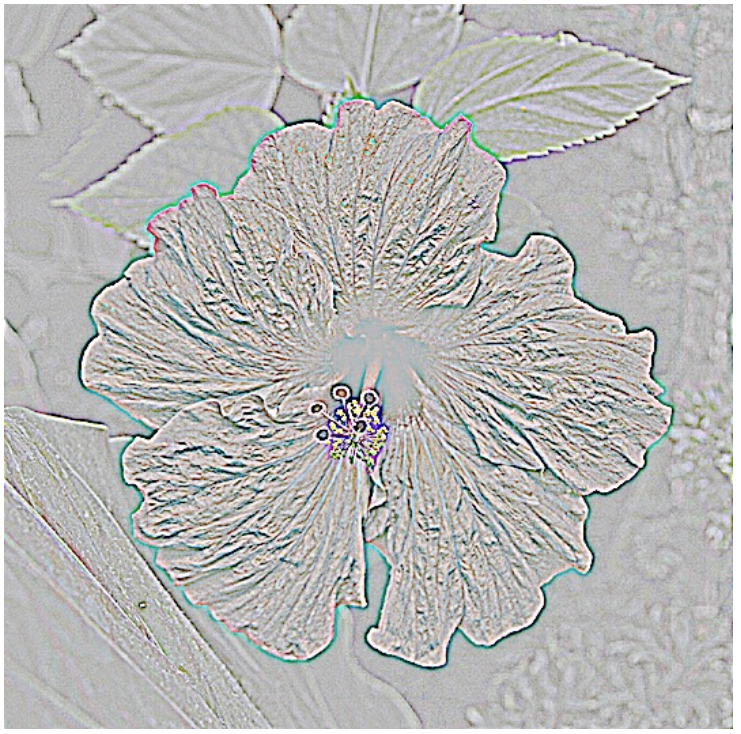
Review: Image Pyramid & Fourier Transform

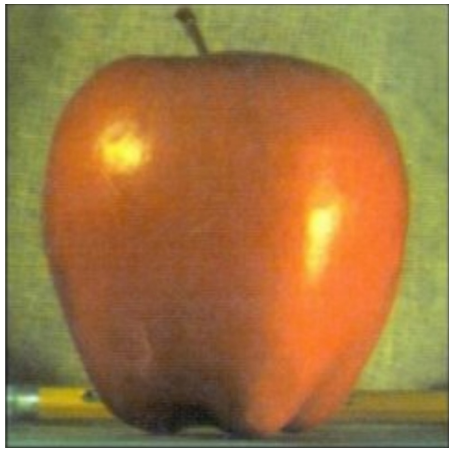
Gaussian vs Laplacian Pyramid

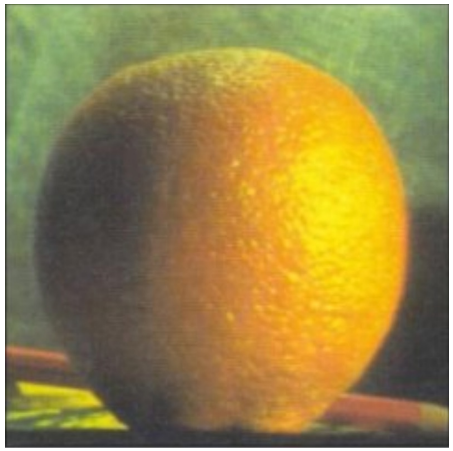
Shown in opposite order for space.



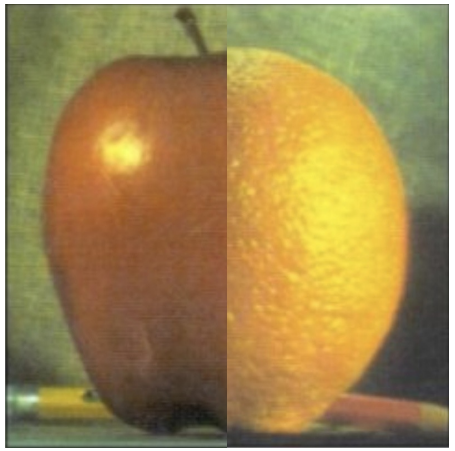
Which one takes more space to store?

Image Blending



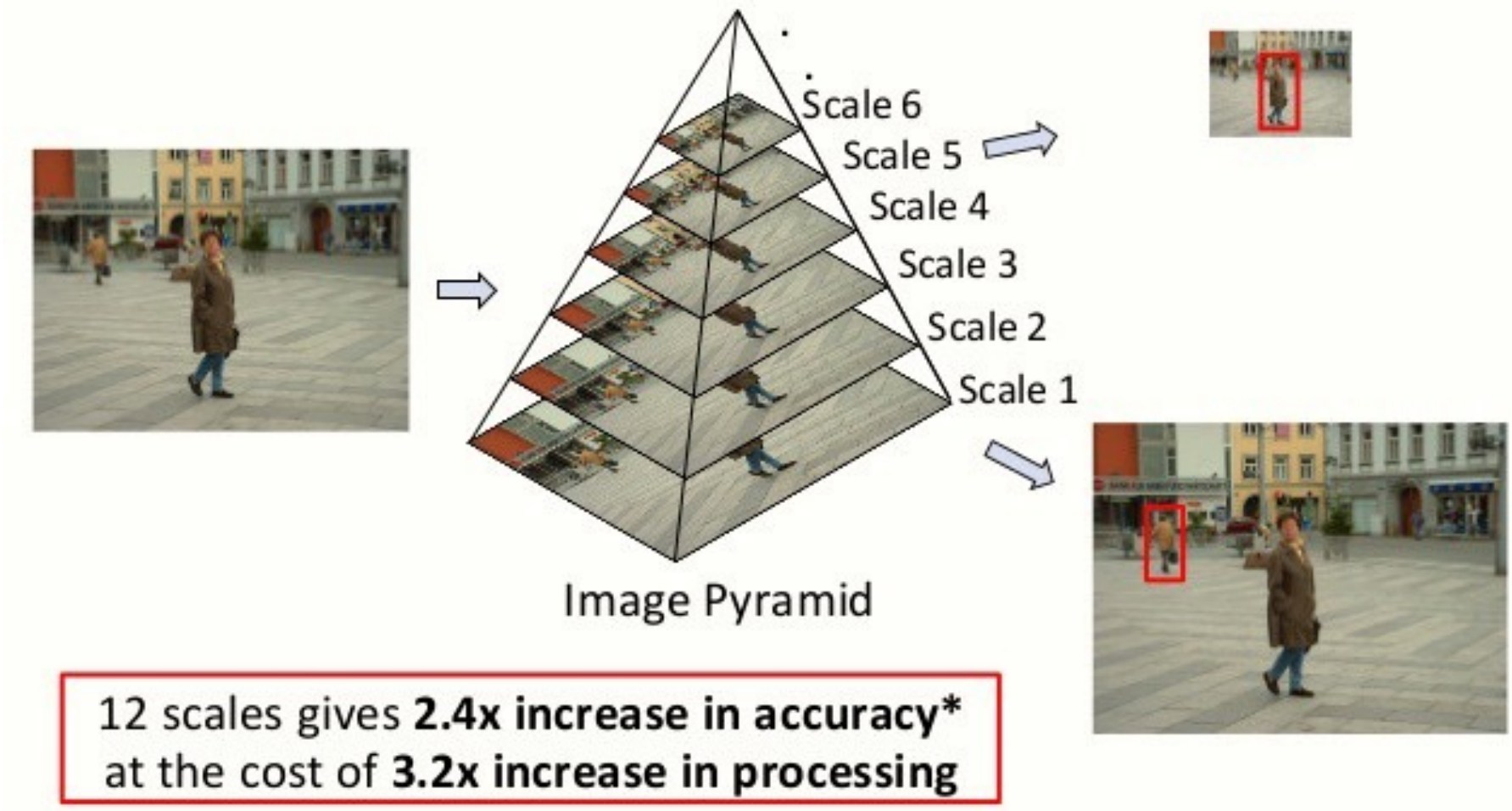


- Load the two images of apple and orange
- Find the Gaussian Pyramids for apple and orange
- From Gaussian Pyramids, find their Laplacian Pyramids
- Now join the left half of apple and right half of orange in each levels of Laplacian Pyramids
- From this joint image pyramids, reconstruct the original image.



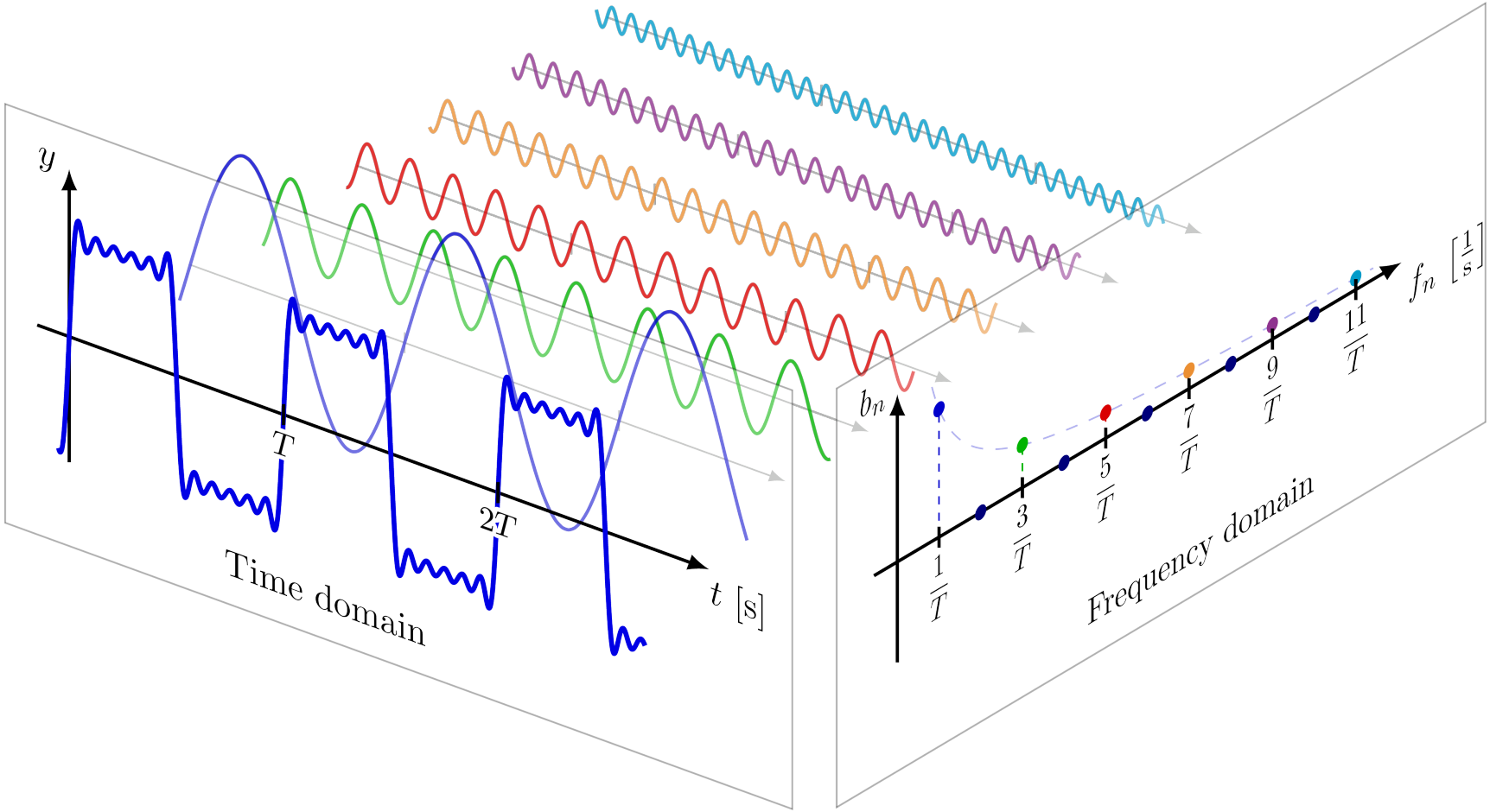
https://docs.opencv.org/3.4/dc/dff/tutorial_py_pyramids.html

Multi-scale Object Detection



12 scales gives **2.4x increase in accuracy*** at the cost of **3.2x increase in processing**

Fourier Transform



Fourier transform

The connection to the 'summation of sine waves' idea

$$F(k) = \frac{1}{N} \sum_{x=0}^{N-1} f(x) e^{-j2\pi kx/N} \qquad f(x) = \sum_{k=0}^{N-1} F(k) e^{j2\pi kx/N}$$

Euler's formula $e^{j\theta} = \cos\theta + j\sin\theta$

sum over frequencies

$$f(x) = \sum_{k=0}^{N-1} F(k) \left\{ \cos\left(2\pi kx/N\right) + j \sin\left(2\pi kx/N\right) \right\}$$

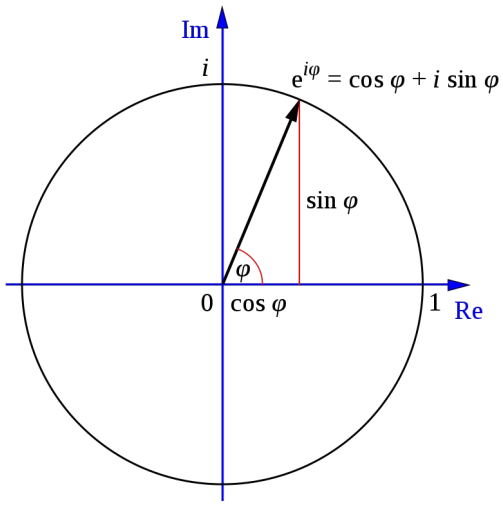
scaling parameter

wave components

2D-Fourier Transform

$$F[k,l] = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f[m,n] e^{-j2\pi \left(\frac{k}{M} m + \frac{l}{N} n \right)}$$

$$f[m,n] = \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} F[k,l] e^{j2\pi \left(\frac{k}{M} m + \frac{l}{N} n \right)}$$



2D-Fourier Transform

There is the **fft2** function in the **numpy** toolbox that can do a two-dimensional fast Fourier transform on the image.

To better visualize the output spectrogram, we need to move the center points of the four corners to the center of the matrix and do logarithmic transformation

```python
def my_fft(img):
    dft = np.fft.fft2(img)

    ## shift transform
    dft_shift = np.fft.fftshift(dft)

    ## calculate the magnitude
    magnitude_spectrum = np.abs(dft_shift)

    ## calculate the phase
    phase_spectrum = np.angle(dft_shift)

    return magnitude_spectrum, phase_spectrum
```

https://numpy.org/doc/stable/reference/generated/numpy.fft.fft2.html

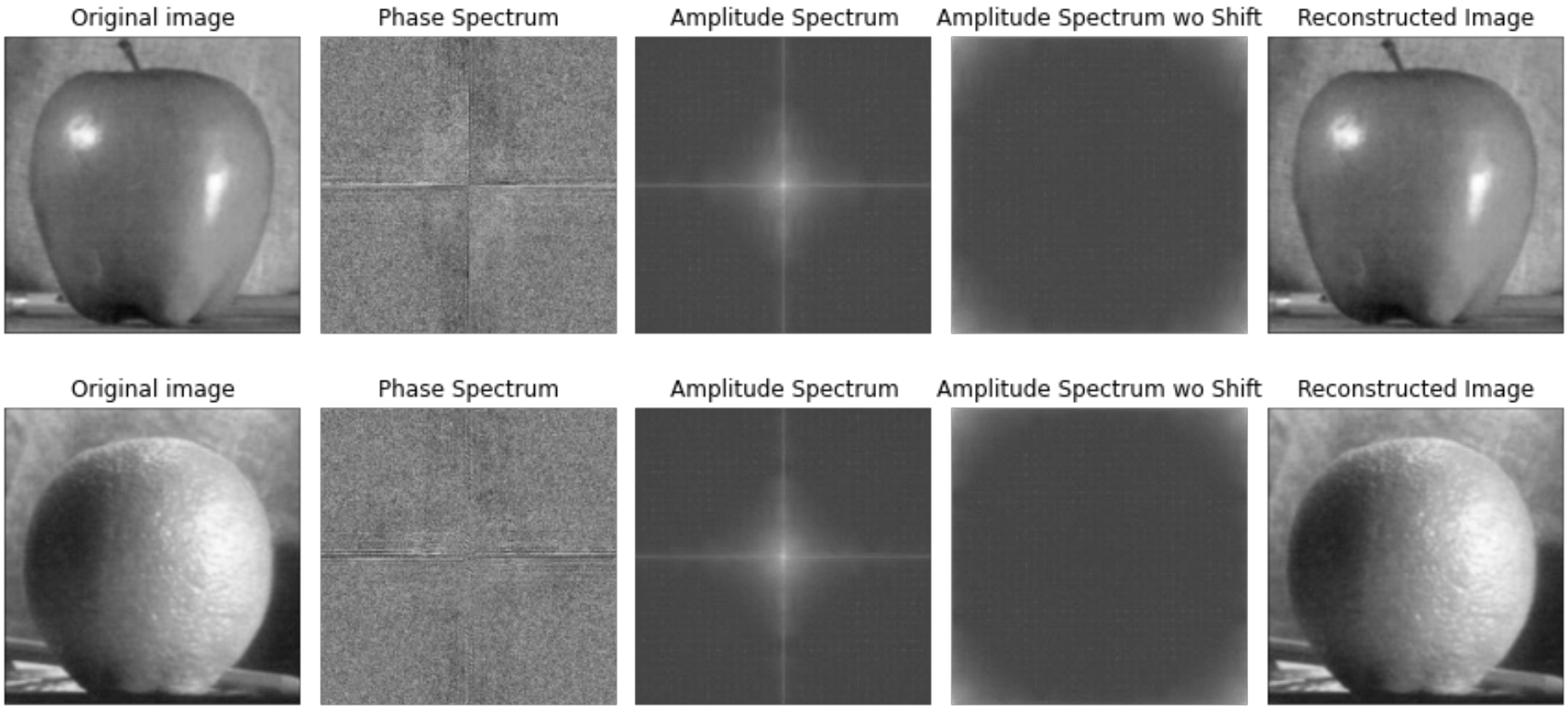
2D-Fourier Transform

There is the **ifft2** function in the **numpy** toolbox that can do a two-dimensional inverse Fourier transform from magnitude and phase.

