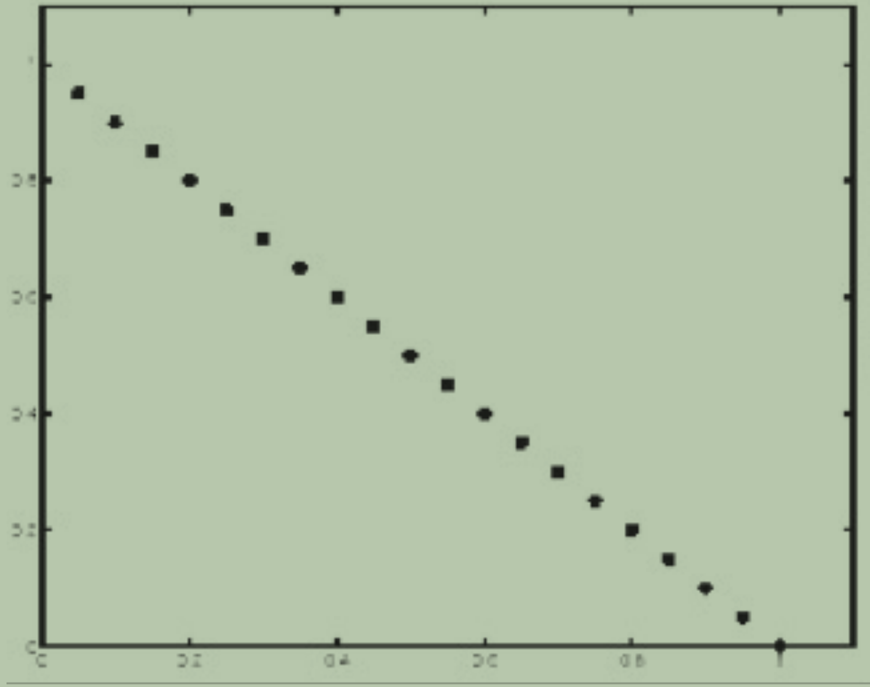
$$\rho = x_3 \cos\theta + y_3 \sin\theta$$
$$= \alpha_3 \sin(\theta + \beta_3)$$