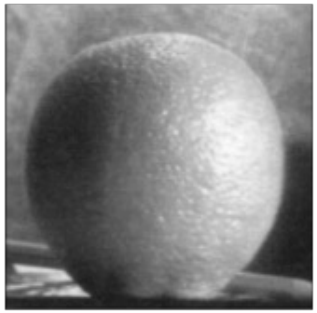
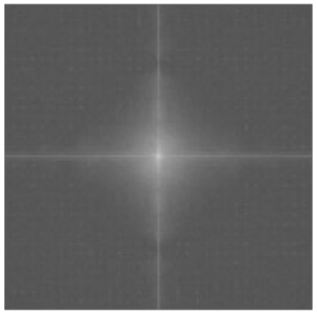
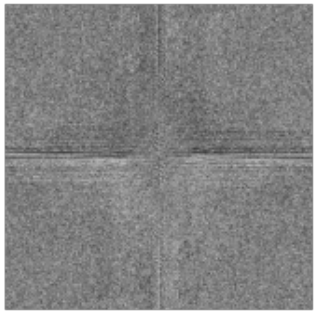
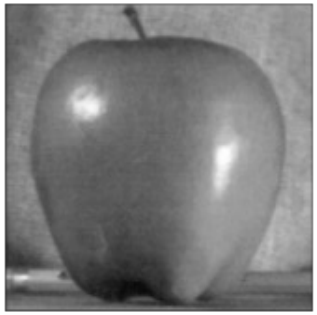
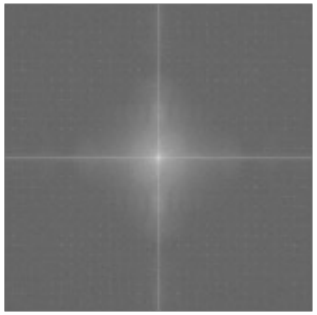
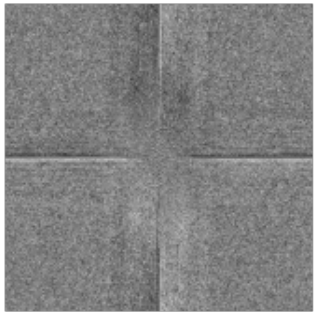
```
def my_ifft(img_m, img_p):
    ## integrate magnitude and phase
    img_mandp = img_m*np.e**(1j*img_p)

    img_mandp = np.uint8(np.abs(np.fft.ifft2(img_mandp)))

    img_mandp=img_mandp/np.max(img_mandp)*255
    return img_mandp
```

https://numpy.org/doc/stable/reference/generated/numpy.fft.fft2.html

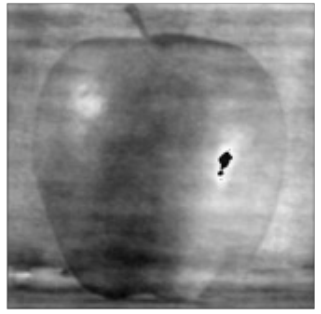
2D-Fourier Transform

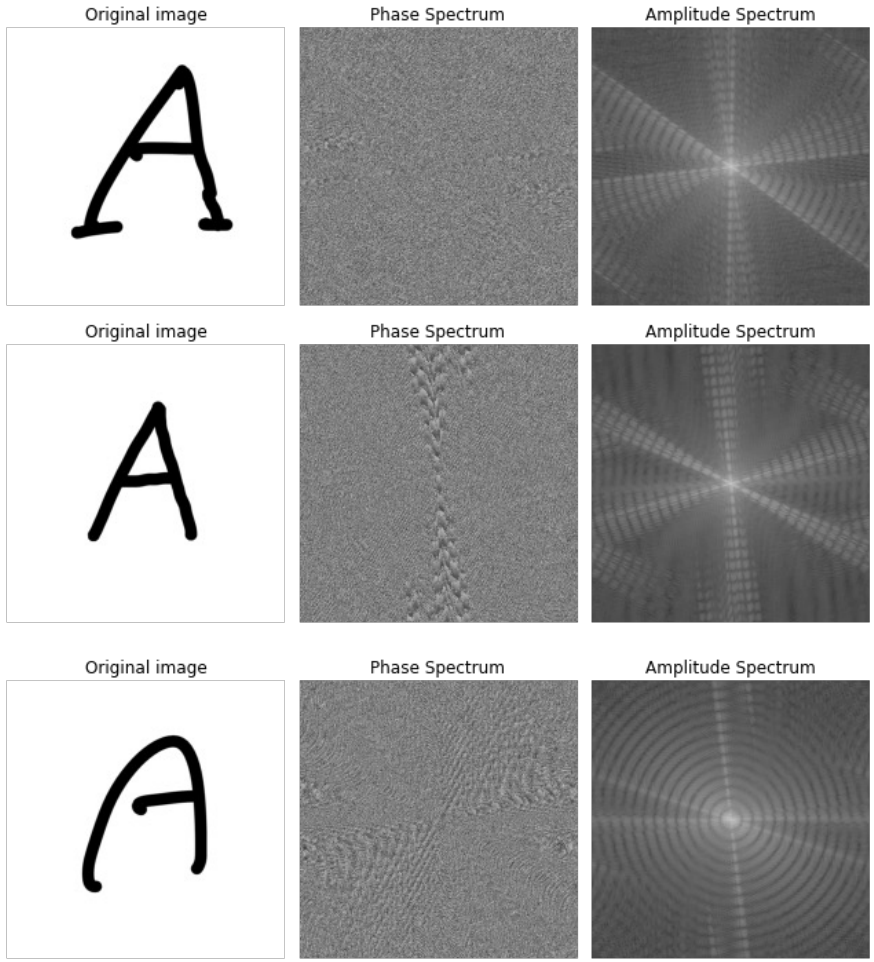


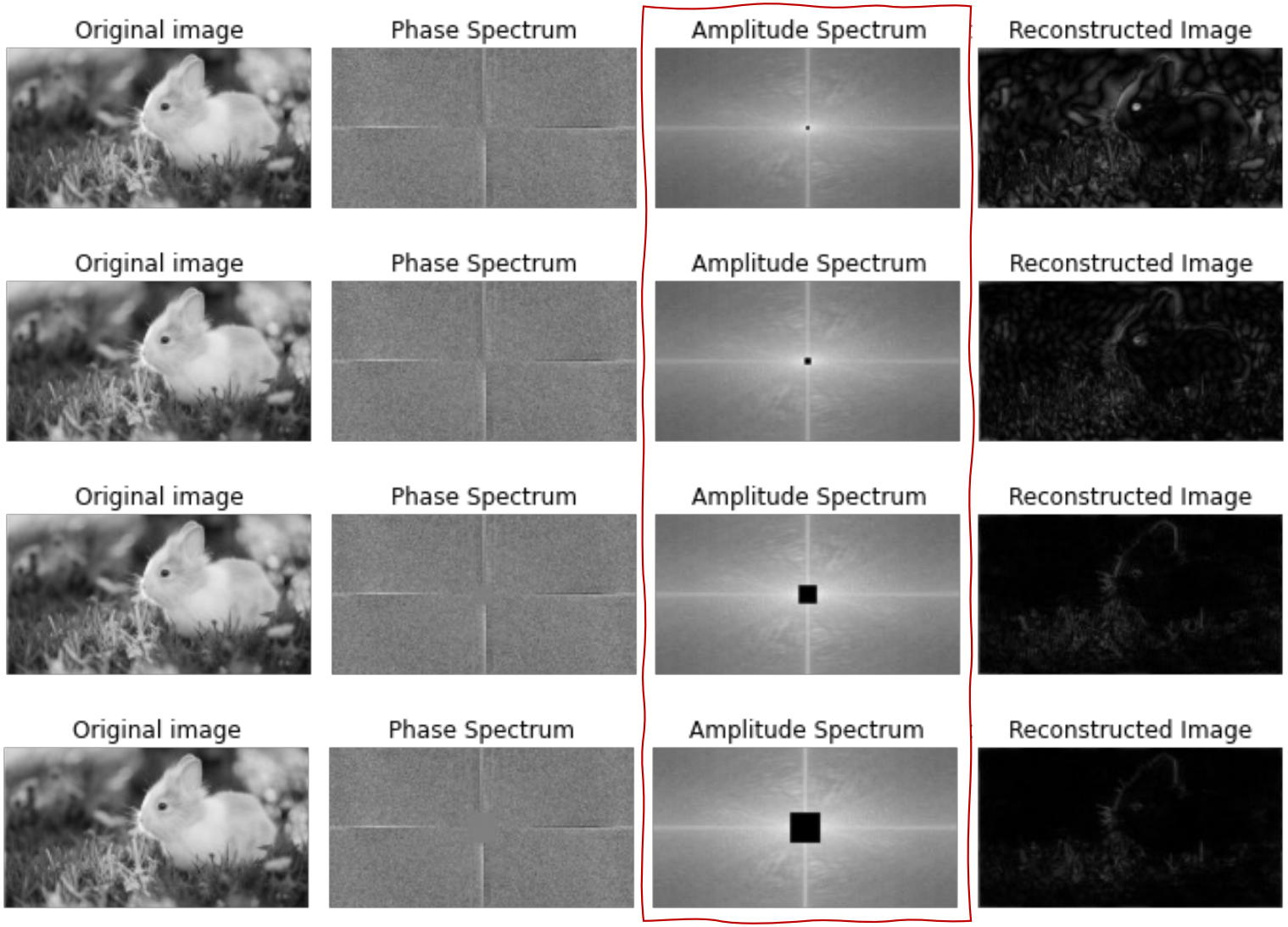
2D-Fourier Transform

original amplitude phase

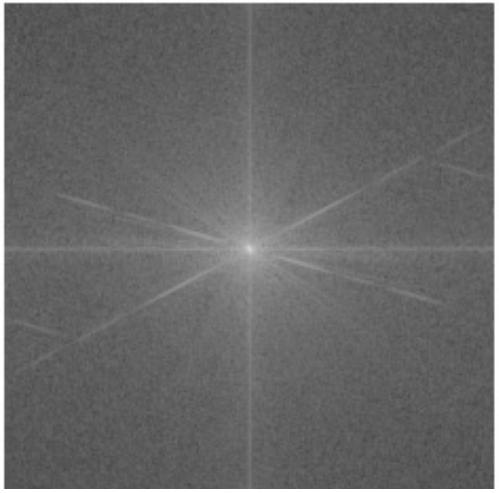




Original image
Phase Spectrum
Amplitude Spectrum
Original image
Phase Spectrum
Amplitude Spectrum
Original image
Phase Spectrum
Amplitude Spectrum
Original image
Phase Spectrum
Amplitude Spectrum
Original image
Phase Spectrum
Amplitude Spectrum
Original image
Phase Spectrum
Amplitude Spectrum

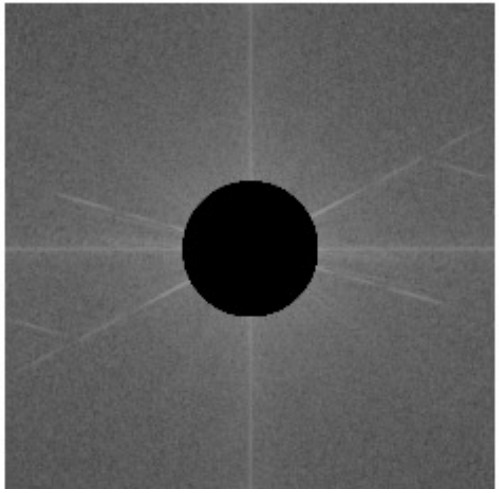

Original image
Phase Spectrum
Amplitude Spectrum
Reconstructed Image
Original image
Phase Spectrum
Amplitude Spectrum
Reconstructed Image
Original image
Phase Spectrum
Amplitude Spectrum
Reconstructed Image
Original image
Phase Spectrum
Amplitude Spectrum
Reconstructed Image

Original image

Log-scaled FFT magnitude



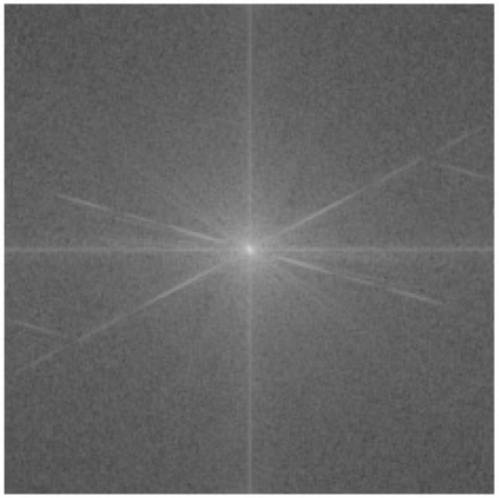
Low freqs removed



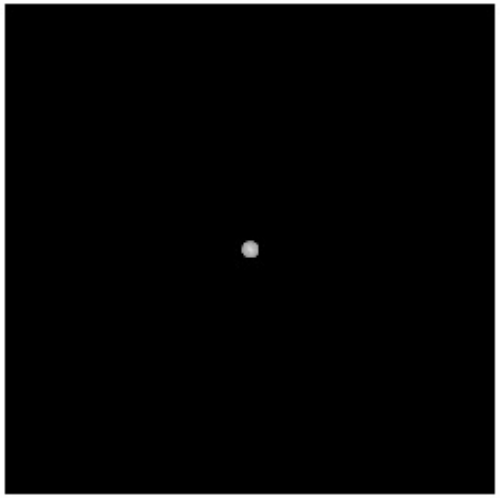
Visualized high frequencies

Original image

Log-scaled FFT magnitude



High freqs removed



Visualized low frequencies

Frequency-domain filtering

The convolution theorem

The Fourier transform of the convolution of two functions is the product of their Fourier transforms:

$$\mathcal{F}\{g * h\} = \mathcal{F}\{g\}\mathcal{F}\{h\}$$

The Fourier transform of the product of two functions is the convolution of the two Fourier transforms:

$$\mathcal{F}\{g \cdot h\} = \mathcal{F}\{g\} * \mathcal{F}\{h\}$$

Convolution in spatial domain is equivalent to multiplication in frequency domain!

$$g * h = \mathcal{F}^{-1}\left\{\mathcal{F}\{g\} \cdot \mathcal{F}\{h\}\right\}$$

$$g \cdot h = \mathcal{F}^{-1}\left\{\mathcal{F}\{g\} * \mathcal{F}\{h\}\right\}$$

What do we use convolution for?

Convolution for 1D continuous signals

Definition of linear shift-invariant filtering as convolution:

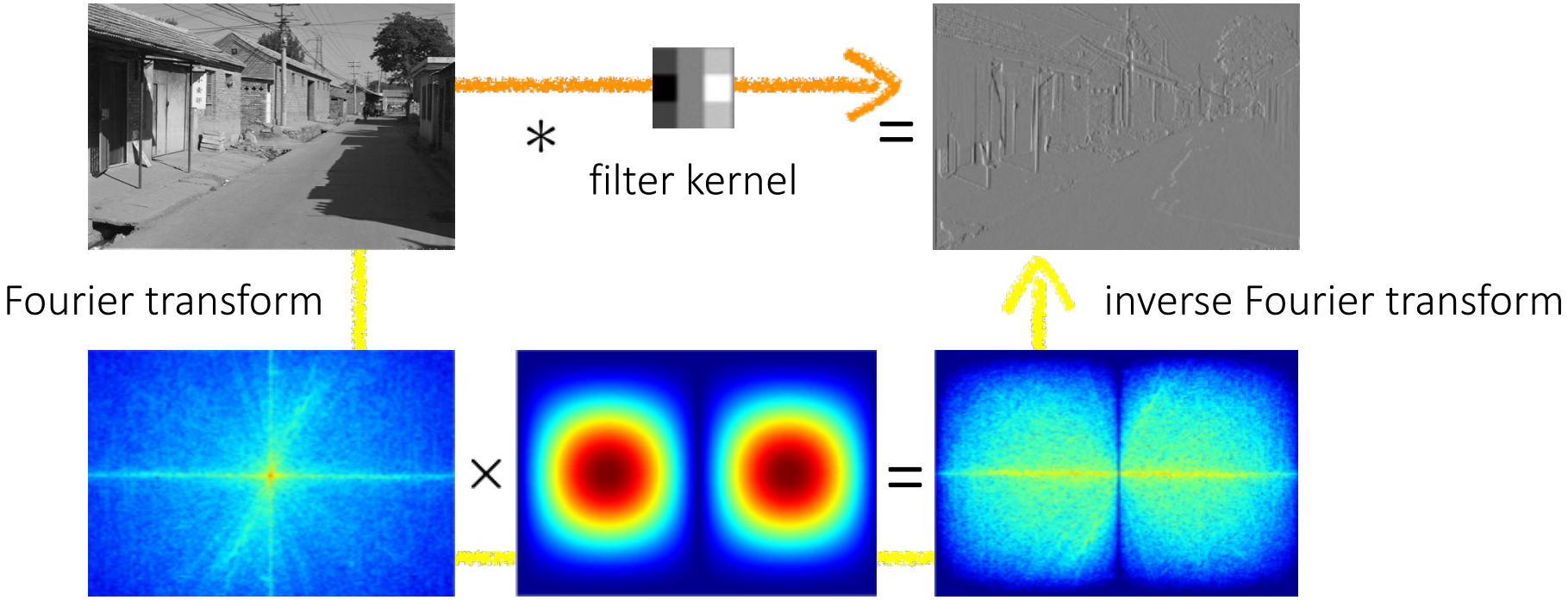
$$(f * g)(x) = \int_{-\infty}^{\infty} f(y)g(x-y)dy$$

filtered signal ↗ | ↖ filter | ↖ input signal

Using the convolution theorem, we can interpret and implement all types of linear shift-invariant filtering as multiplication in frequency domain.

Why implement convolution in frequency domain?

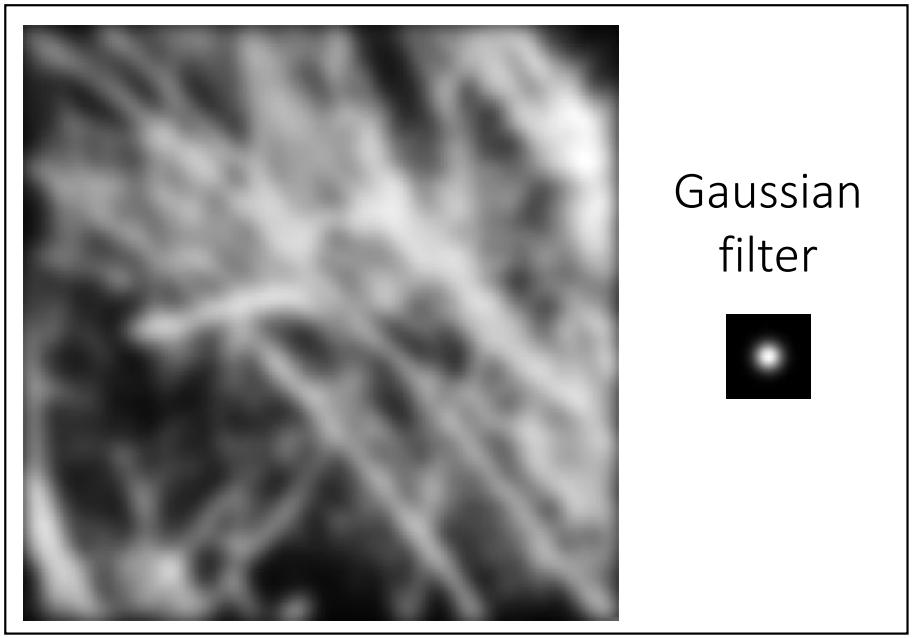
Spatial domain filtering

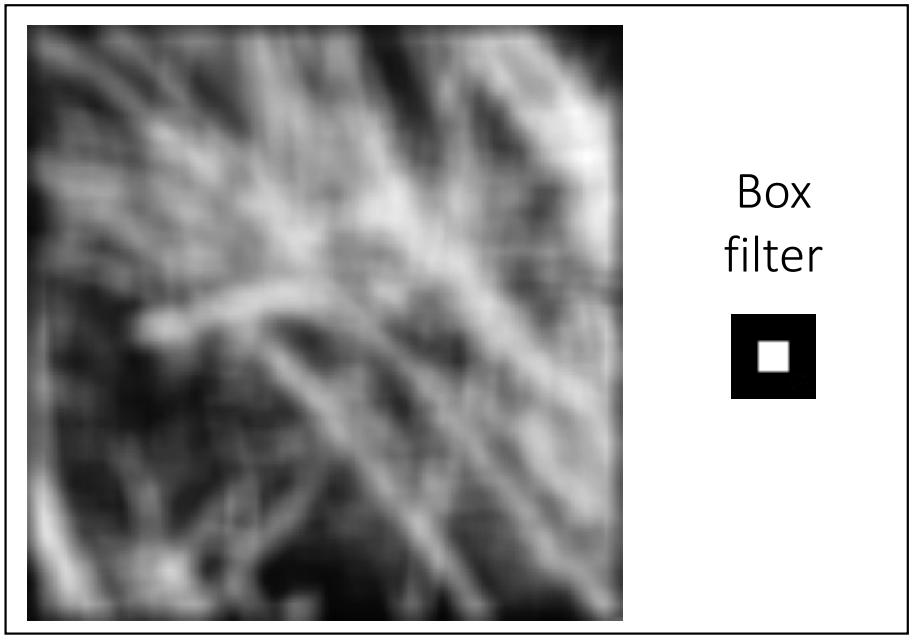


Frequency domain filtering

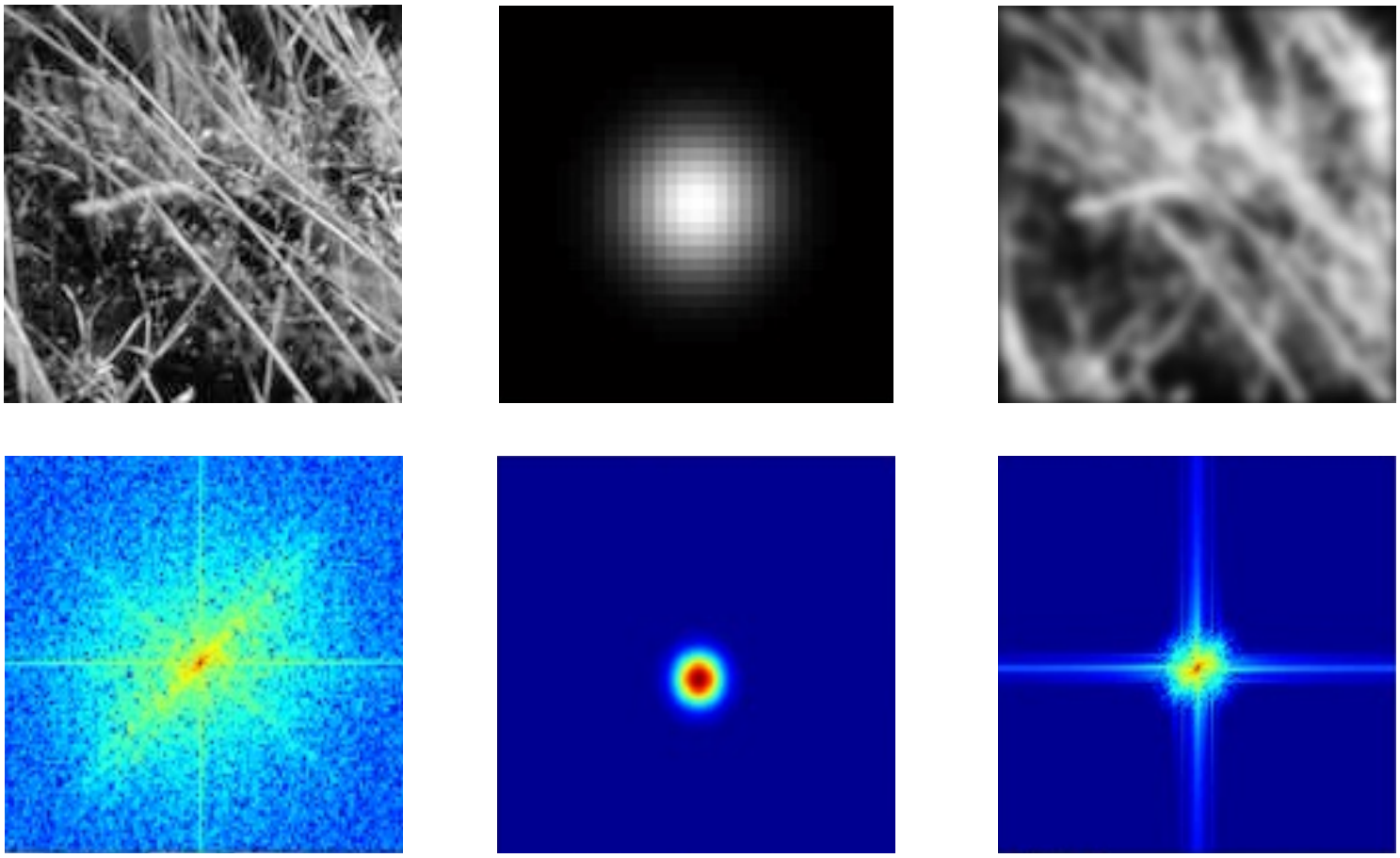
Revisiting blurring

Why does the Gaussian give a nice smooth image, but the square filter give edgy artifacts?