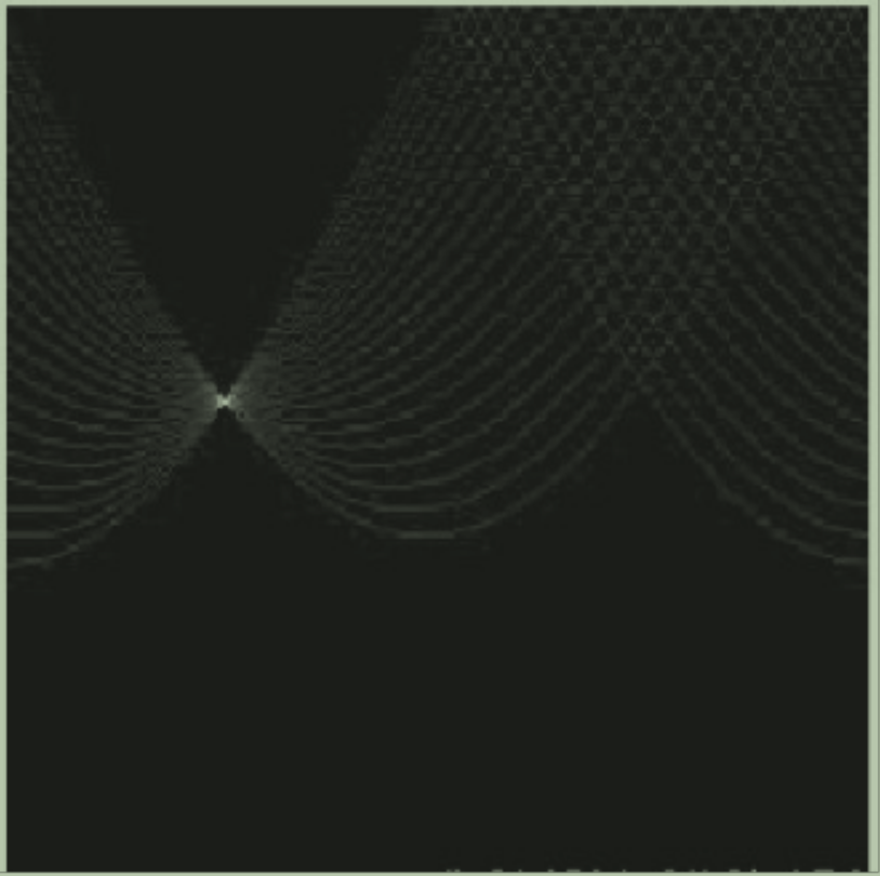
(ρ_1, θ_1)
satisfies all 3 pixel points

Image space

Hough Transform: Example



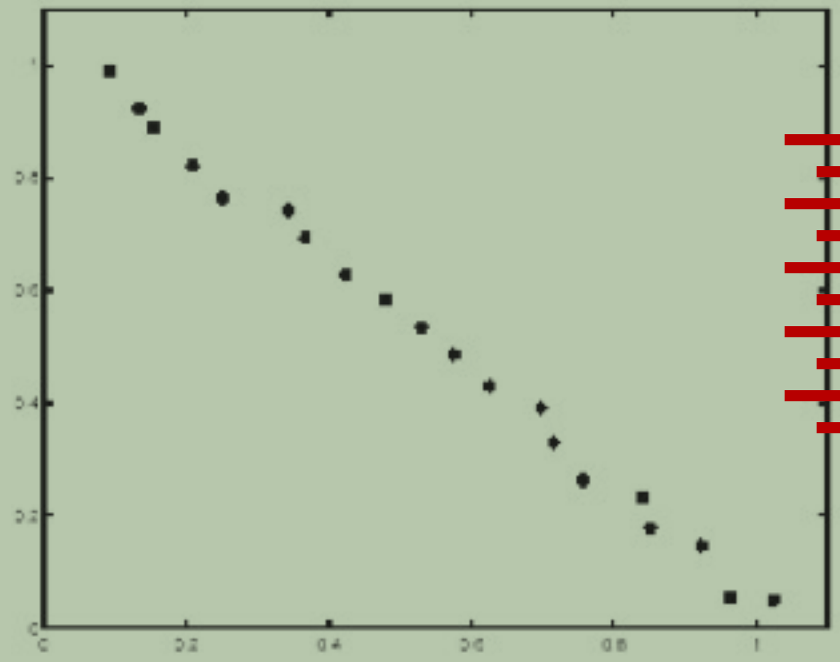
features



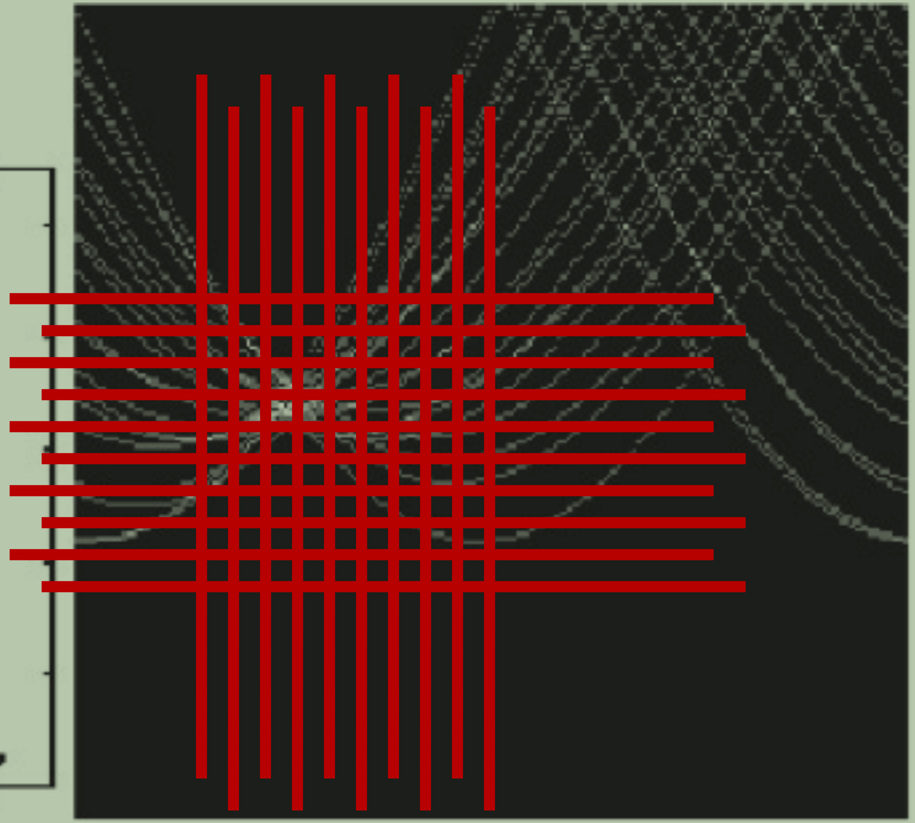
votes

Hough Transform: Example

Noisy data



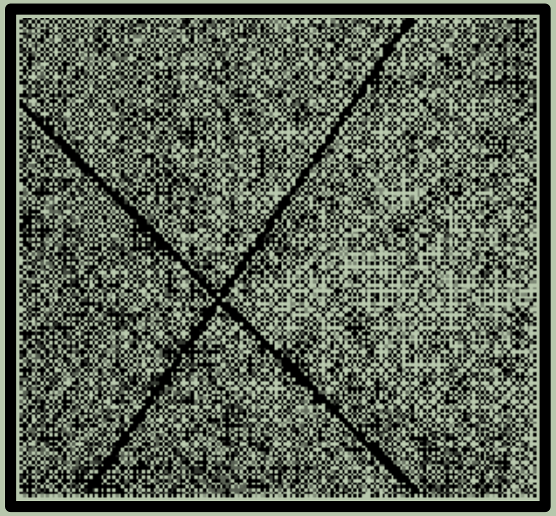
features



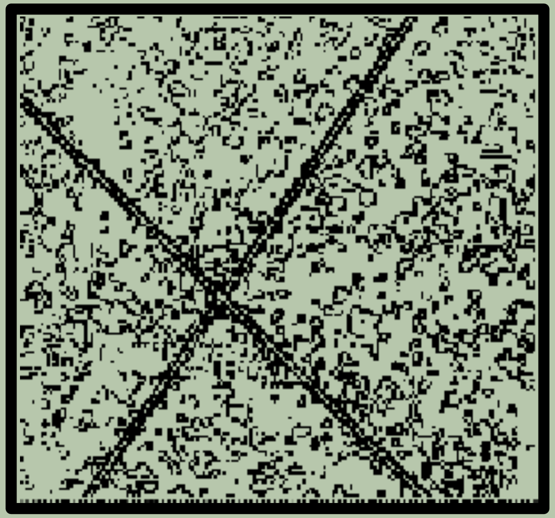
votes

Issue: the grid size needs to be adjusted...

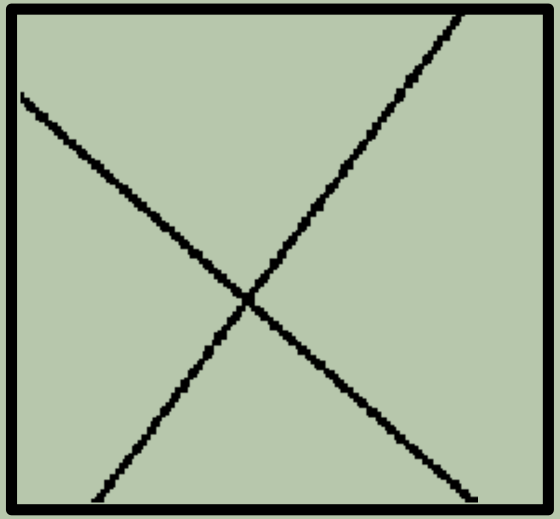
Hough Transform: Example



Image



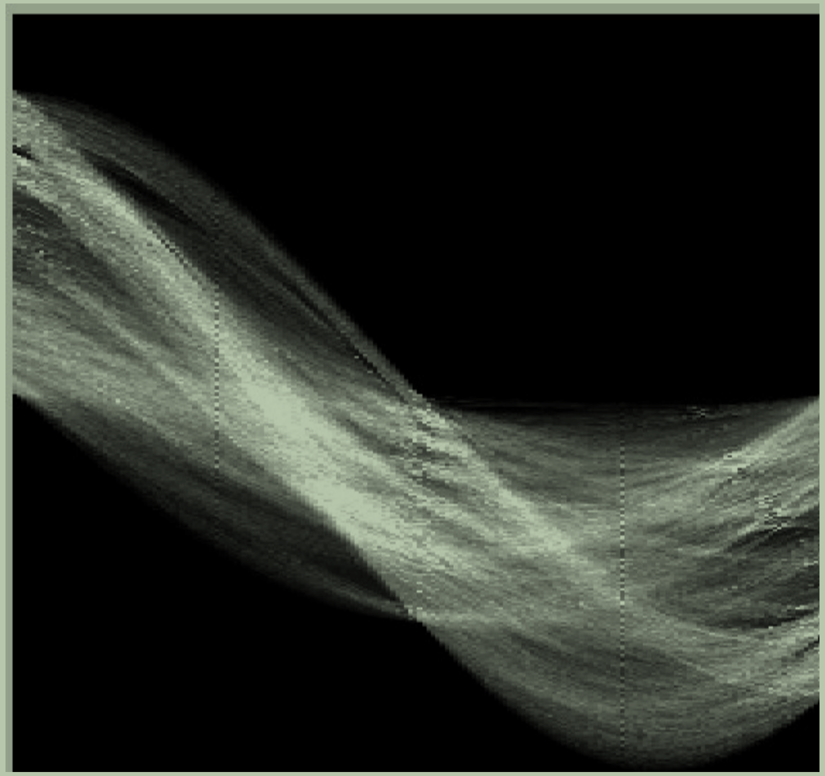
Edge Detection



Hough Transform

Hough Transform: 1) Image → Canny Edges

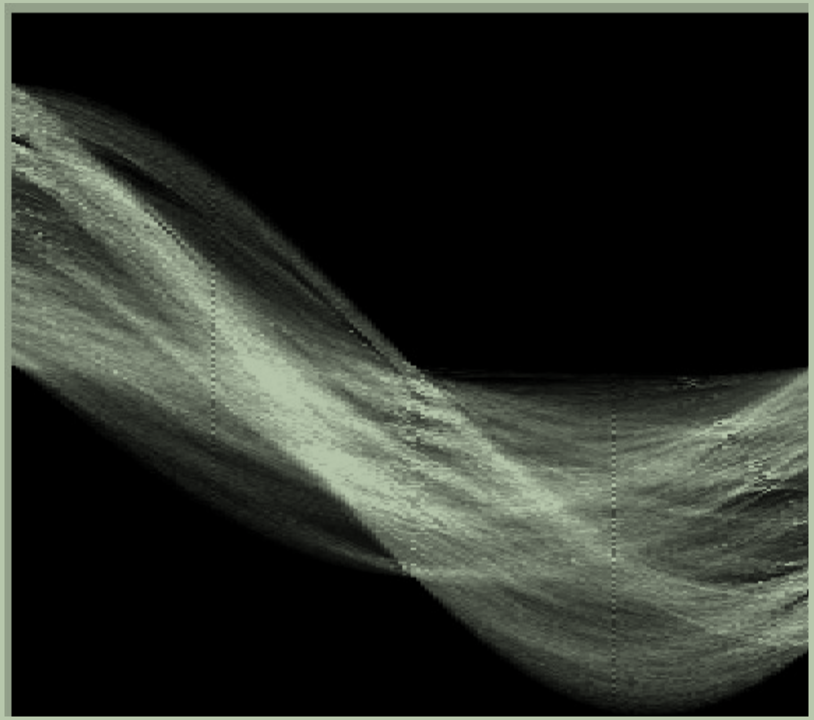
Hough Transform: 2) Canny Edges → Votes



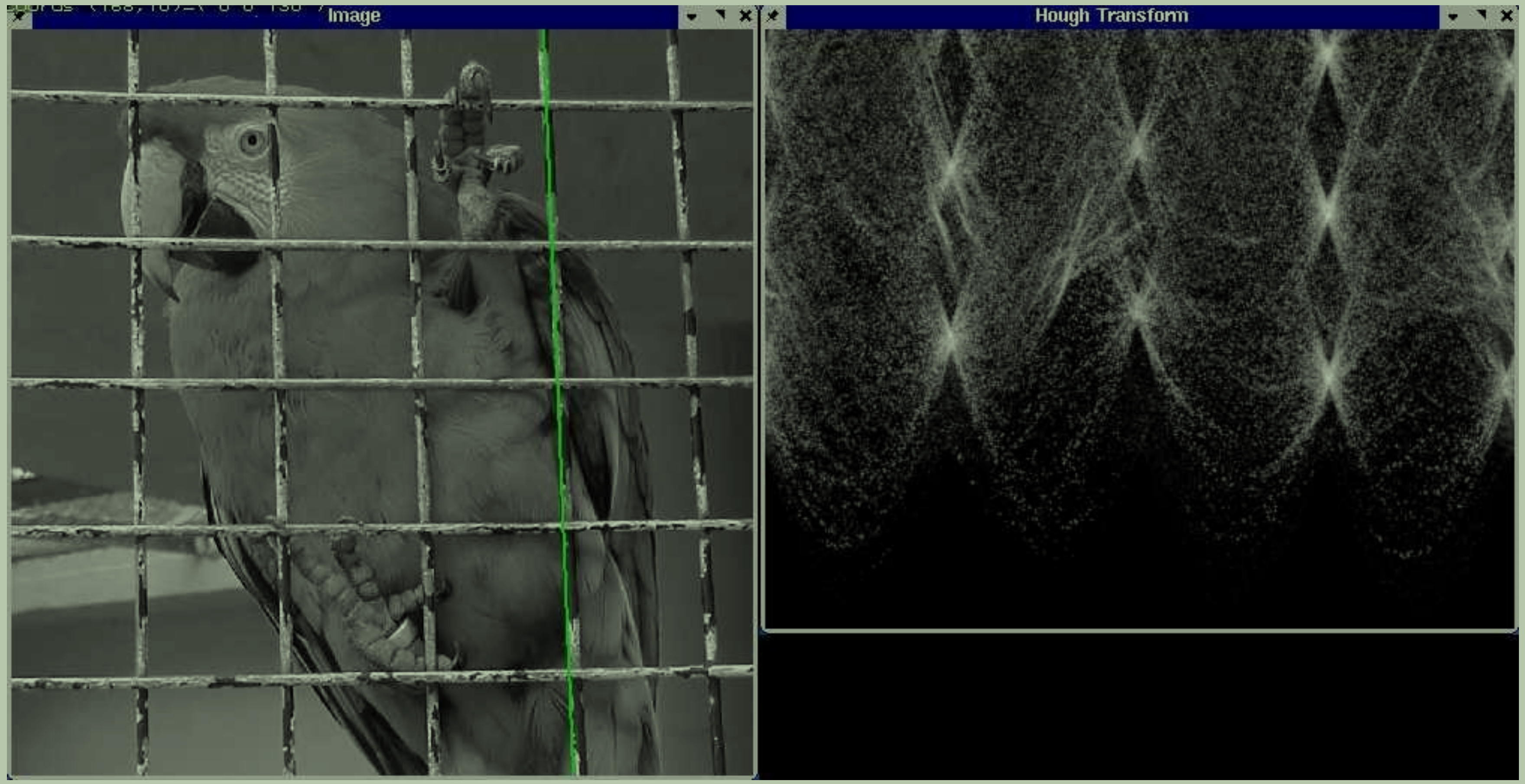
Hough Transform: 3) Votes → Lines

threshold + non-maximum suppression

- Find peaks and post-process

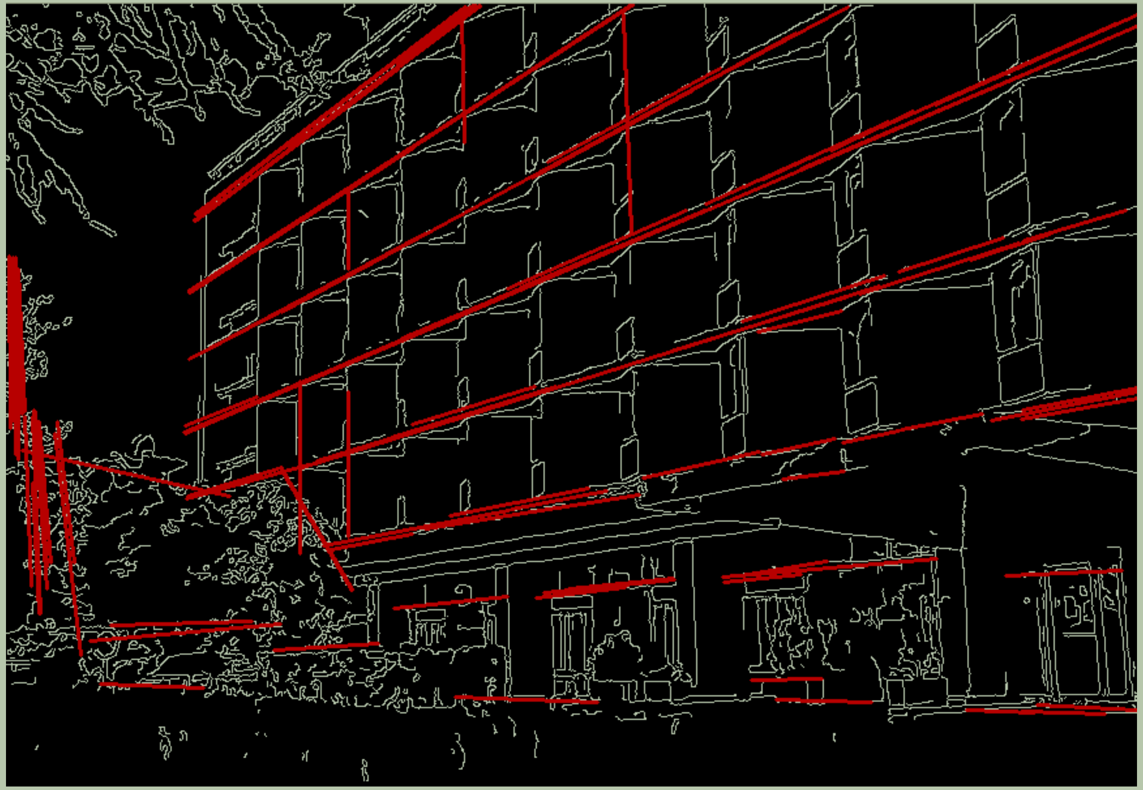


Hough Transform: Example



http://ostatic.com/files/images/ss_hough.jpg

Hough Transform: Example



Hough Transform: Pros and Cons

Pros:

- All points processed independently, so can cope with occlusion
- Some robustness to noise: noisy points are unlikely to contribute consistently to any single bin
- Can detect multiple instances of line/circle/ellipse in a single pass

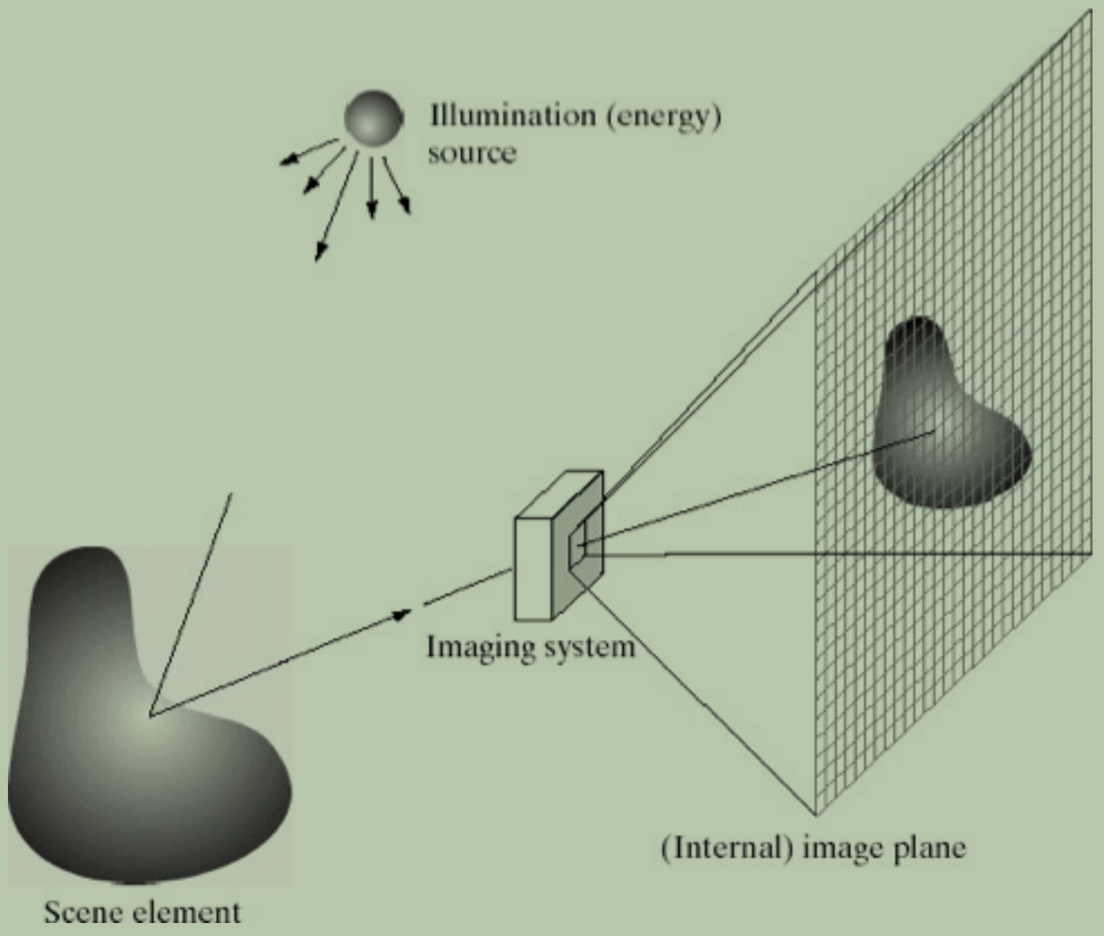
Cons:

- Complexity of search time increases exponentially with the number of model parameters
- Non-target shapes can produce spurious peaks in parameter space
- Quantization: hard to pick a grid size