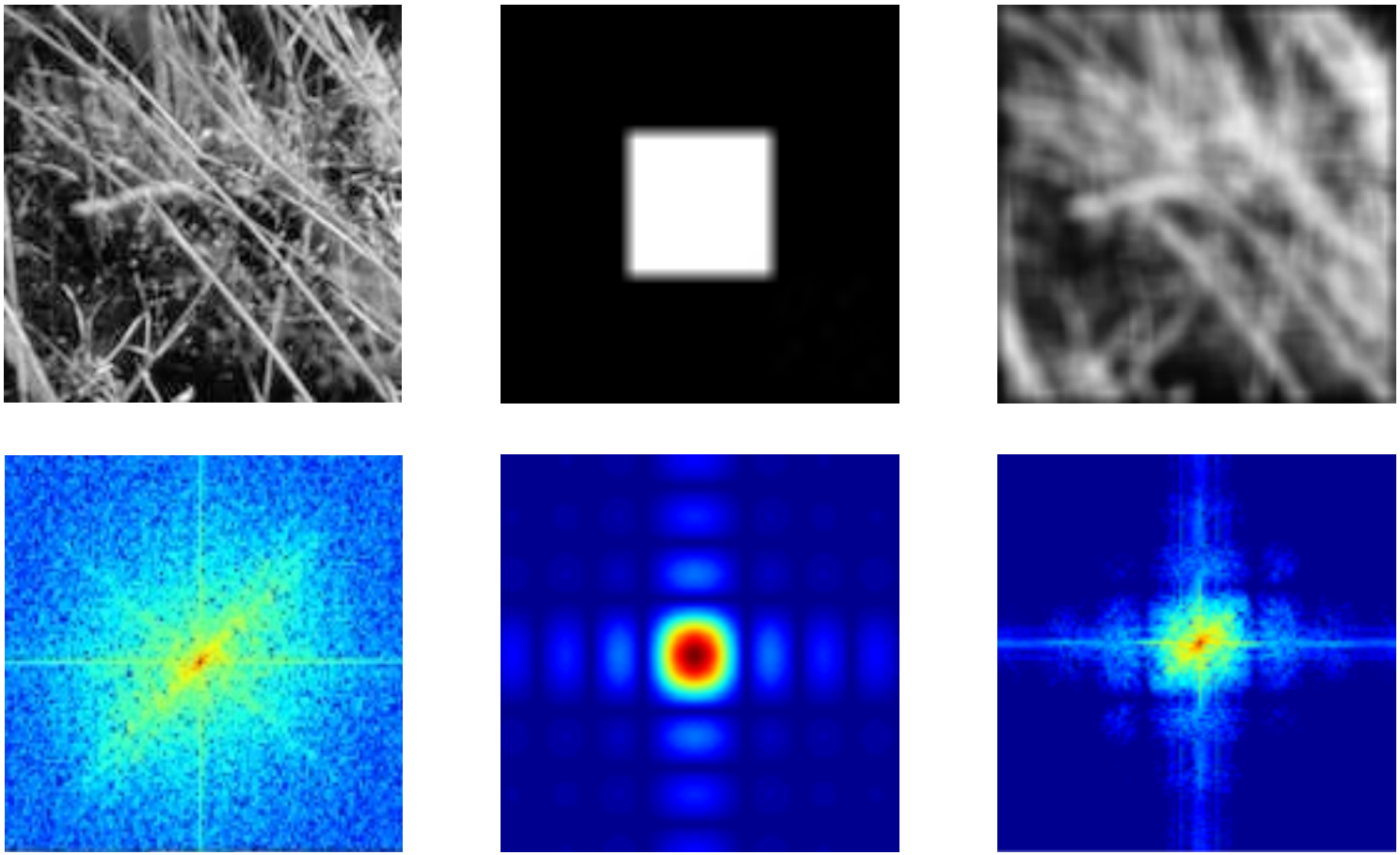




Gaussian blur

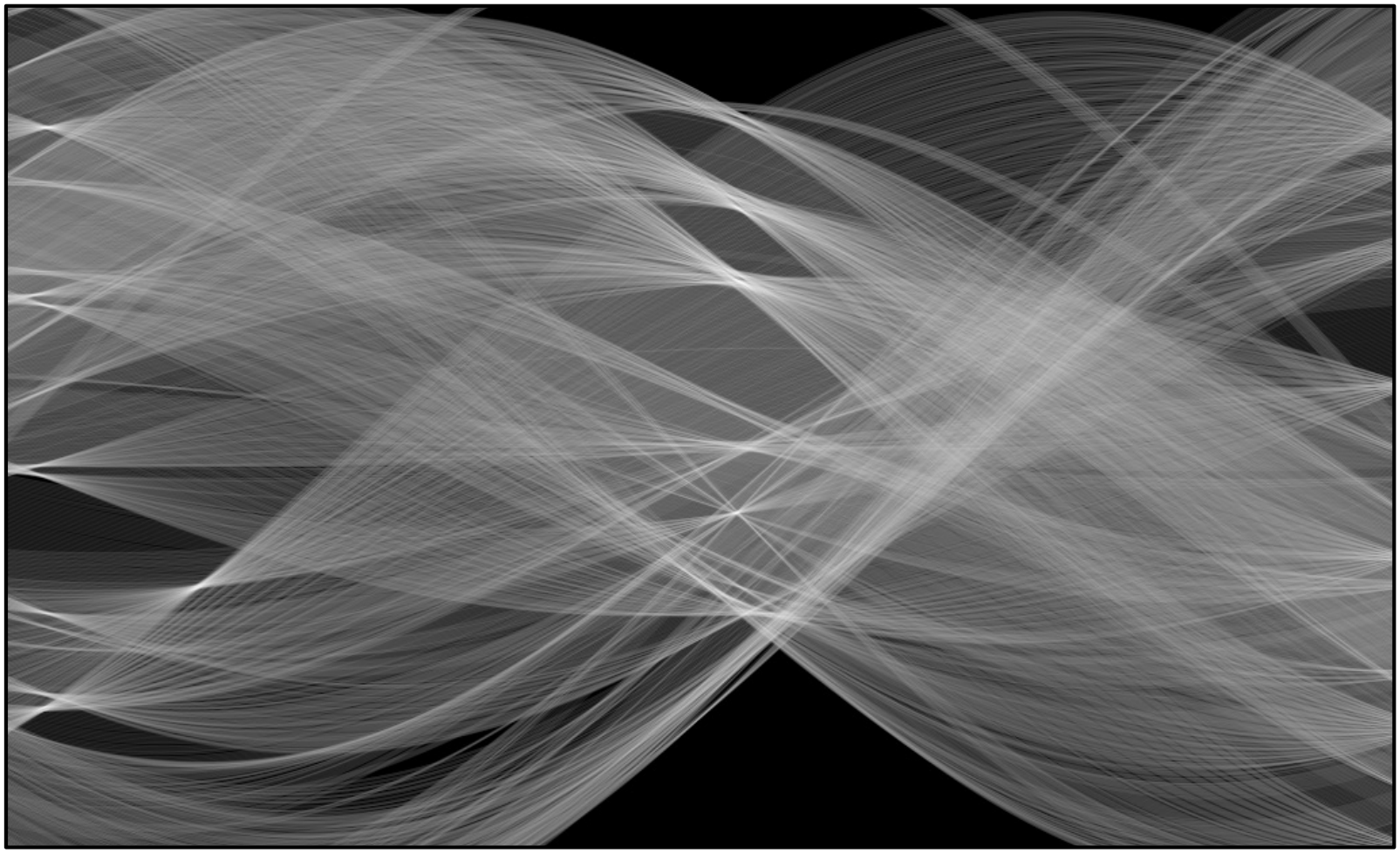


Box blur



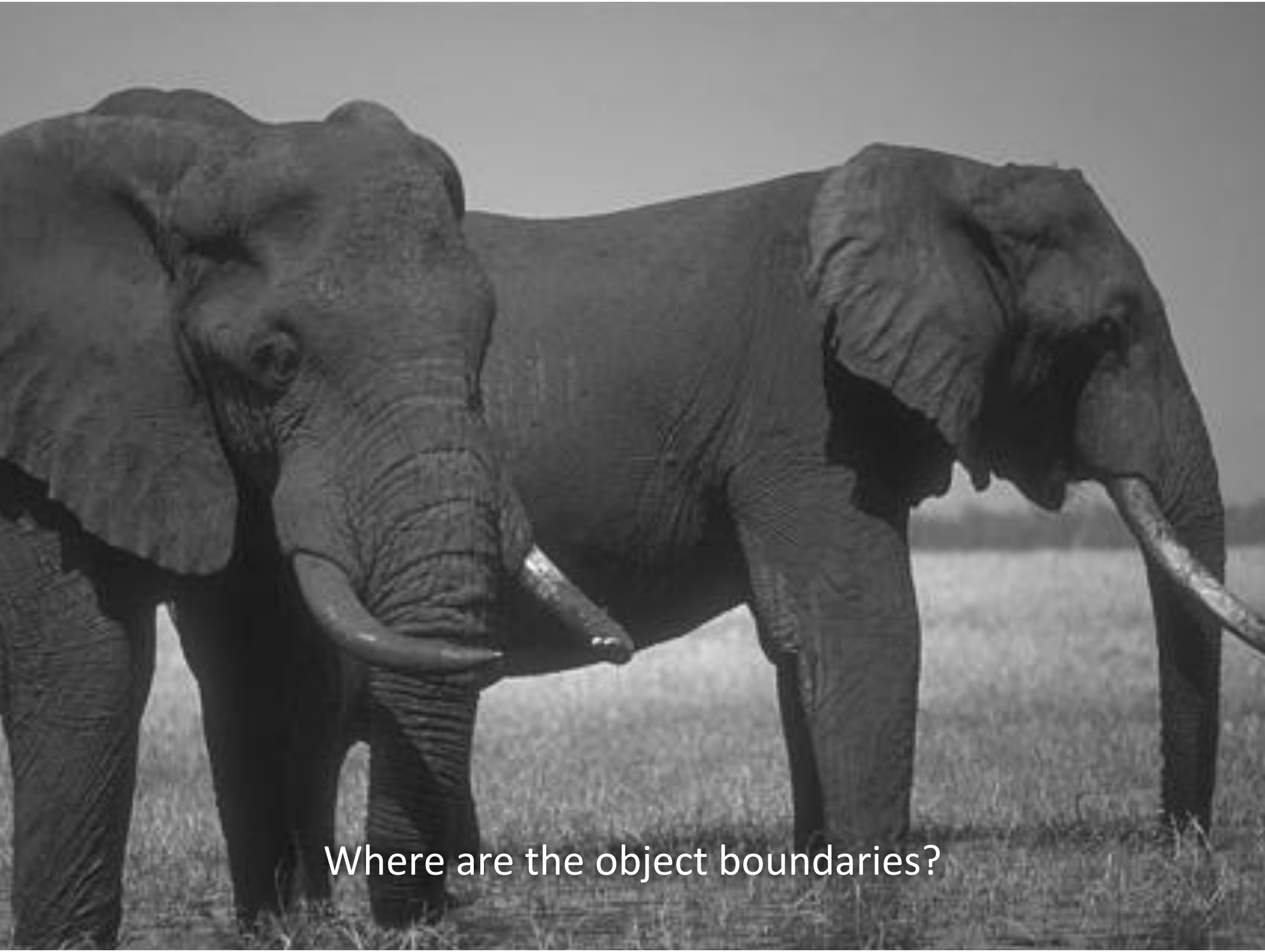
Demo: Python Program

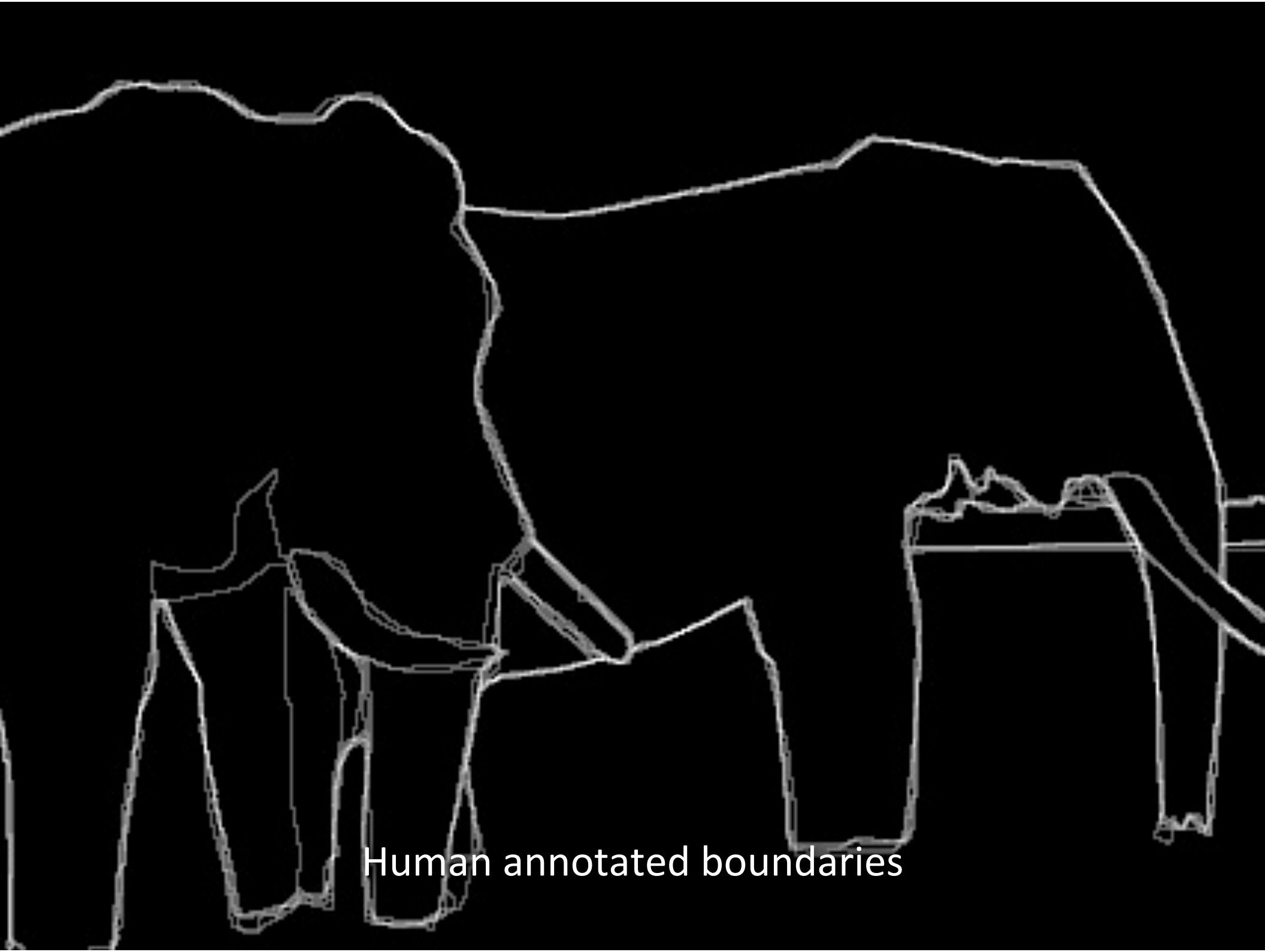
Hough transform

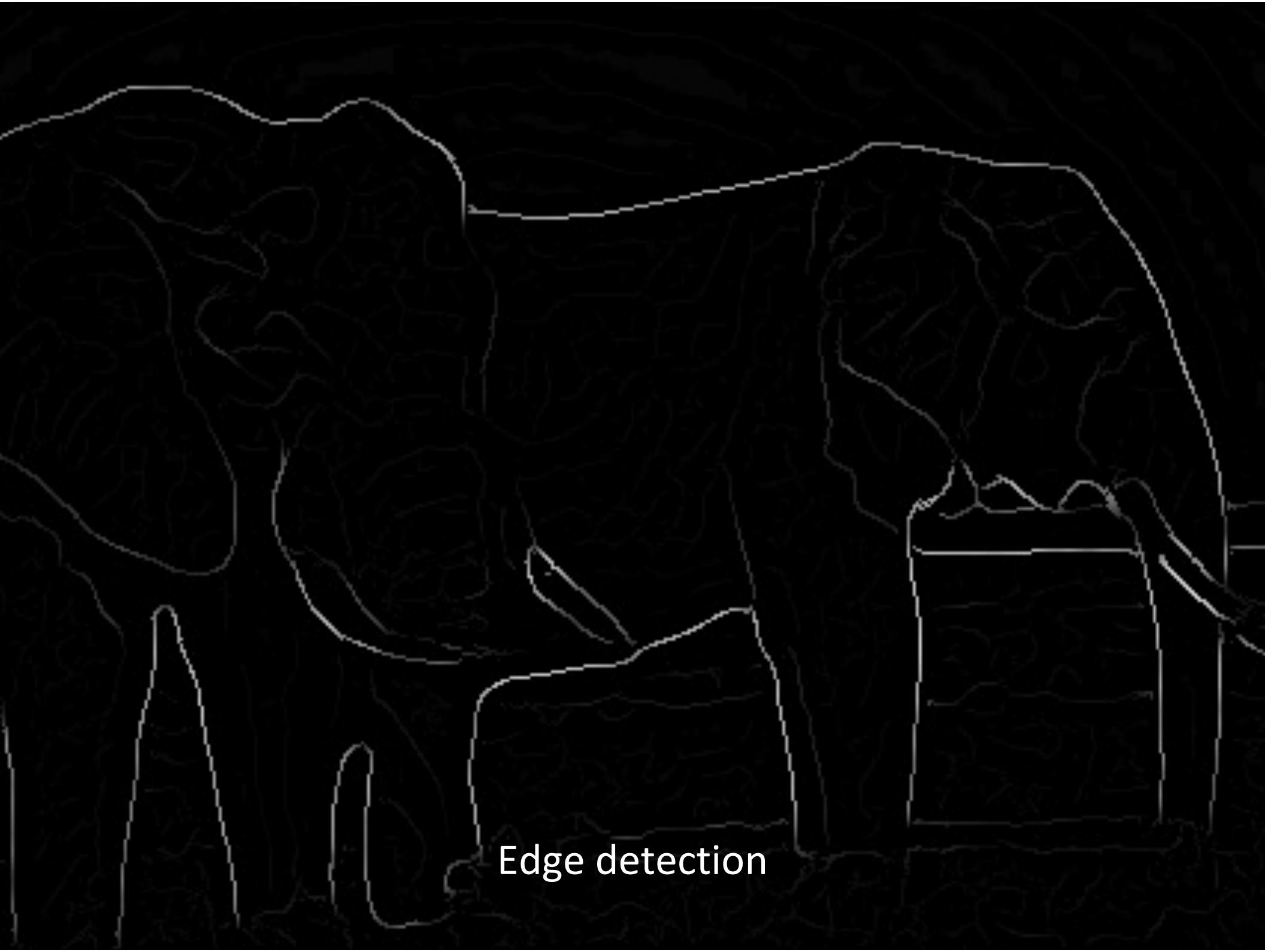


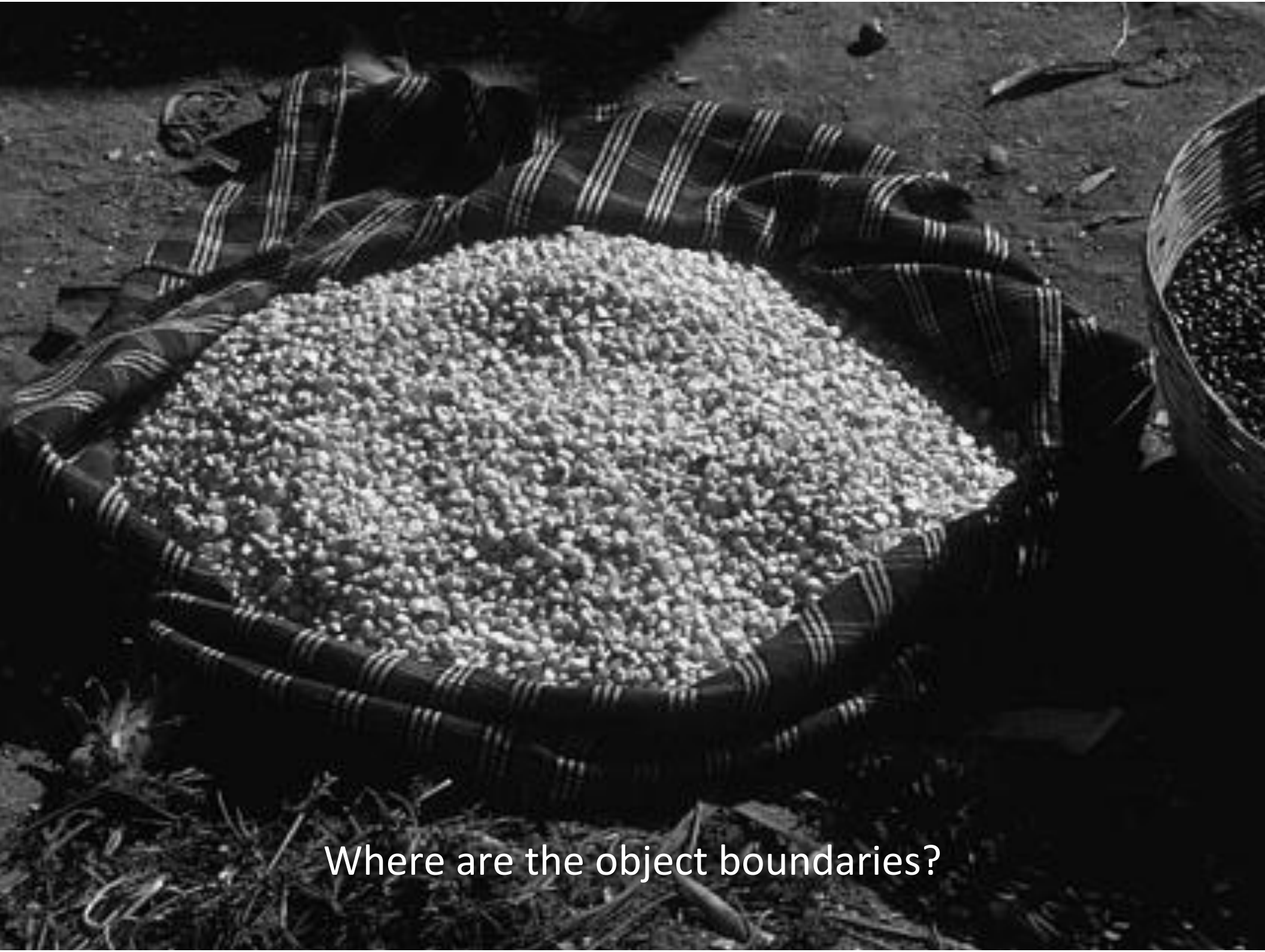
Computer Vision
Fall 2022, Lecture 4

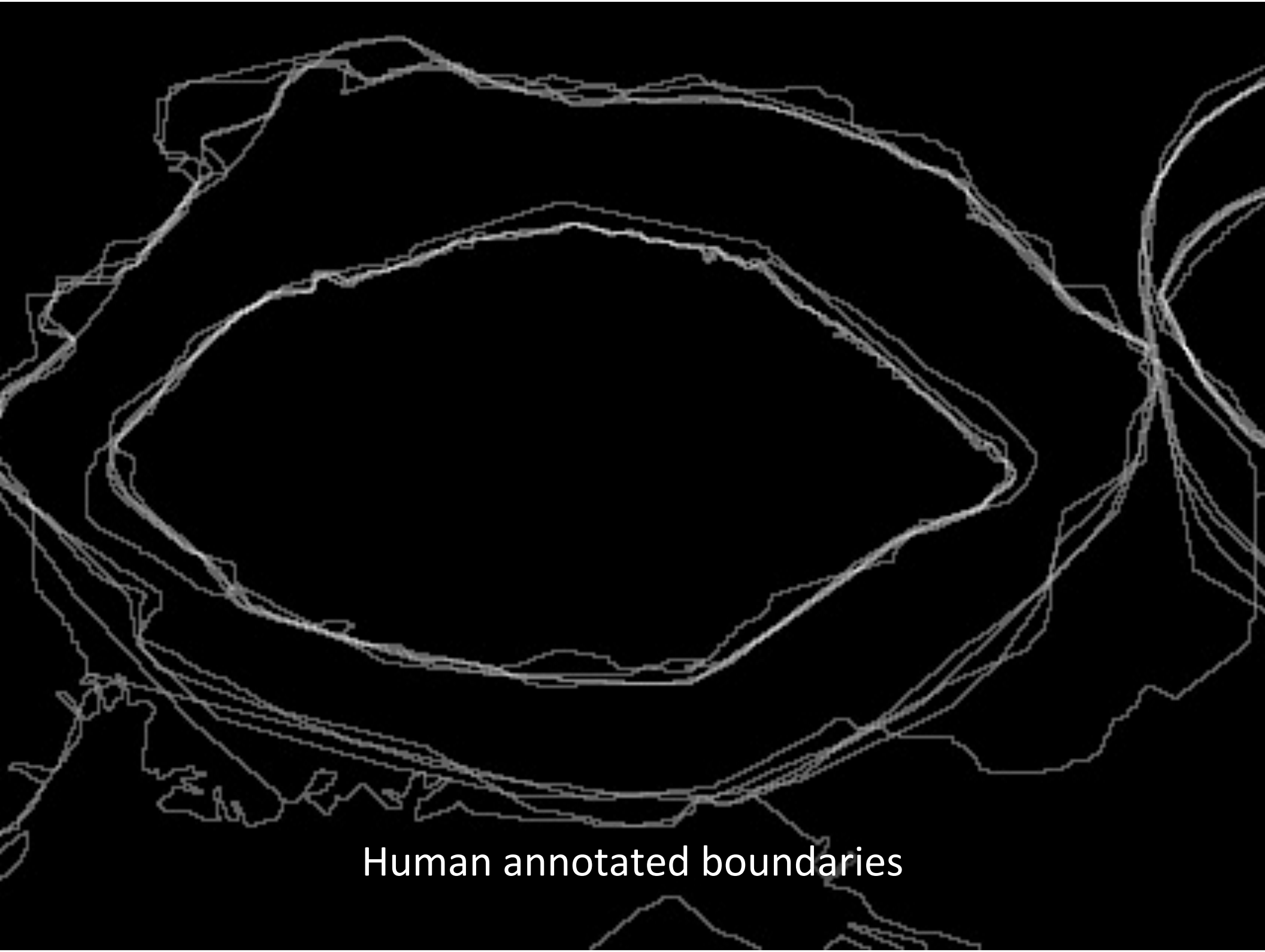
Finding boundaries

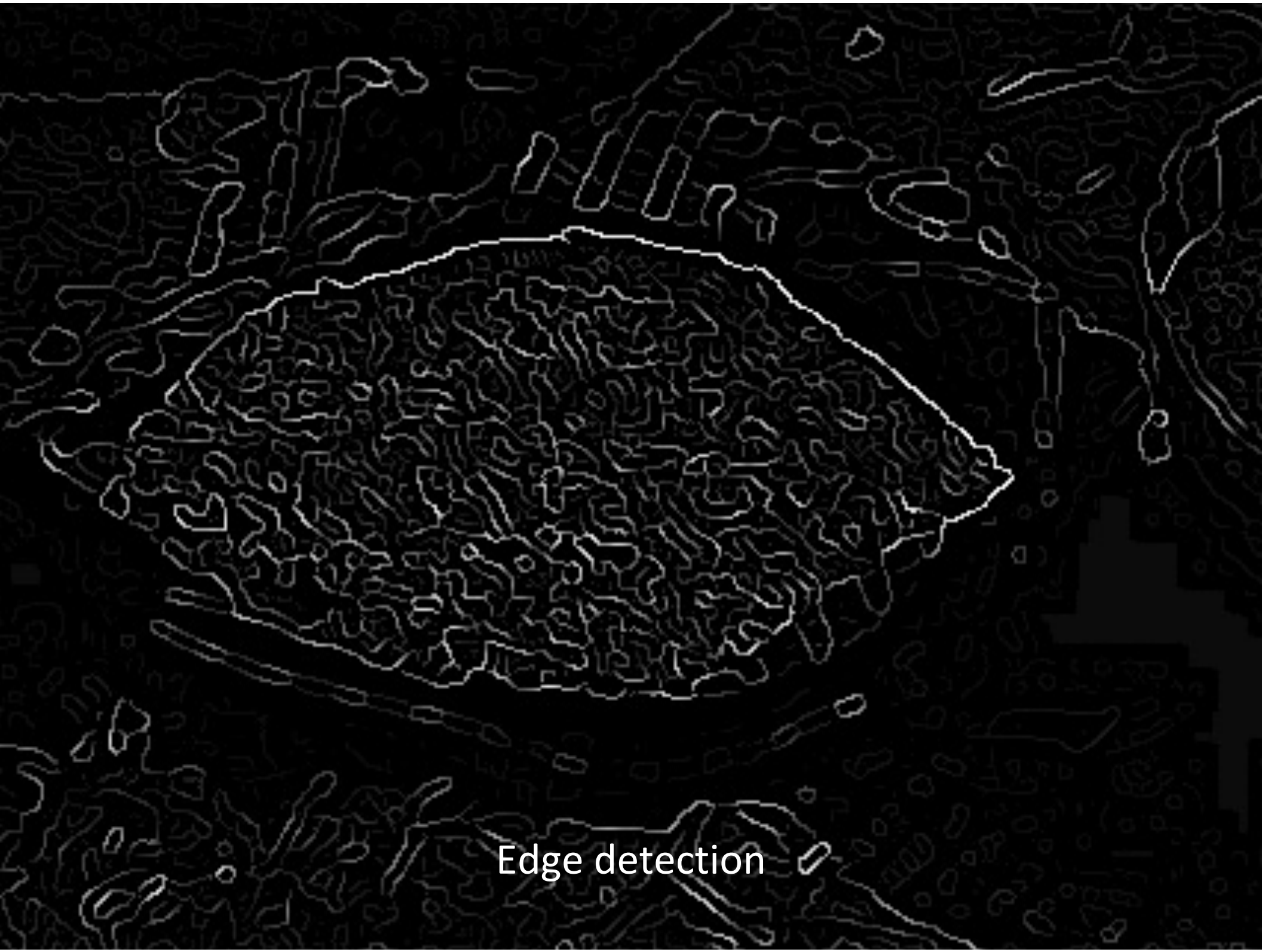

Where are the object boundaries?


Human annotated boundaries

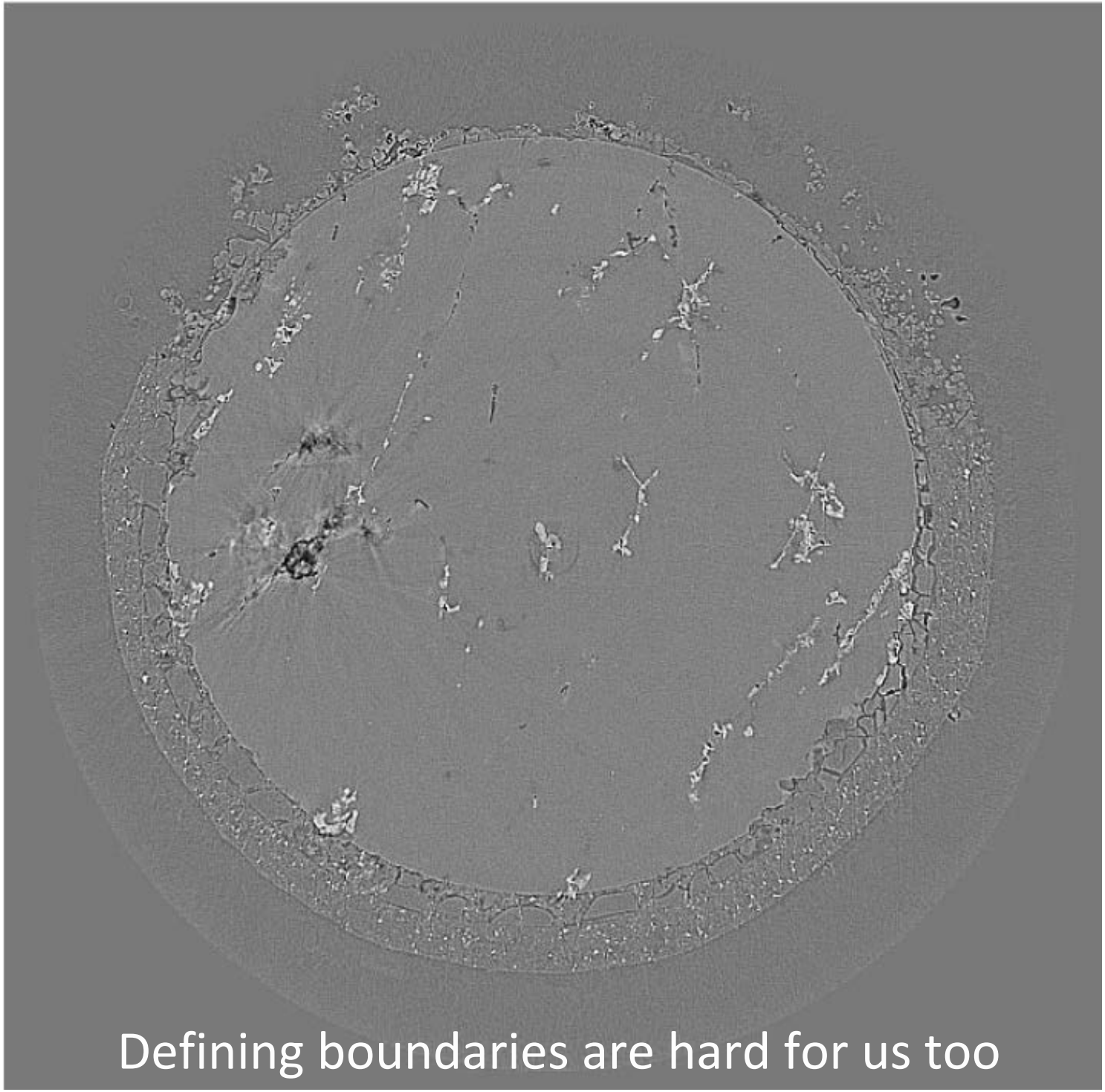

Edge detection

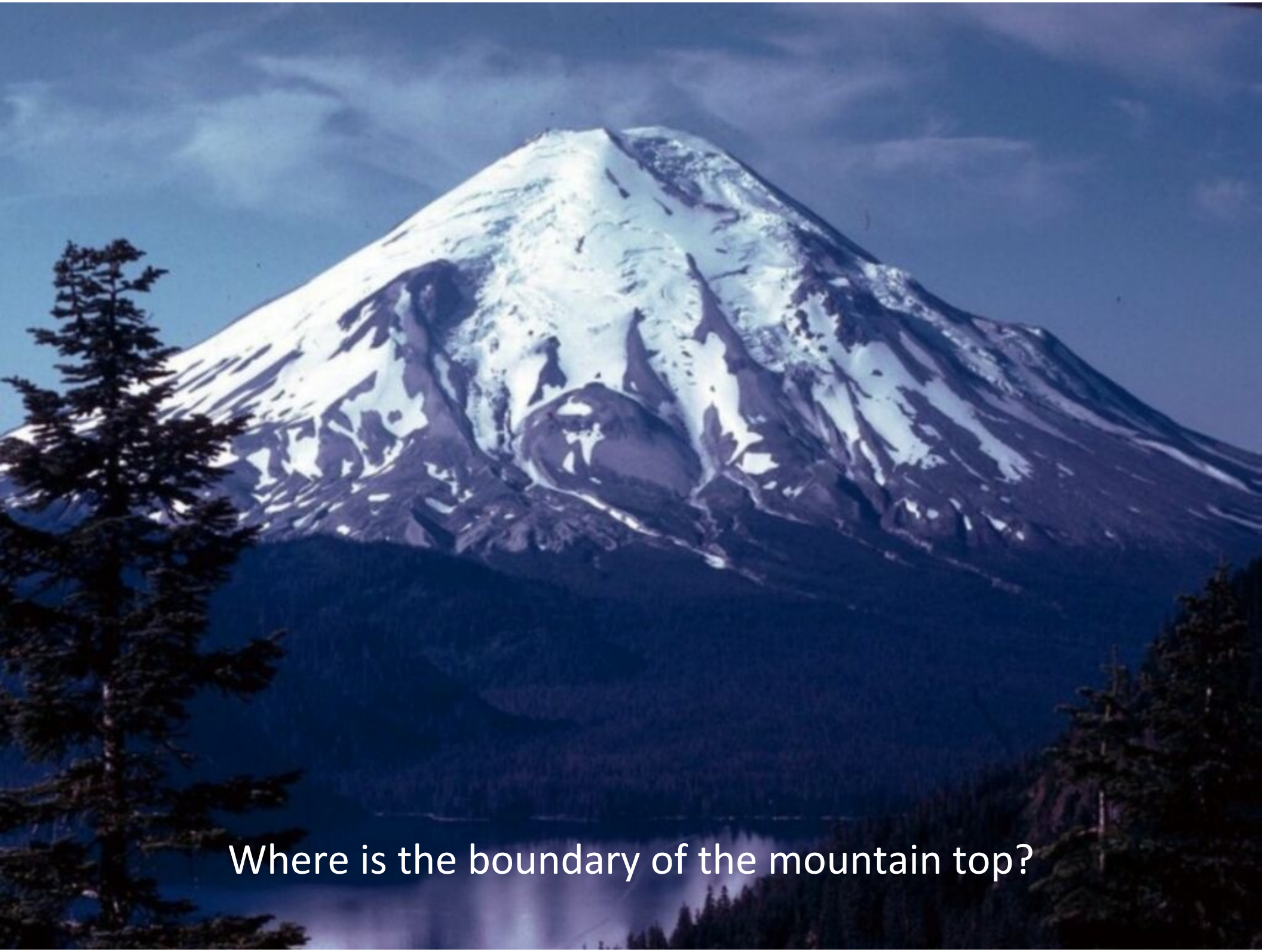

Where are the object boundaries?


Human annotated boundaries


Edge detection

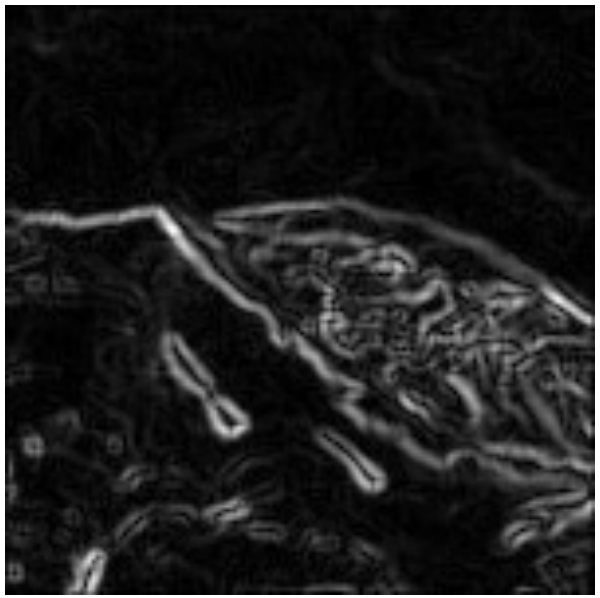

Defining boundaries are hard for us too


Defining boundaries are hard for us too

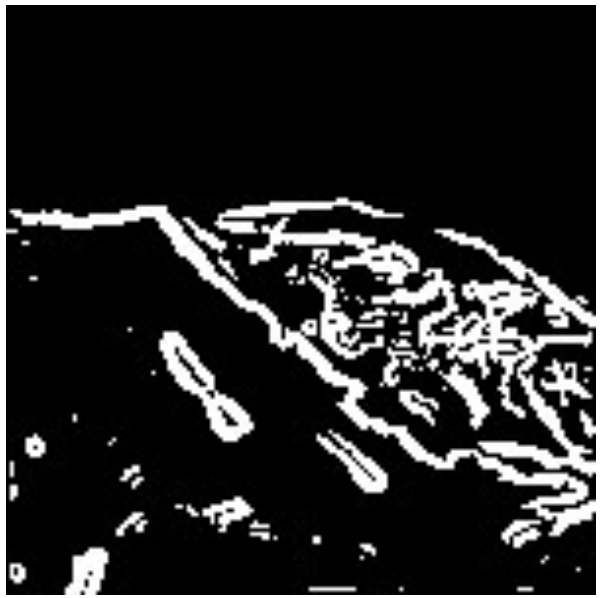

Where is the boundary of the mountain top?

Lines are hard to find

Original image



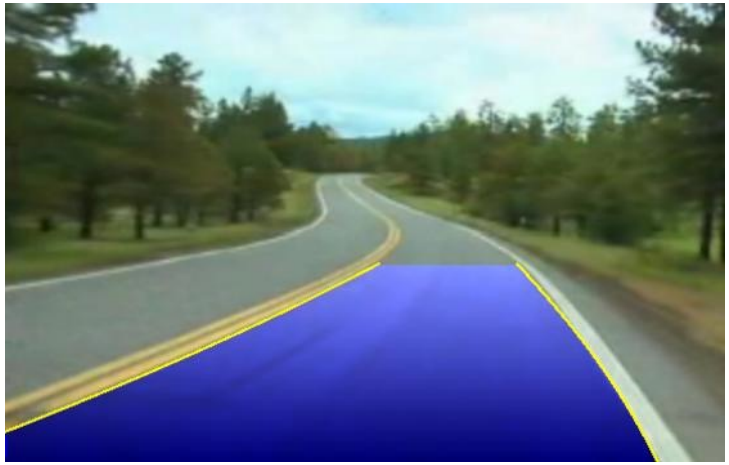
Edge detection



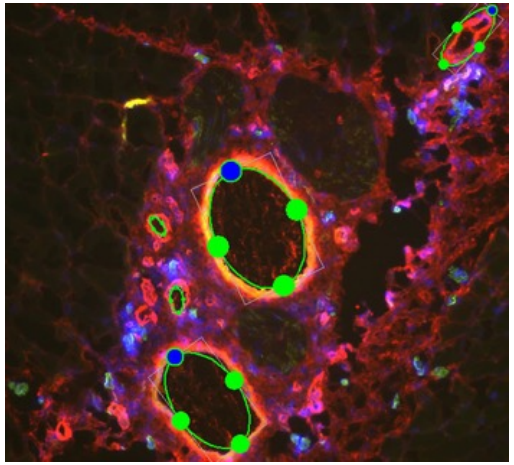
Thresholding

Noisy edge image
Incomplete boundaries

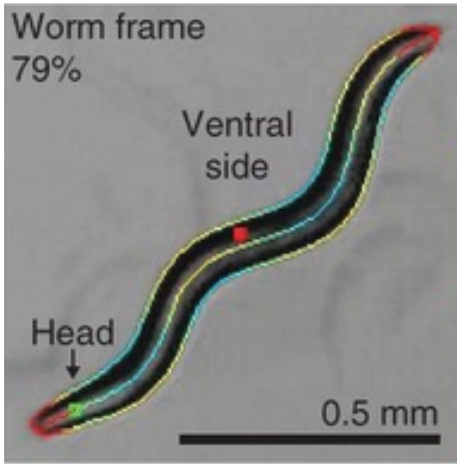
Applications



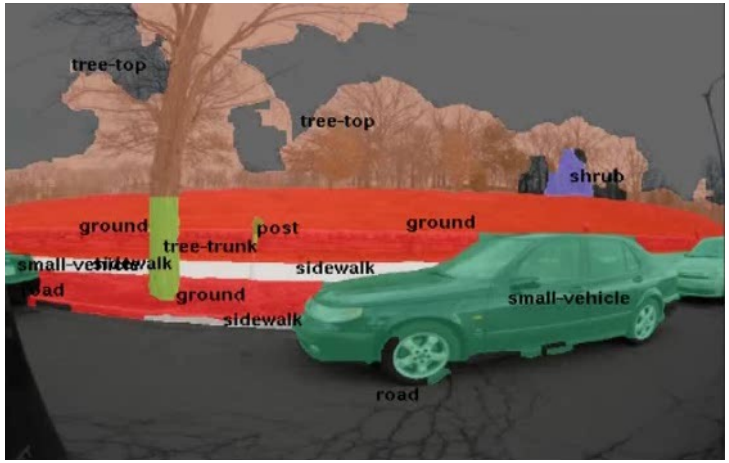
Autonomous Vehicles
(lane line detection)



tissue engineering
(blood vessel counting)



behavioral genetics
(earthworm contours)