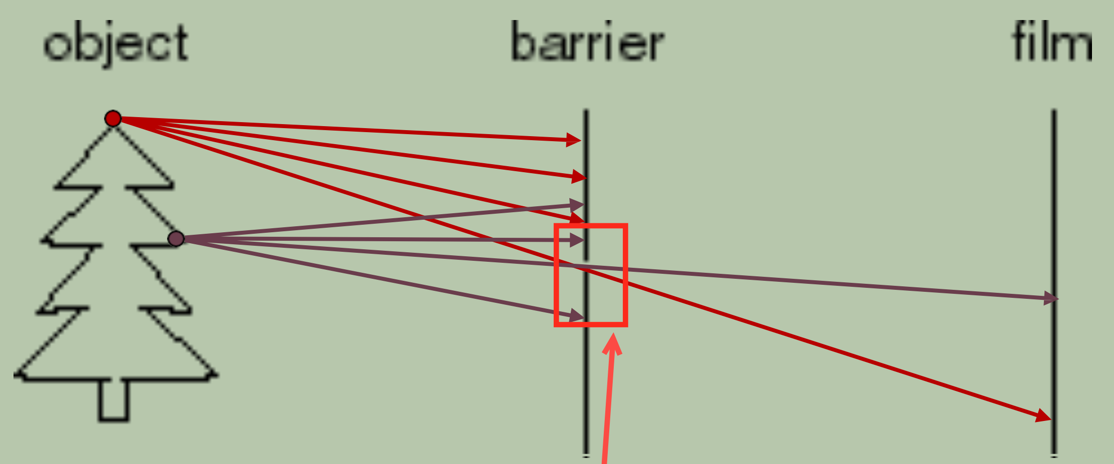
Image Formation (Review)

Pinhole Camera Model

Photometric Image Formation

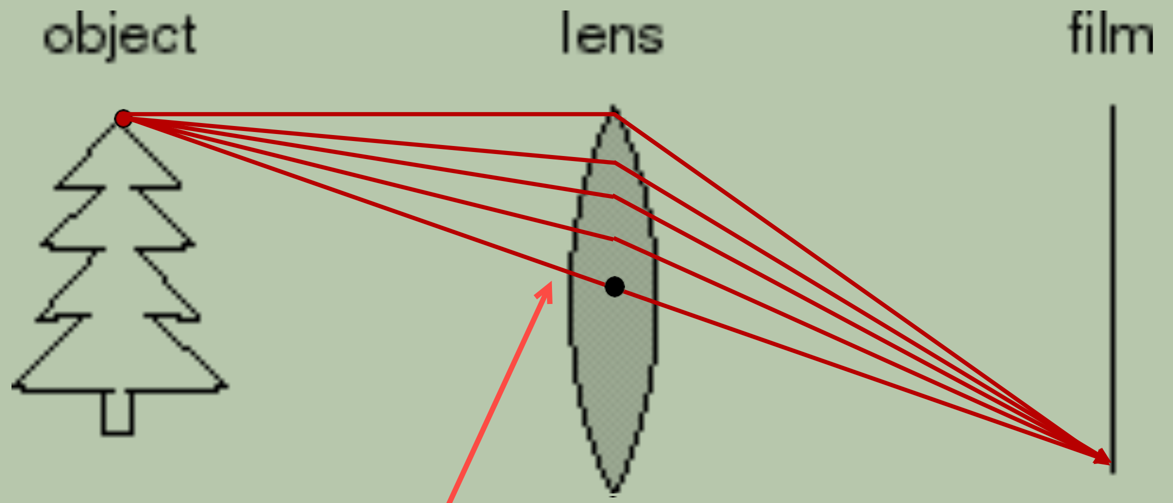


Pinhole Camera



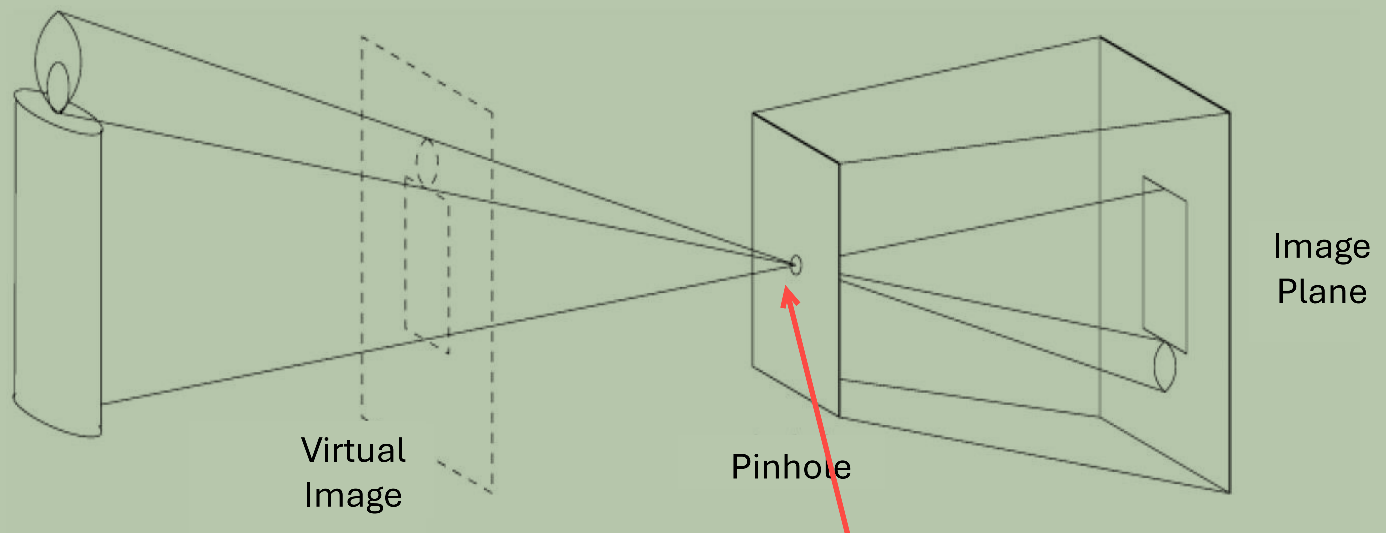
- Add a barrier to block off most of the rays
- This reduces blurring
- The opening is known as the aperture

Lensed Camera



- A lens focuses light onto the film: captures more light
- Rays passing through the centre do not deviate

Pinhole Camera Model



- Pinhole camera is an abstract model to approximate the imaging process: **perspective projection**
- If we treat the pinhole as a point, only one ray from any given point can enter the camera

Image Credit: Forsyth & Ponce

Camera Projection Matrix and Single View Geometry

Homogeneous Coordinates

from Cartesian to homogeneous

- To homogeneous: $(x, y) \rightarrow (x, y, 1)$ $(x, y, z) \rightarrow (x, y, z, 1)$
- From homogeneous: $(x, y, w) \rightarrow (x/w, y/w)$ $(x, y, z, w) \rightarrow (x/w, y/w, z/w)$

from homogeneous to Cartesian

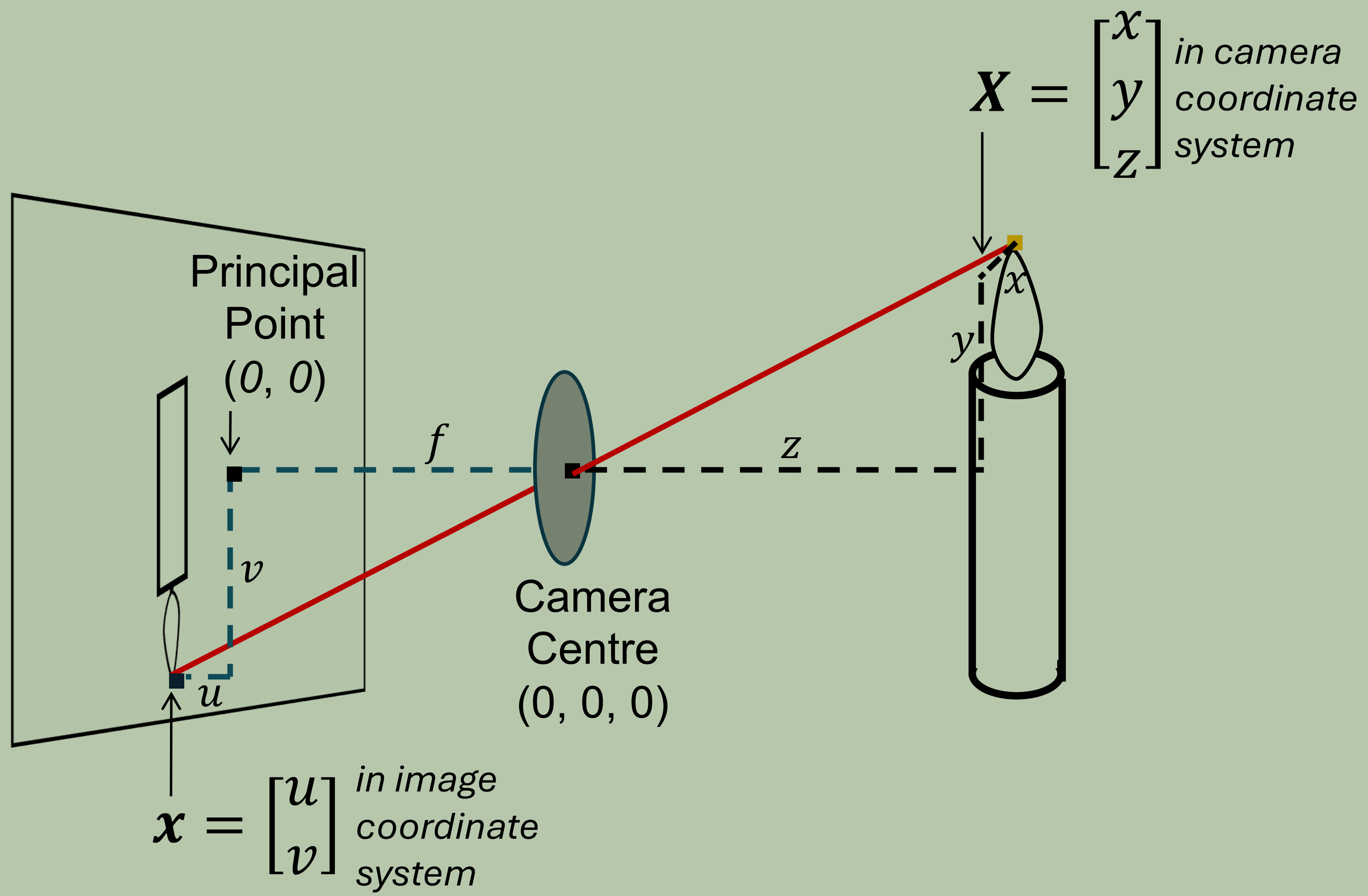
- Invariant to scaling: $k\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} kx \\ ky \\ kw \end{bmatrix} \Rightarrow \begin{bmatrix} \frac{kx}{kw} \\ \frac{ky}{kw} \end{bmatrix} = \begin{bmatrix} \frac{x}{w} \\ \frac{y}{w} \end{bmatrix}$

Homogeneous Cartesian

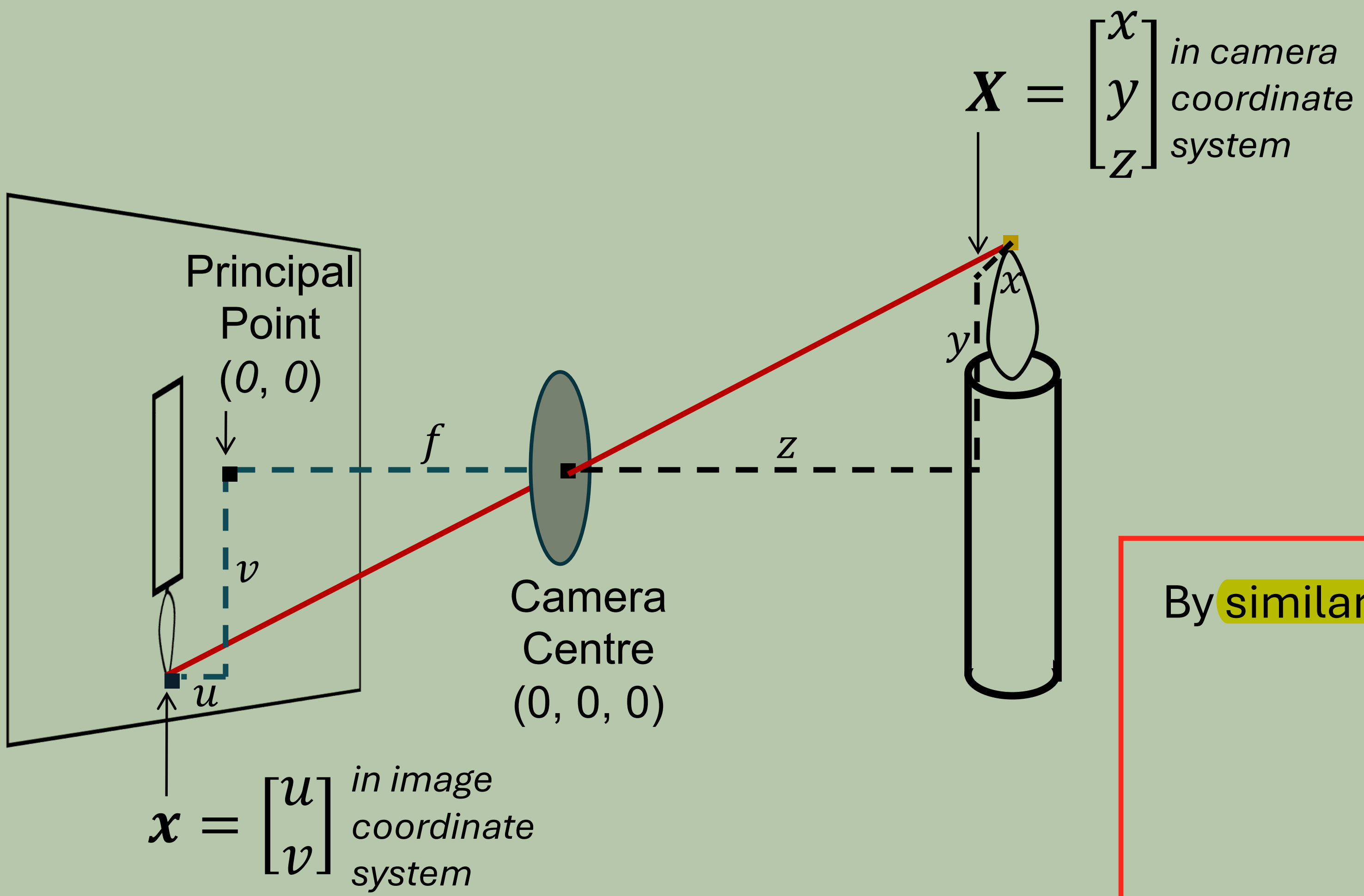
- A **point** in Cartesian coordinates is a **ray** in homogeneous coords

k in (-inf, +inf)

Perspective Projection



Perspective Projection



By similar triangles:

$$u = f\frac{x}{z}$$

$$v = f\frac{y}{z}$$

Perspective Projection

- $(x, y, z) \mapsto (\frac{fx}{z}, \frac{fy}{z})$ In Catesian coord: 3D → 2D
- Using homogeneous coordinates:
  2D: $(\frac{fx}{z}, \frac{fy}{z}) \mapsto (\frac{fx}{z}, \frac{fy}{z}, 1) = (fx, fy, z)$
  3D: $(x, y, z) \mapsto (x, y, z, 1)$
  Cart. Homo.
- Linear projection in homogeneous coordinates!

In homogeneous coord:
3D → 2D

$$\overset{\text{3D}}{\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}} \mapsto \overset{\text{2D}}{\boxed{\begin{pmatrix} fx \\ fy \\ z \end{pmatrix}}} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} \overset{\text{3D}}{\boxed{\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}}}$$

2D projected coord. in homo. coord. 3D point coord. in homo. coord.

Perspective Projection

[fx/z, fy/z].T

Cart. to Homo. for 2D

[x, y, z].T

Homo. to Cart. for 3D

2D in homo. coord.

$$\begin{pmatrix} fx \\ fy \\ z \end{pmatrix} = \boxed{\begin{bmatrix} f & & \\ & f & \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix}} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

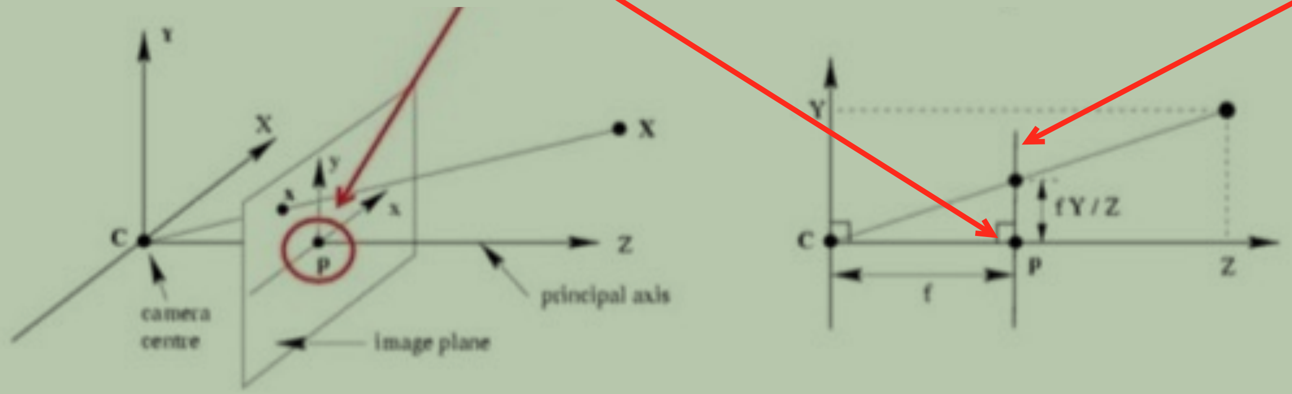
3D in homo. coord.

$$\Leftrightarrow \mathbf{x} = \mathbf{P}\mathbf{X} \;\; \text{with} \; \boxed{\mathbf{P} = \mathrm{diag}(f, f, 1)\,[\,\mathbf{I} \mid 0\,]}$$

$$= \text{3×4 homogeneous camera projection matrix}$$

Principal Point

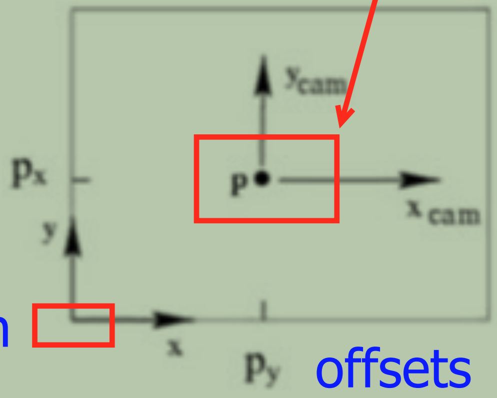
- The point where the principal axis intersects with the image plane