Autonomous Vehicles
(semantic scene segmentation)

Computational Photography
(image inpainting)

Line fitting

Slope intercept form

$$y = mx + b$$

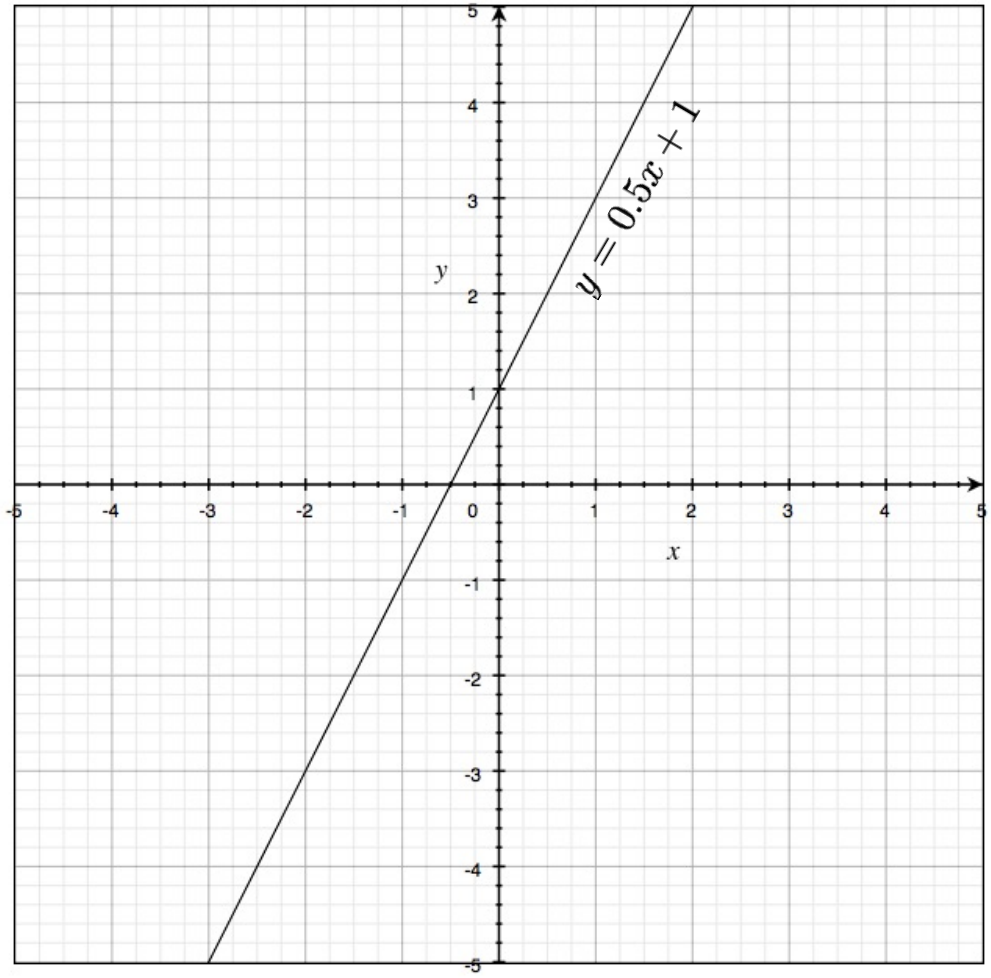
slope y-intercept

What are m and b?

Slope intercept form

$$y = mx + b$$

slope y-intercept



Double intercept form

$$\frac{x}{a} + \frac{y}{b} = 1$$

x-intercept

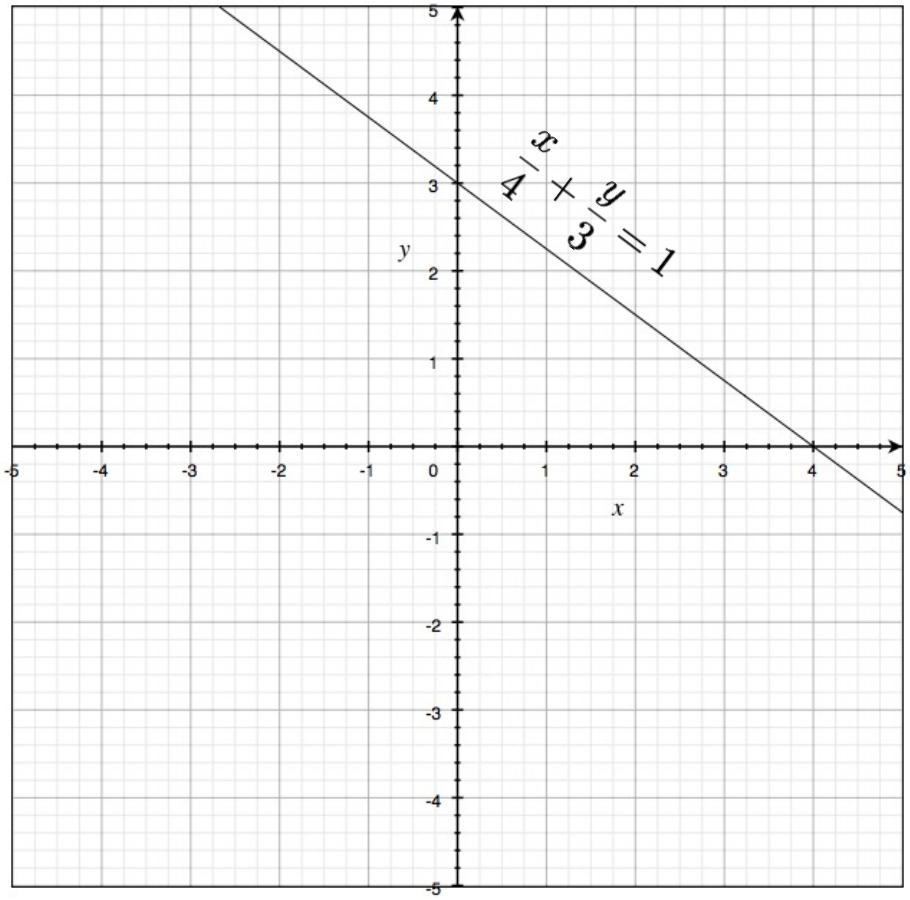
y-intercept

What are a and b?

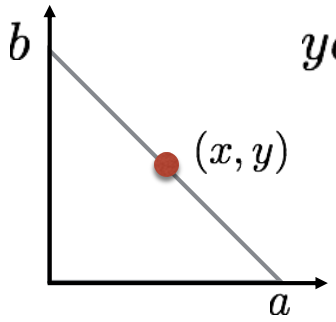
Double intercept form

$$\frac{x}{a} + \frac{y}{b} = 1$$

x-intercept y-intercept



Derivation:



(Similar slope) $$\frac{y-b}{x-0} = \frac{0-y}{a-x}$$

$$ya + yx - ba + bx = -yx$$

$$ya + bx = ba$$

$$\frac{y}{b} + \frac{x}{a} = 1$$

Normal Form

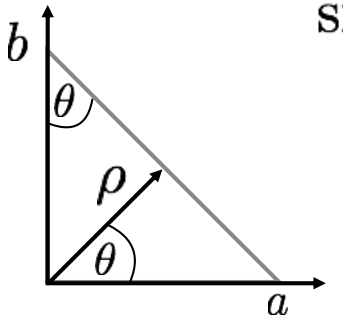
$$x \cos \theta + y \sin \theta = \rho$$

What are rho and theta?

Normal Form

$$x\cos\theta + y\sin\theta = \rho$$

Derivation:

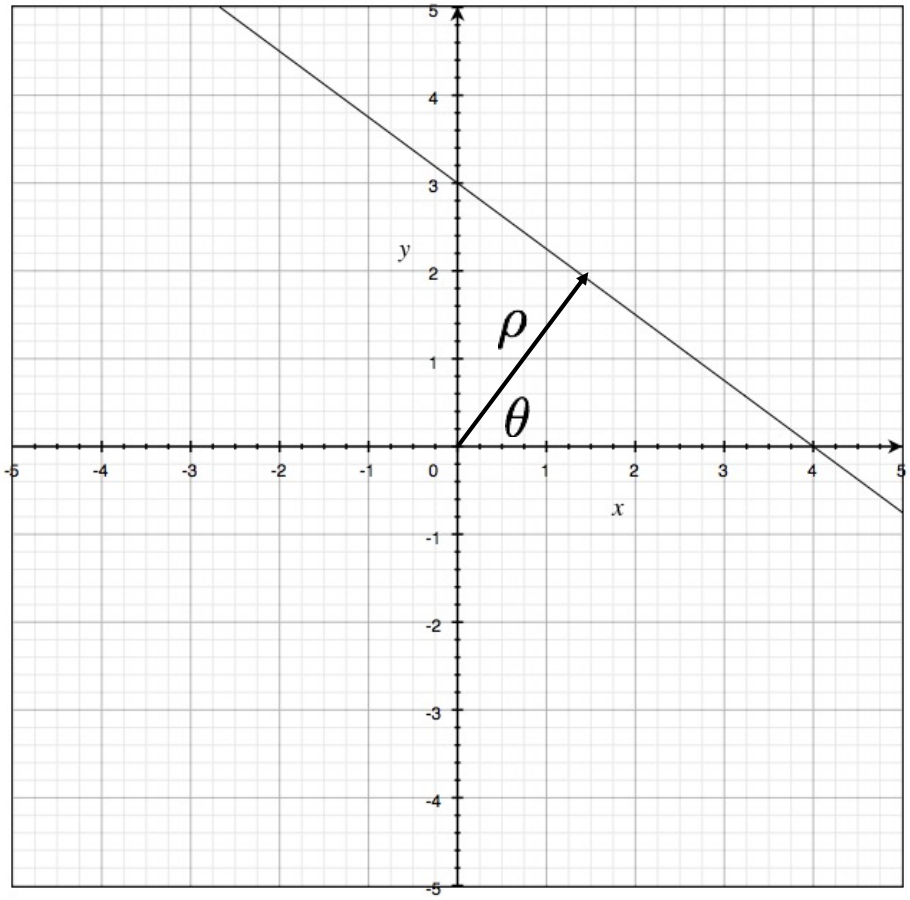


$$\cos\theta = \frac{\rho}{a} \rightarrow a = \frac{\rho}{\cos\theta}$$

$$\sin\theta = \frac{\rho}{b} \rightarrow b = \frac{\rho}{\sin\theta}$$

plug into: $\frac{x}{a} + \frac{y}{b} = 1$

$$x\cos\theta + y\sin\theta = \rho$$



Hough transform

Hough transform

- Generic framework for detecting a parametric model
- Edges don’t have to be connected
- Lines can be occluded
- Key idea: edges **vote** for the possible models

Image and parameter space

variables

$$y = mx + b$$

parameters

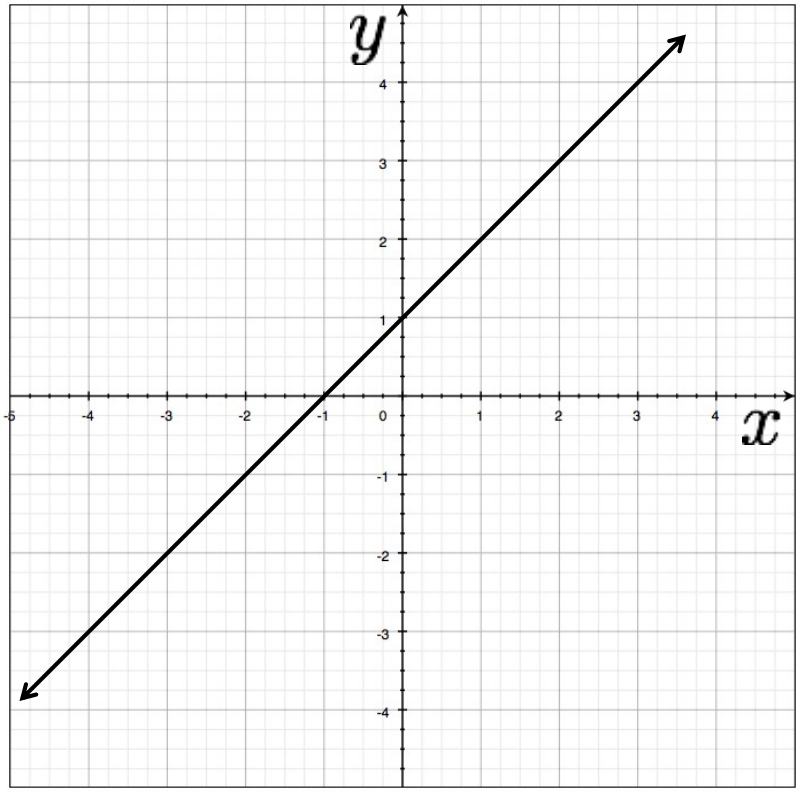


Image space

Image and parameter space

variables

$$y = mx + b$$

parameters

variables

$$y - mx = b$$

parameters

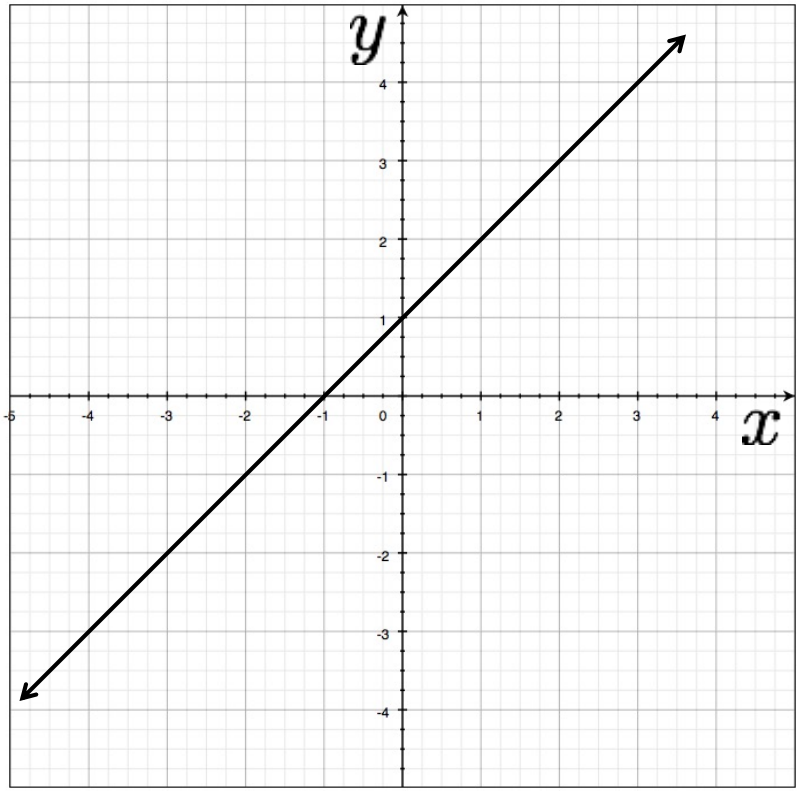
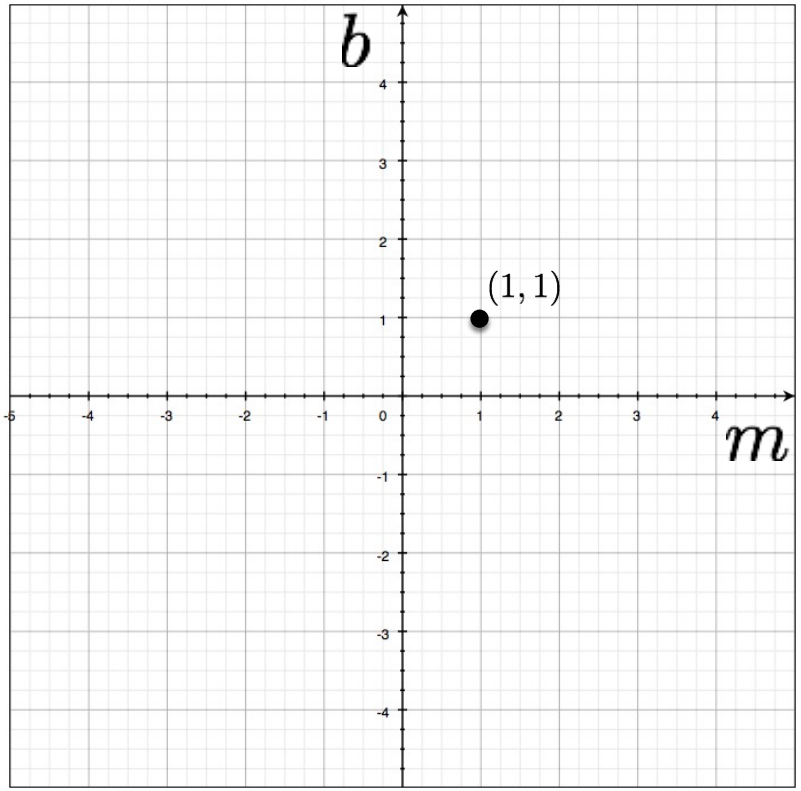