Principal Point Offset

https://au.mathworks.com/help/vision/ug/camera-calibration.html

- So far, we have assumed that the origin of points in the image plane is at principal point
- However, origin is often elsewhere (e.g., at image corner)
- ~~Inhomogeneous~~ Cartesian.: $\overset{\text{3D}}{(X, Y, Z)} \mapsto \overset{\text{2D}}{(fX/Z + p_x, fY/Z + p_y)}$
- Homogeneous: $\overset{\text{3D}}{\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}} \mapsto \overset{\text{2D}}{\begin{pmatrix} fX + Zp_x \\ fY + Zp_y \\ Z \end{pmatrix}} = \overset{\text{proj. mat.}}{\begin{bmatrix} f & & p_x & 0 \\ & f & p_y & 0 \\ & & 1 & 0 \end{bmatrix}} \overset{\text{3D}}{\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}}$



offsets

new origin

offsets

Camera (Intrinsic) Calibration Matrix

x_img (homo)

x_world (homo)

$$\begin{pmatrix} fX + Zp_x \\ fY + Zp_y \\ Z \end{pmatrix} = \begin{bmatrix} f & & p_x & 0 \\ & f & p_y & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

calibration matrix

$$= \boxed{\begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix}} \begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

$$= \mathbf{K}[I \mid 0]\mathbf{X}$$

$$\mathbf{K} = \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix} =$$

camera calibration matrix

Rectangular Pixels: CCD Cameras


- Charge-Coupled Device (CCD)
- From image plane to pixel coordinates

mersured by # pixels

$$\mathbf{K} = \begin{bmatrix} \alpha_x & & x_0 \\ & \alpha_y & y_0 \\ & & 1 \end{bmatrix} = \begin{bmatrix} m_x & & \\ & m_y & \\ & & 1 \end{bmatrix} \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix}$$

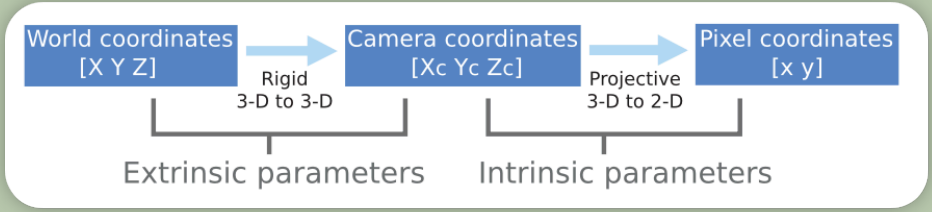
$$\begin{cases} x_0 = m_x p_x \\ y_0 = m_y p_y \\ \alpha_x = m_x f \\ \alpha_y = m_y f \end{cases}, \text{with} \begin{cases} m_x = \text{\# pixels / unit distance along x} \\ m_y = \text{\# pixels / unit distance along y} \end{cases}$$

offsets × #pixels per offset

on the image plane

https://au.mathworks.com/help/vision/ug/camera-calibration.html

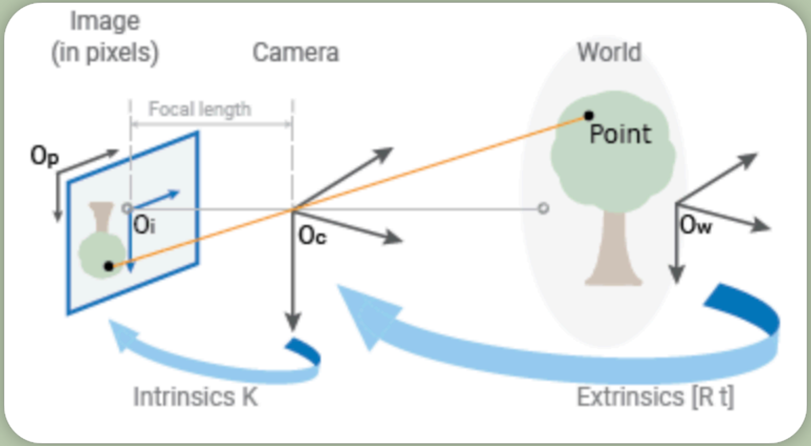
Camera Parameters



- So far: **intrinsic** camera parameters
  - cam → img
  - How to map spatial directions to pixel coordinates
  - f, p_x, p_y (offsets), m_x, m_y
  - Focal length, principal point, pixel width/height

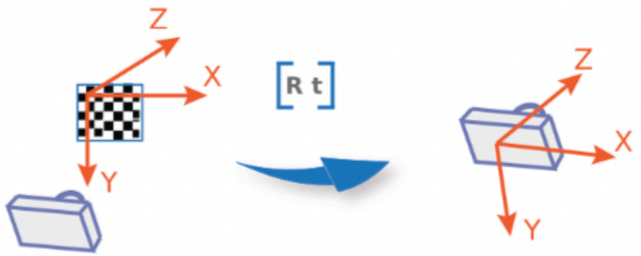
$$\mathbf{K} = \begin{bmatrix} \alpha_x & & x_0 \\ & \alpha_y & y_0 \\ & & 1 \end{bmatrix} = \begin{bmatrix} m_x & & \\ & m_y & \\ & & 1 \end{bmatrix} \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix}$$

- What else? **Extrinsic** camera parameters
  - world → cam
  - How to transform a 3D point into the camera frame
  - Depends on the camera rotation and translation



Extrinsic Parameters

The extrinsic parameters consist of a rotation, R, and a translation, t. The origin of the camera's coordinate system is at its optical center and its x- and y-axis define the image plane.

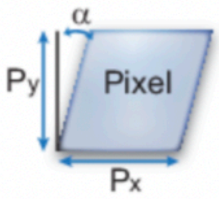


Intrinsic Parameters

The intrinsic parameters include the focal length, the optical center, also known as the *principal point*, and the skew coefficient. The camera intrinsic matrix, K, is defined as:

$$\begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

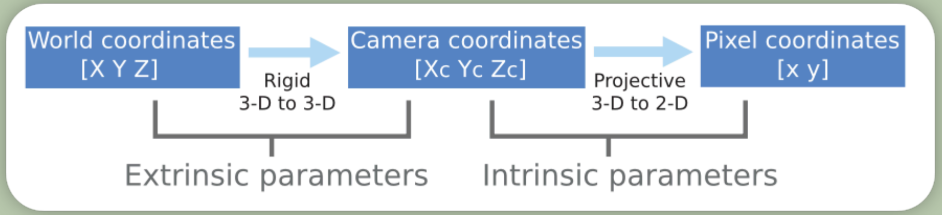
The pixel skew is defined as:



Skew

$[c_x \quad c_y]$ — Optical center (the principal point), in pixels.
$(f_x, \quad f_y)$ — Focal length in pixels. $f_x = F/p_x$ $f_y = F/p_y$ F — Focal length in world units, typically expressed in millimeters. $(p_x, \quad p_y)$ — Size of the pixel in world units.
s — Skew coefficient, which is non-zero if the image axes are not perpendicular. $s = f_x \tan \alpha$

Extrinsic Camera Parameters

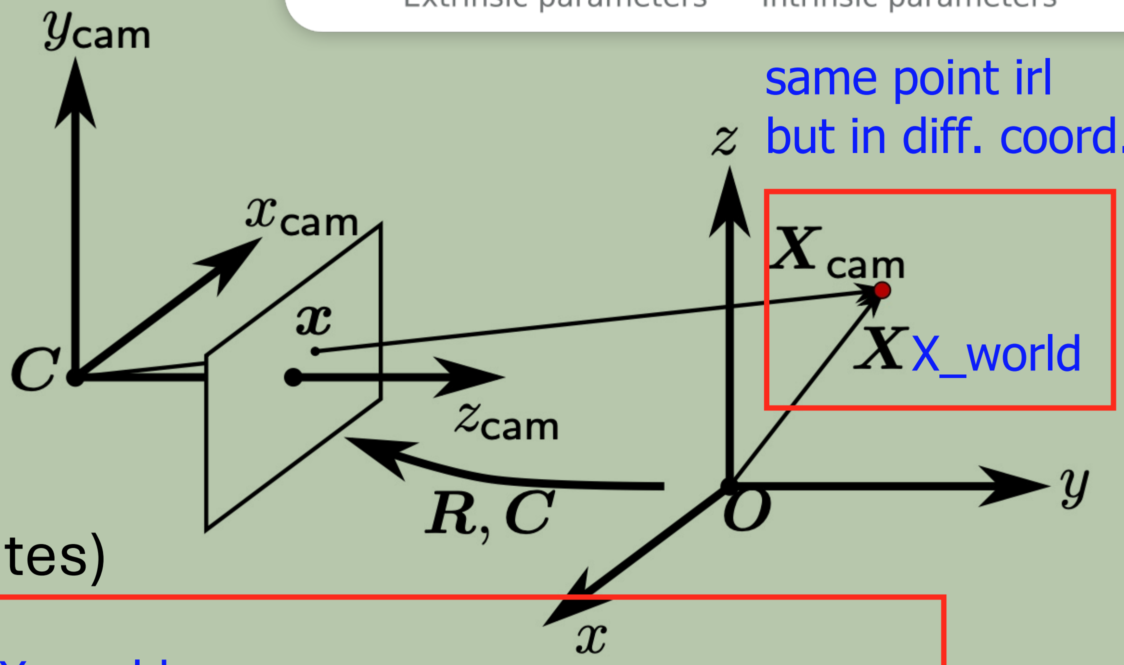


- Camera rotation and translation
- **R**: Rotation matrix
  - Orthogonal + unit determinant
  - SO(3)
- **C:** Camera centre (vector) C_world
  - Location of the camera in the world coordinate system

(X_world - C_world)

- $\boldsymbol{X}_{cam} = \mathbf{R}(\boldsymbol{X} - \mathbf{C})$ (in camera coordinates)

same point irl
but in diff. coord. sys.



X_world

- $\boldsymbol{X}_{cam} = \begin{bmatrix} \mathbf{R} & -\mathbf{RC} \\ 0 & 1 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{bmatrix} \mathbf{R} & -\mathbf{RC} \\ 0 & 1 \end{bmatrix} \boldsymbol{X}$ (in homogeneous coordinates)

Rotation About Coordinate Axes in 3D

- Express 3D rotation as series of rotations around coordinate coordinate axes by angles α, β, γ
- The overall rotation is the product of these elementary rotations:
- $\mathbf{R} = \mathbf{R}_x \mathbf{R}_y \mathbf{R}_z$ rotation matrix
- They describe clockwise rotations

$$\mathbf{R}_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix}$$

$$\mathbf{R}_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}$$

$$\mathbf{R}_z(\gamma) = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

https://en.wikipedia.org/wiki/Rotation_matrix#In_three_dimensions

Complete Camera Matrix

3*1 (homo) X_img   4*1 (homo) X_cam

- $\mathrm{x} = \mathbf{K}[\,\mathbf{I} \mid 0\,]\mathbf{X}_{\text{cam}}$

3*3 K (intrinsic)

$$= \boxed{\begin{bmatrix} \alpha_x & & x_0 \\ & \alpha_y & y_0 \\ & & 1 \end{bmatrix}} [\,\mathbf{I} \mid 0\,] \boxed{\begin{bmatrix} \mathbf{R} & -\mathbf{RC} \\ 0 & 1 \end{bmatrix} \mathbf{X}}$$

4*1 (homo) X_world

3*3 3*4

$$= \underbrace{\mathbf{K}}_{\substack{\text{Camera} \\ \text{intrinsics}}} \underbrace{\mathbf{R}[\,\mathbf{I} \mid -\mathbf{C}\,]}_{\substack{\text{Camera} \\ \text{extrinsics}}} \mathbf{X}$$

4*1 (homo) X_world

3*3   3*4

$$\begin{cases} x_0 = m_x p_x \\ y_0 = m_y p_y \\ \alpha_x = m_x f \\ \alpha_y = m_y f \end{cases}, \text{with} \begin{cases} m_x = \#\text{ pixels / unit distance along x} \\ m_y = \#\text{ pixels / unit distance along y} \end{cases}$$

offsets × #pixels per offset

on the image plane

$$= \mathbf{P}\mathbf{X}$$

4*1 (homo) X_cam

3x4 projective camera matrix

Camera (Intrinsic) Calibration Matrix

- $\mathbf{K}$ is a 3×3 upper triangular matrix, the "camera calibration matrix"

$$\mathbf{K} = \begin{bmatrix} \alpha_x & & x_0 \\ & \alpha_y & y_0 \\ & & 1 \end{bmatrix}$$ measured by # pixels

- Four parameters:
  - The scaling in the image x and y directions, α_x and α_y
  - The principal point (x_0, y_0), which is the point where the optical axis intersects the image plane
- The aspect ratio is α_y / α_x

$$\begin{cases} x_0 = m_x p_x \\ y_0 = m_y p_y \\ \alpha_x = m_x f \\ \alpha_y = m_y f \end{cases}, \text{with} \begin{cases} m_x = \text{\# pixels / unit distance along x} \\ m_y = \text{\# pixels / unit distance along y} \end{cases}$$

offsets × #pixels per offset

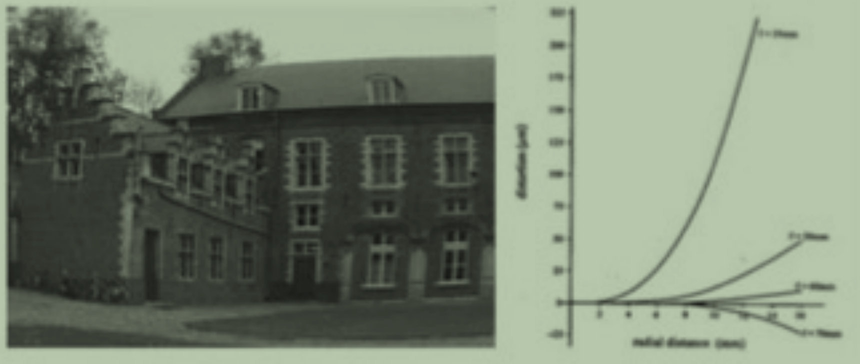
on the image plane

https://au.mathworks.com/help/vision/ug/camera-calibration.html#bu0nj3f

Radial Lens Distortion

Radial distortion occurs when light rays bend more near the edges of a lens than they do at its optical center. The smaller the lens, the greater the distortion.

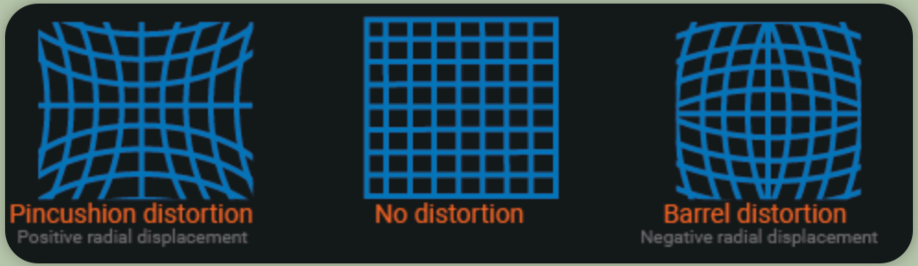
- There is no such thing as a perfect lens
- Straight lines are no longer straight!

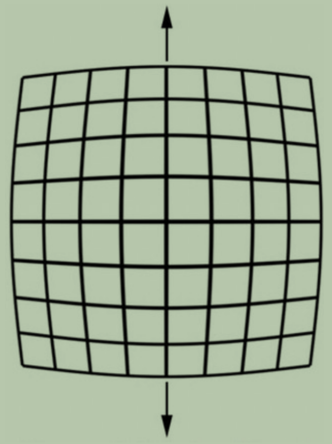


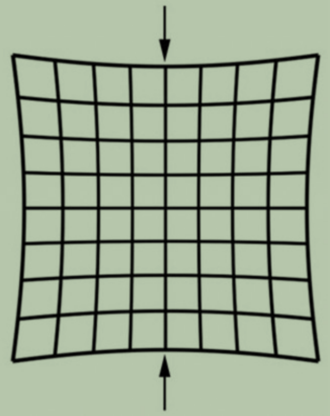
http://foto.hut.fi/opetus/260/luennot/11/atkinson_6-11_radial_distortion_zoom_lenses.jpg

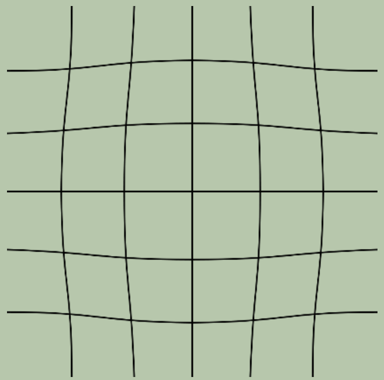
Radial Lens Distortion

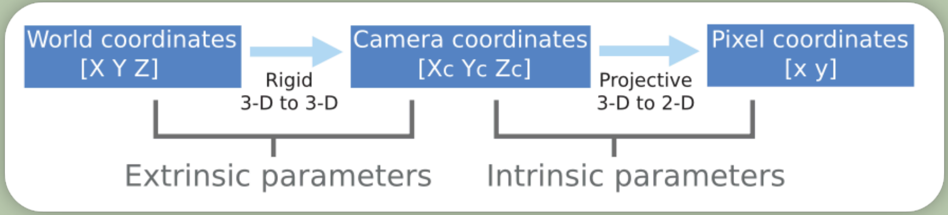
- Due to spherical lenses (cheaper)
  - Barrel distortion
    - Image magnification decreases with distance from optical axis
  - Pincushion distortion
    - Image magnification increases with distance from optical axis
  - Mustache distortion
    - A mixture of both types