Image space

a line becomes a point



Parameter space

Image and parameter space

variables

$$y = mx + b$$

parameters

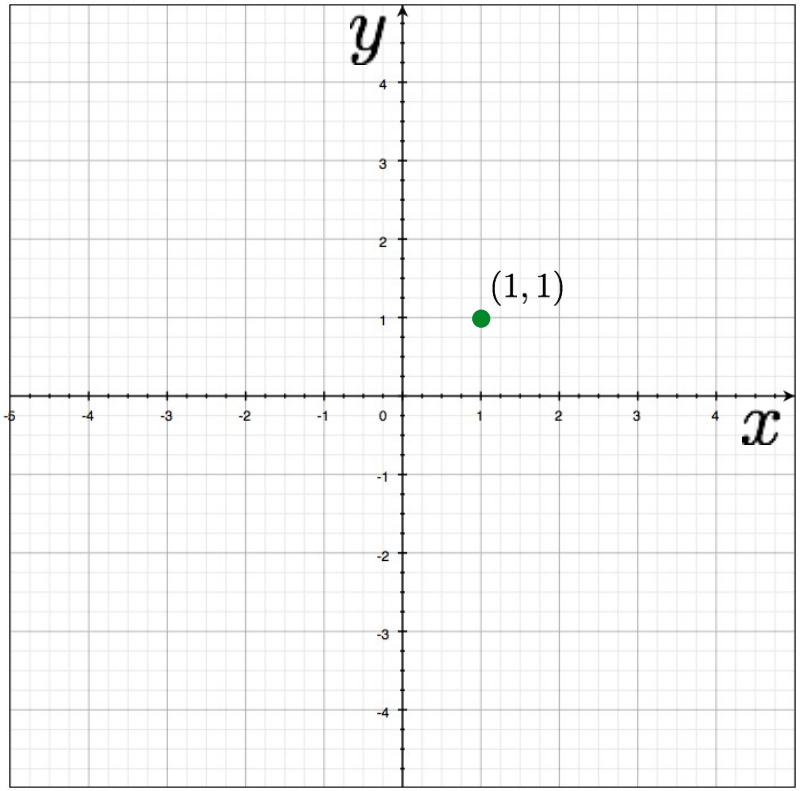


Image space

What would a point in image space become in parameter space?

Image and parameter space

variables

$$y = mx + b$$

parameters

variables

$$y - mx = b$$

parameters

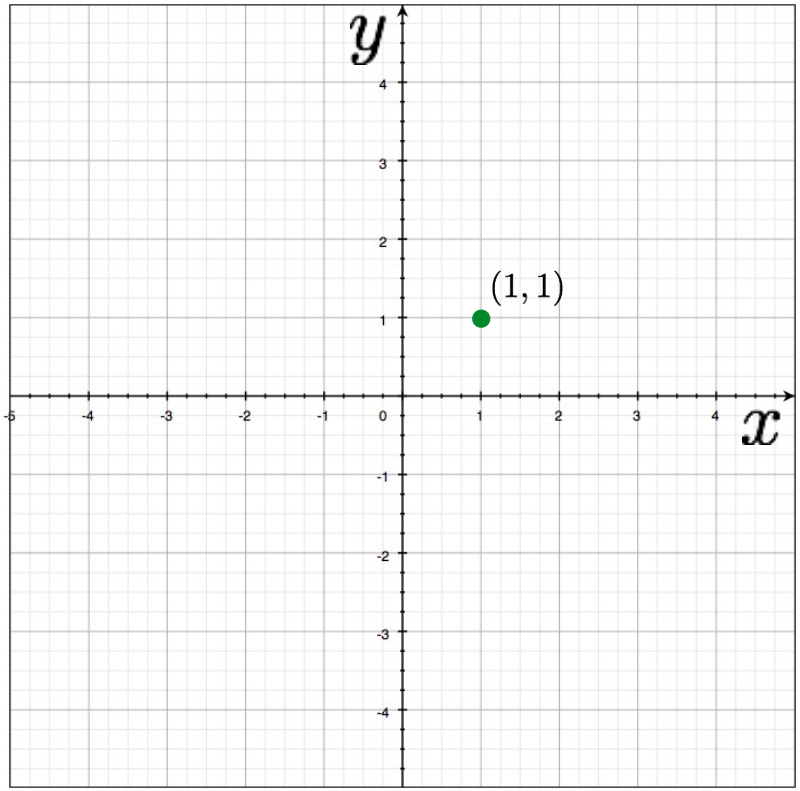
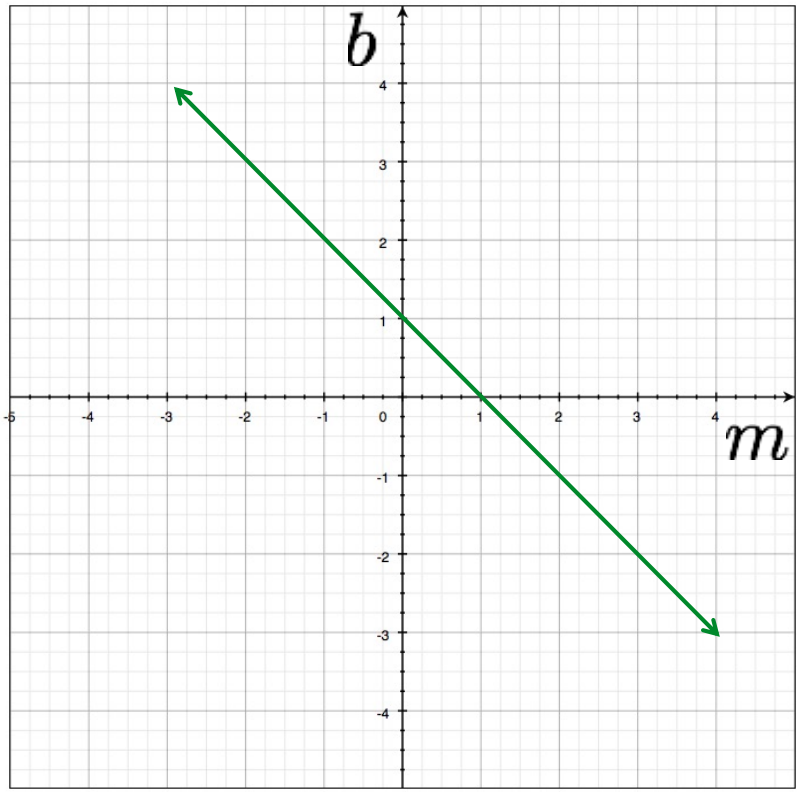


Image space

a point becomes a line



Parameter space

Image and parameter space

variables

$$y = mx + b$$

parameters

variables

$$y - mx = b$$

parameters

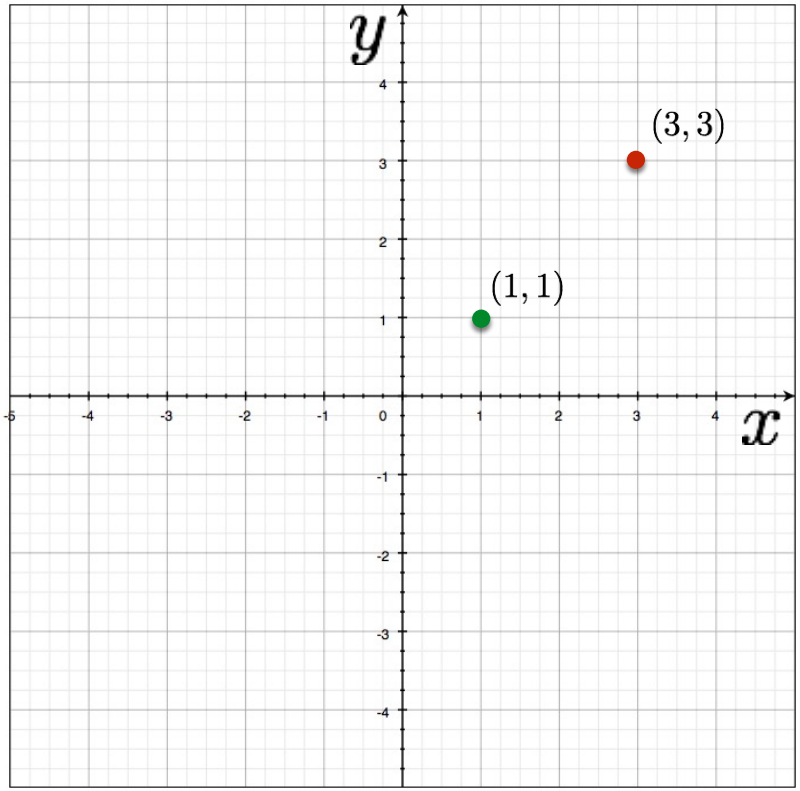
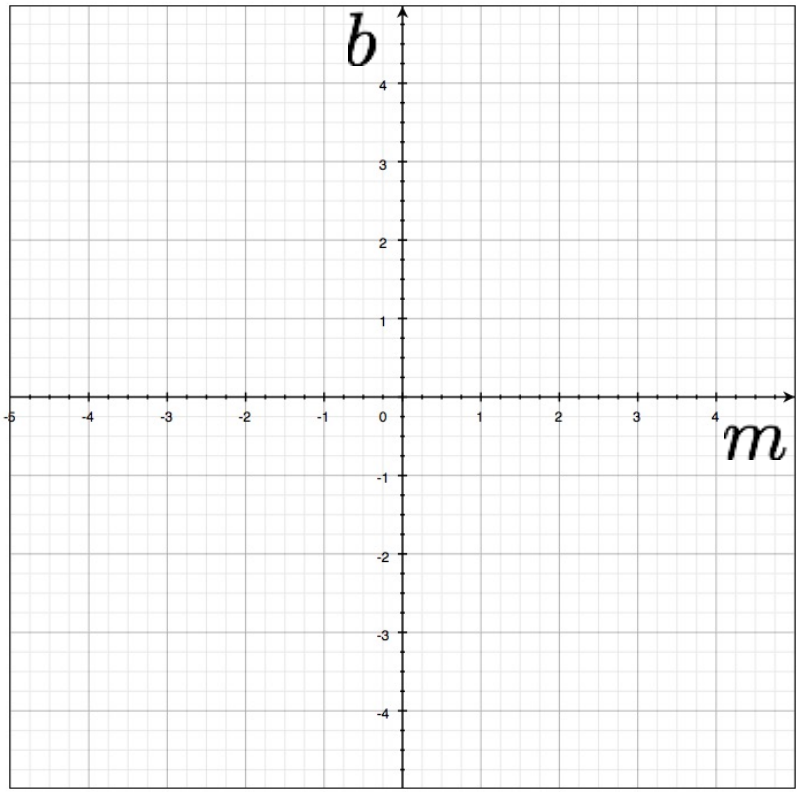


Image space

two points become ?



Parameter space

Image and parameter space

variables

$$y = mx + b$$

parameters

variables

$$y - mx = b$$

parameters

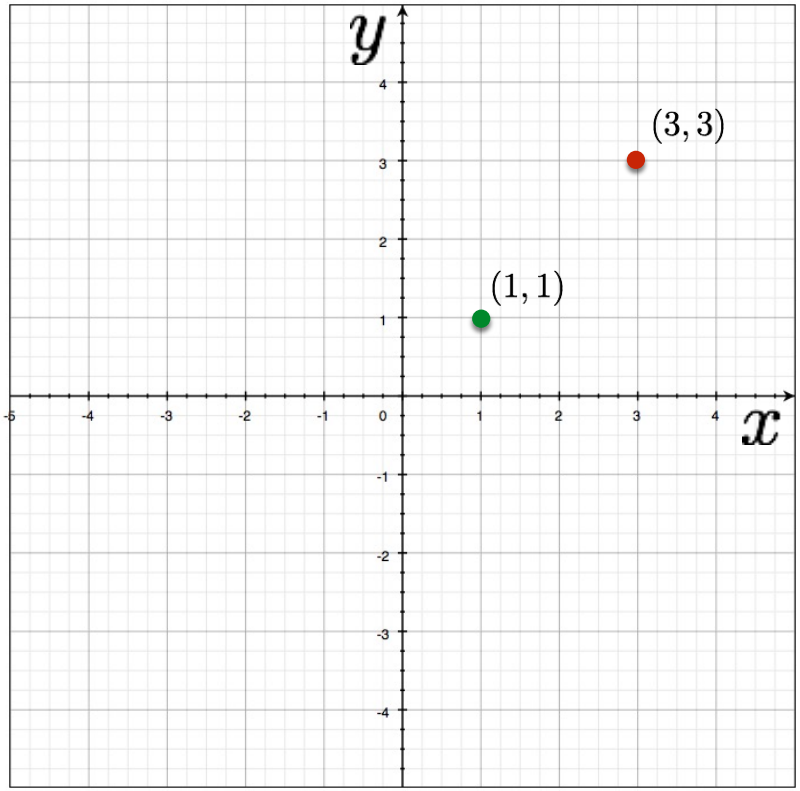
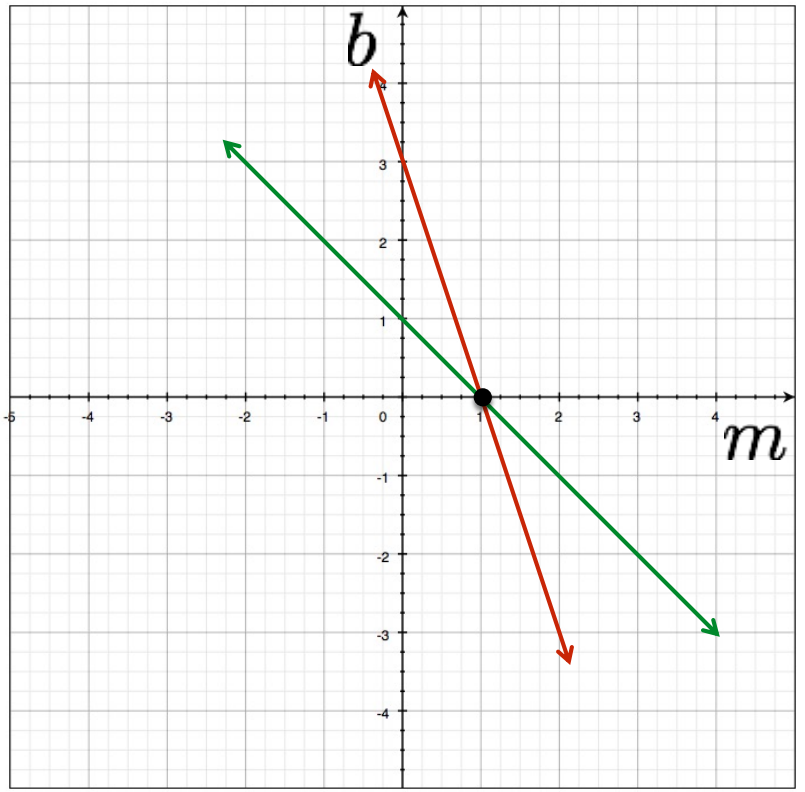


Image space

two points become ?