Radial Lens Distortion

- Model for radial distortion:
  - Change based on distance of point on image plane from principal point
  - X_img calibration mat.
  - If $\mathbf{x} = \mathbf{P}\mathbf{X}_{\text{world}} = \mathbf{K}\mathbf{R}[\,\mathbf{I} \mid -\mathbf{C}\,]\mathbf{X}_{\text{world}} = \mathbf{K}[\,\mathbf{I} \mid 0\,]\mathbf{X}_{\text{cam}}$

X_cam (homo)

$$= \mathbf{K}\begin{bmatrix} X/Z \\ Y/Z \\ 1 \end{bmatrix} = \mathbf{K}\begin{bmatrix} x_{cam} \\ y_{cam} \\ 1 \end{bmatrix}$$

x

X_world

$$\boldsymbol{X}_{cam} = \begin{bmatrix} \mathbf{R} & -\mathbf{RC} \\ 0 & 1 \end{bmatrix}\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{bmatrix} \mathbf{R} & -\mathbf{RC} \\ 0 & 1 \end{bmatrix}\boldsymbol{X}$$ (in homogeneous coordinates)

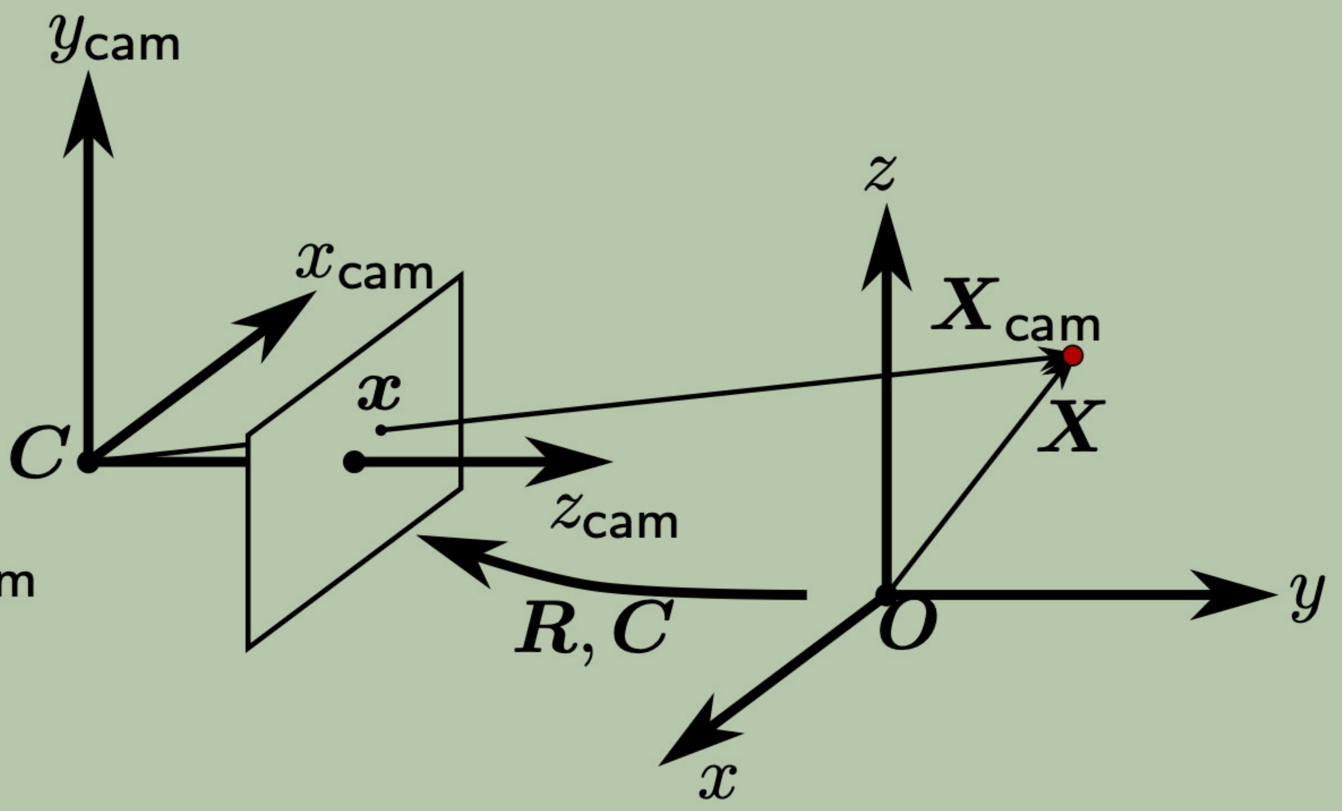
we change to $\mathbf{x} = \mathbf{K}\begin{bmatrix} r & & \\ & r & \\ & & 1 \end{bmatrix}[\mathbf{I} \mid 0]\mathbf{X}_{\text{cam}}$

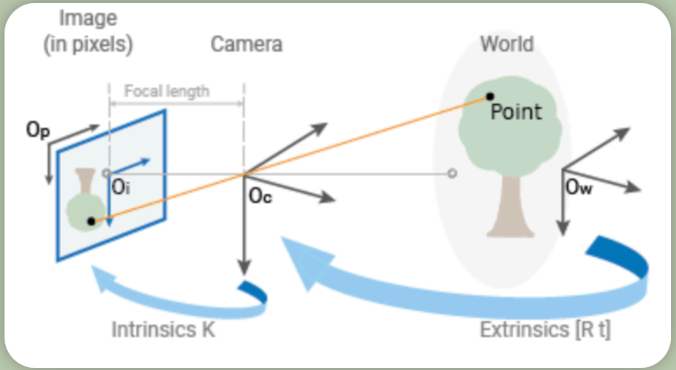
with $r = 1 + k_1(x_{cam}^2 + y_{cam}^2) + k_2(x_{cam}^2 + y_{cam}^2)^2$

x_cam and y_cam are from homo X_cam when Z=1 (X, Y devided by Z)

Summary: Camera Projection Matrix

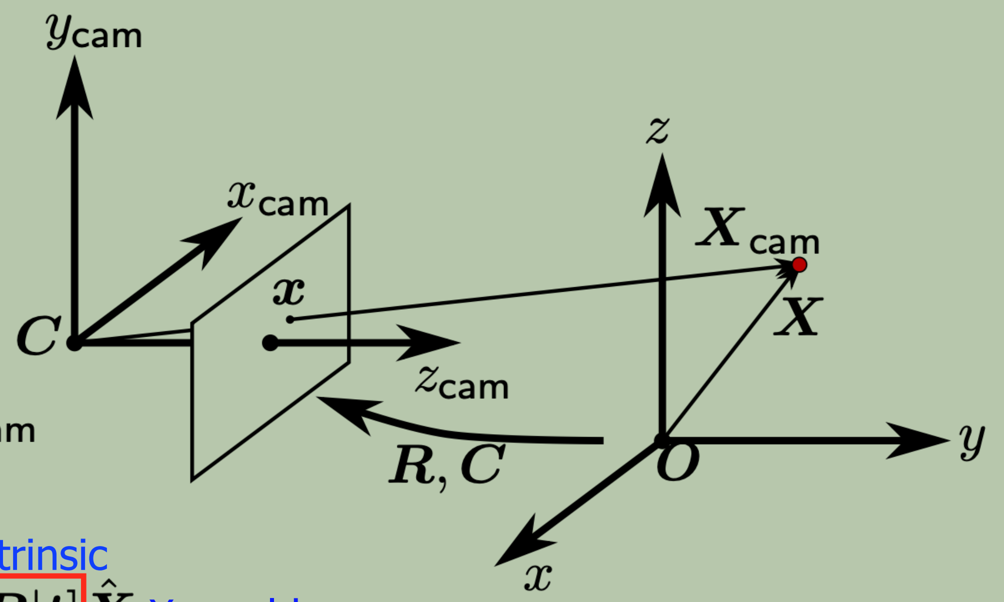
- image plane
- camera centre $\boldsymbol{C}$
- principal axis z_{cam}
- image coordinates $\boldsymbol{x}$
- world coordinates $\boldsymbol{X}$
- camera coordinates $\boldsymbol{X}_{\text{cam}}$





Summary: Camera Projection Matrix

- image plane
- camera centre $\boldsymbol{C}$
- principal axis z_{cam}
- image coordinates $\boldsymbol{x}$
- world coordinates $\boldsymbol{X}$
- camera coordinates $\boldsymbol{X}_{\text{cam}}$



X_img $\hat{\boldsymbol{x}} = \boldsymbol{P}\hat{\boldsymbol{X}}$

intrinsic extrinsic

$$= \boxed{\boldsymbol{K}}\boldsymbol{R}[\boldsymbol{I}|-\boldsymbol{C}]\hat{\boldsymbol{X}} = \boldsymbol{K}\boxed{[\boldsymbol{R}|\boldsymbol{t}]}\hat{\boldsymbol{X}}$$ X_world

mersured by #pixels

$$= \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \hat{\boldsymbol{X}}$$ X_world

$$= \begin{bmatrix} m_x f_x & \gamma & m_x p_x \\ 0 & m_y f_y & m_y p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \hat{\boldsymbol{X}}$$ X_world

Next Week

- How to **calibrate** a perspective camera
- How to find the **P** matrix
- How to estimate camera focal length, etc.
- The DLT algorithm

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \propto \underbrace{\begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix}}_{P_{\{3\times 4\}}} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$