Parameter space

Image and parameter space

variables

$$y = mx + b$$

parameters

variables

$$y - mx = b$$

parameters

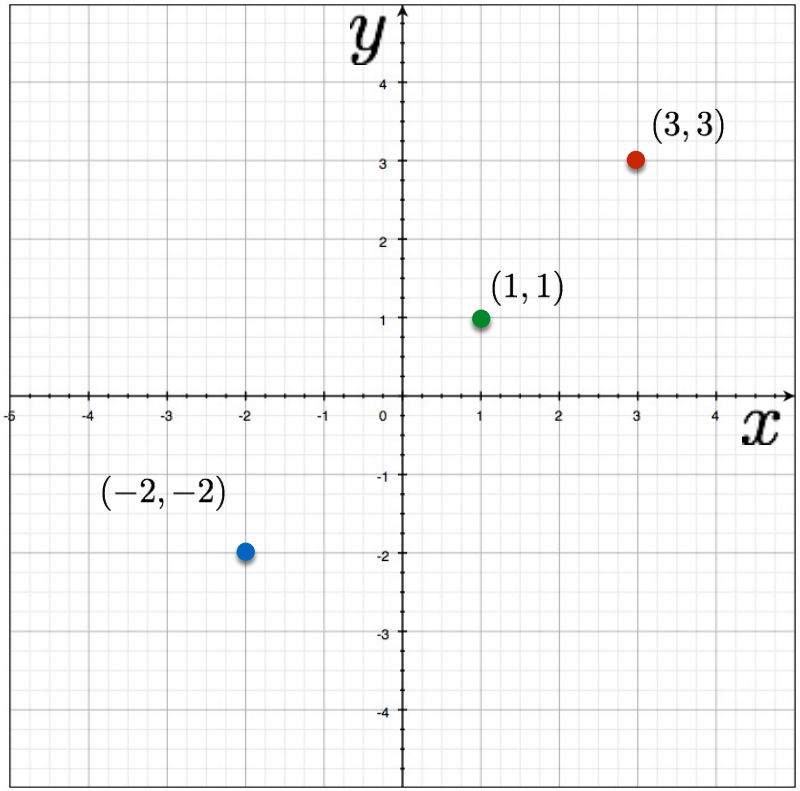
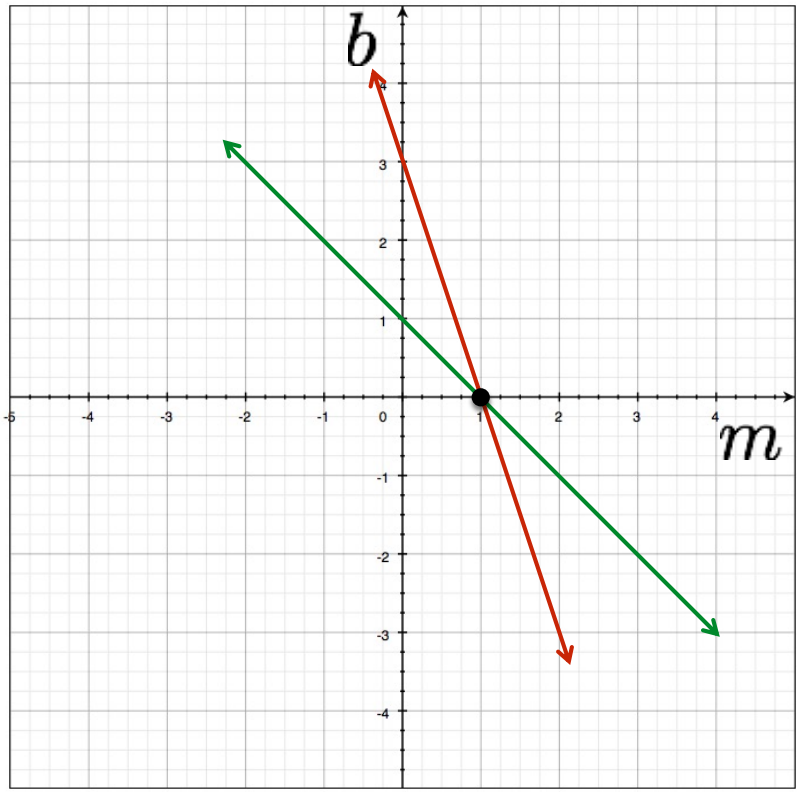


Image space

three points become ?



Parameter space

Image and parameter space

variables

$$y = mx + b$$

parameters

variables

$$y - mx = b$$

parameters

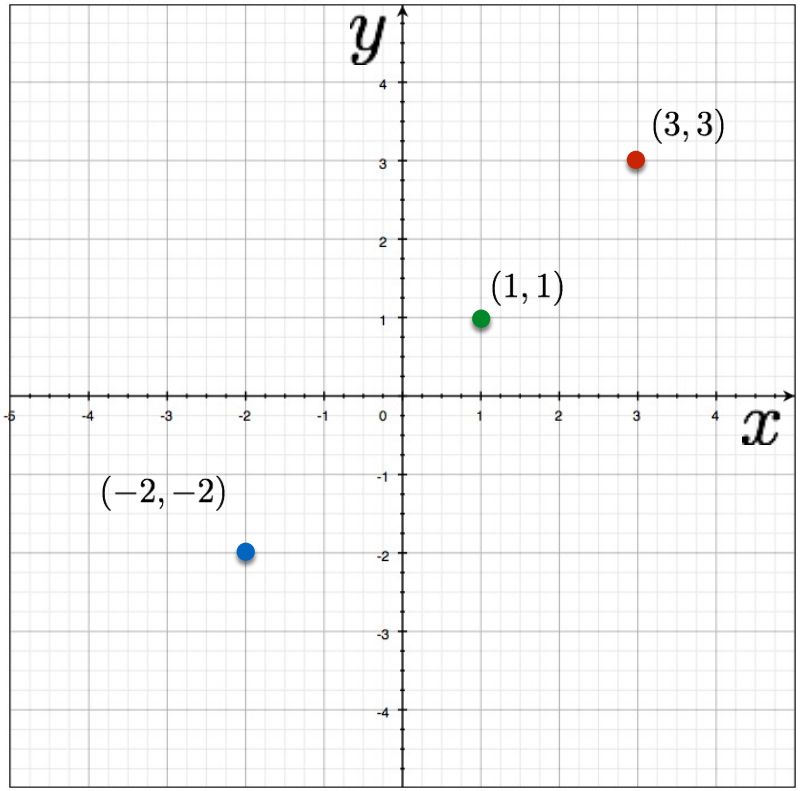
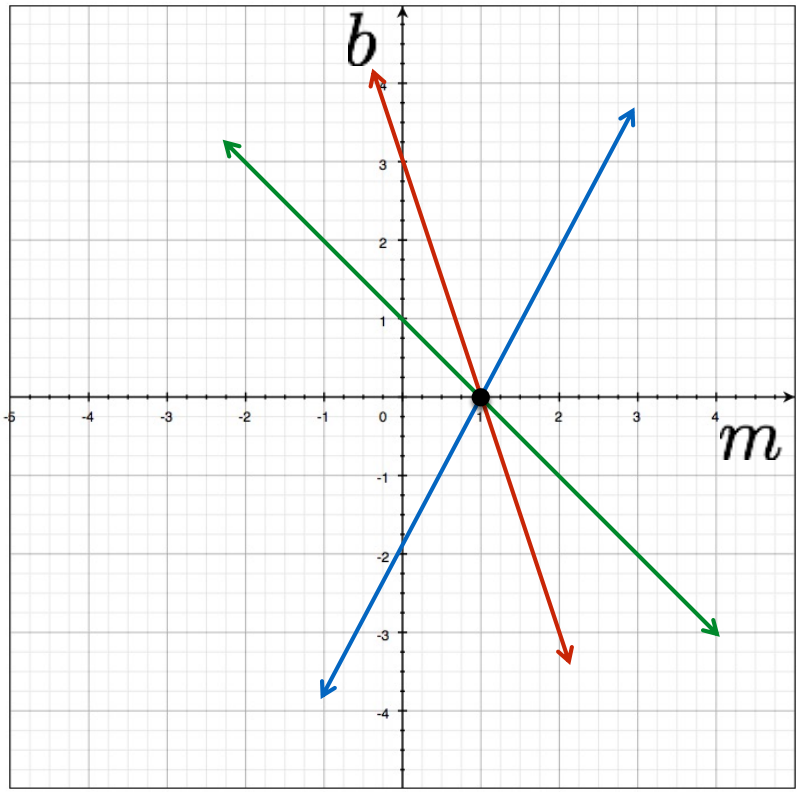


Image space

three points become ?



Parameter space

Image and parameter space

variables

$$y = mx + b$$

parameters

variables

$$y - mx = b$$

parameters

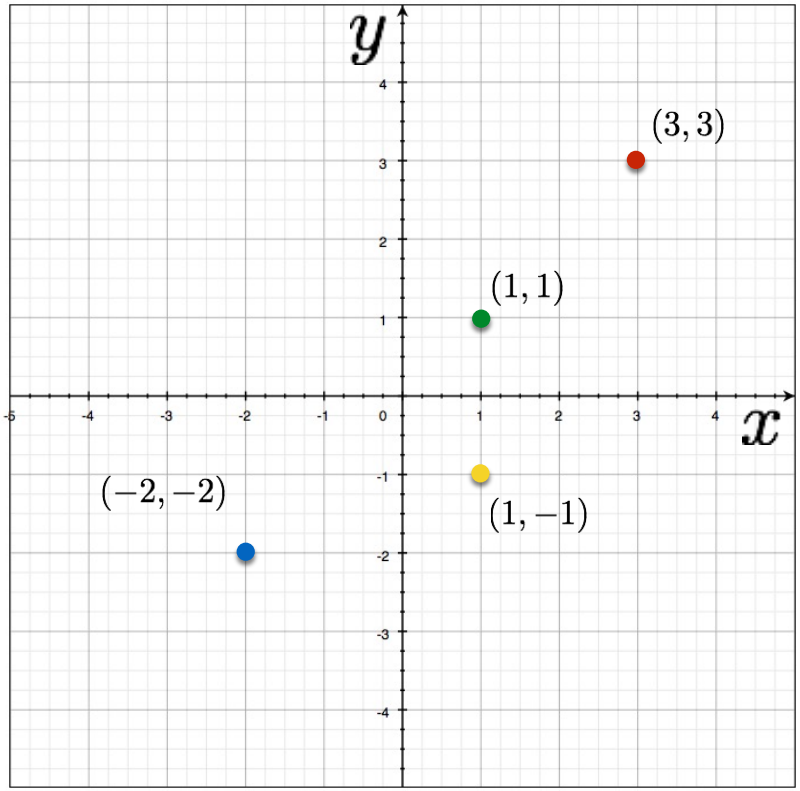
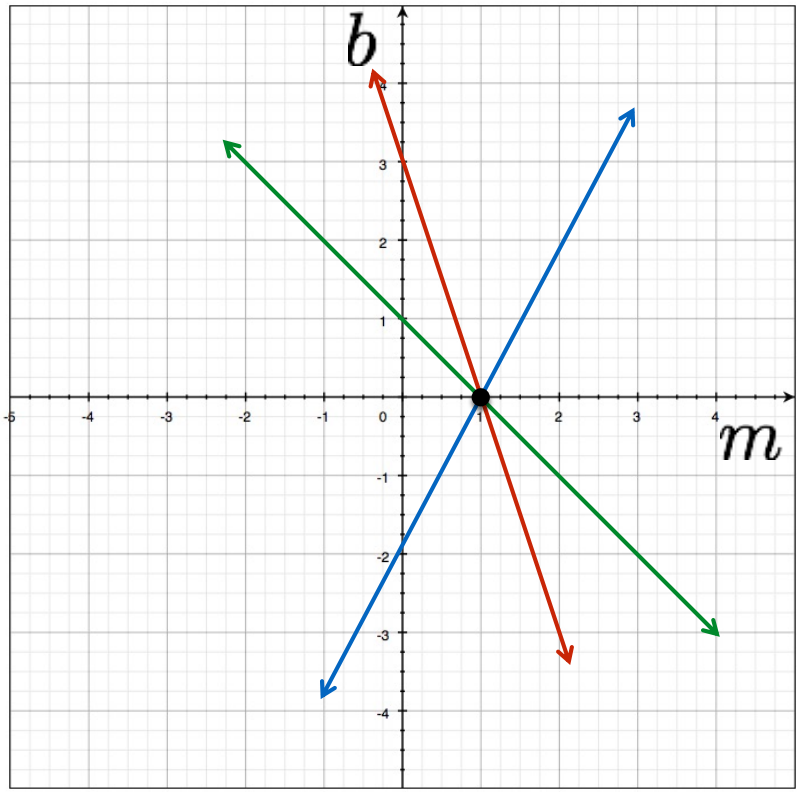


Image space

four points become ?



Parameter space

Image and parameter space

variables

$$y = mx + b$$

parameters

variables

$$y - mx = b$$

parameters

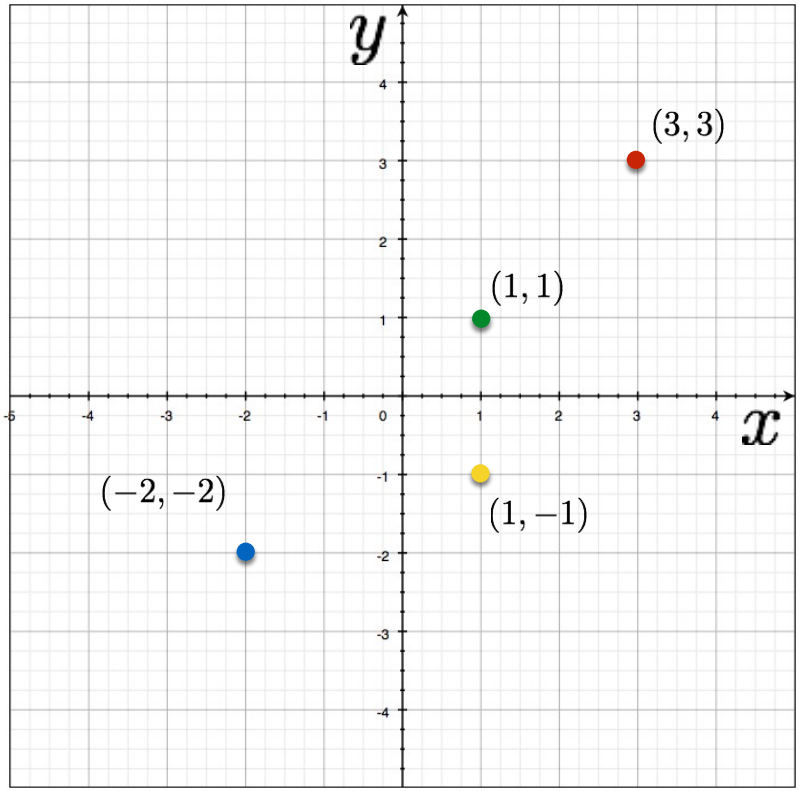
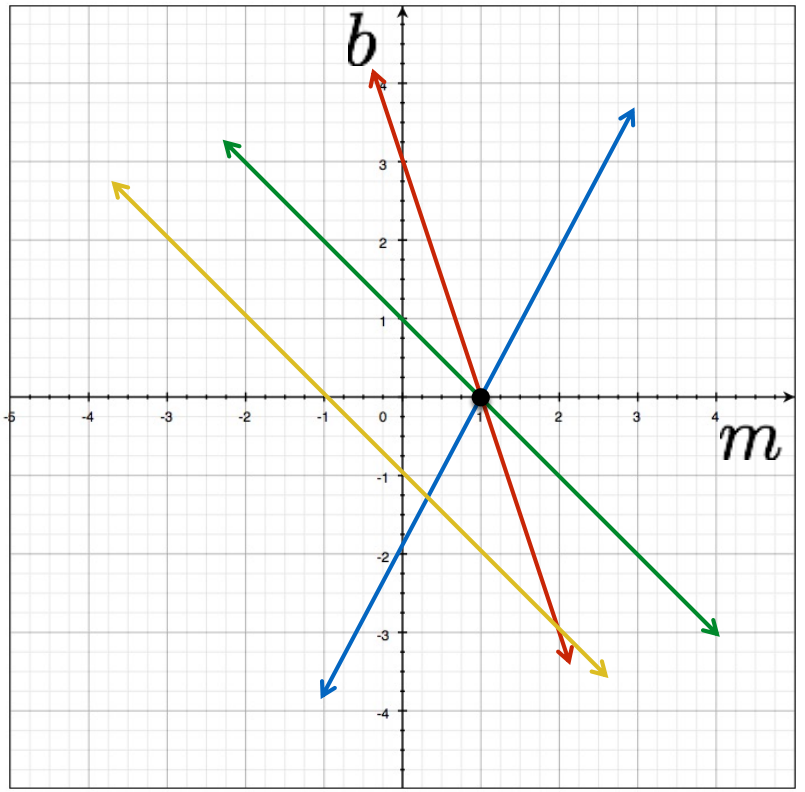


Image space

four points become ?



Parameter space

How would you find the best fitting line?

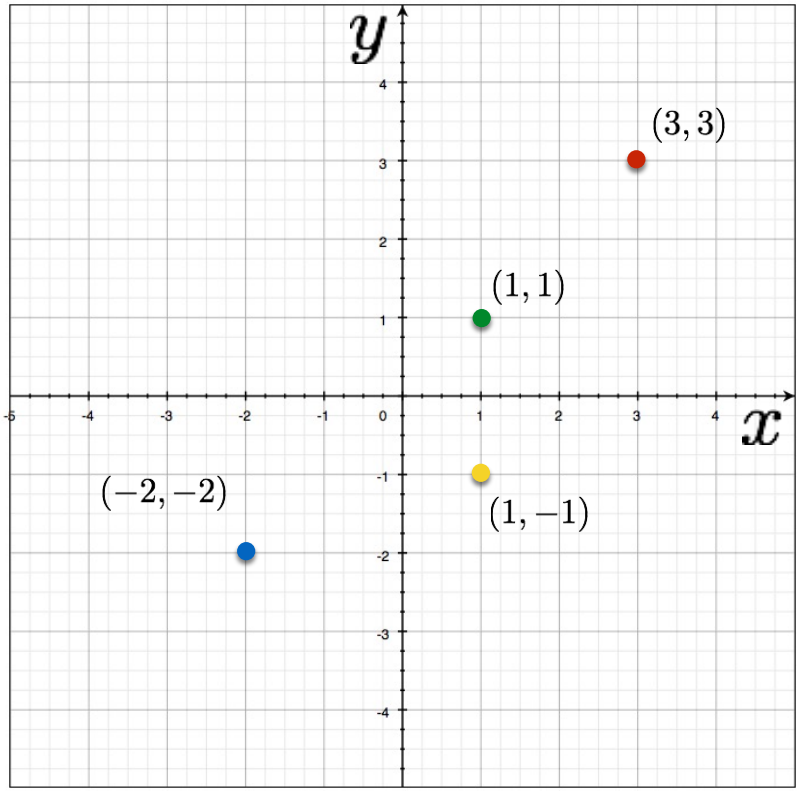
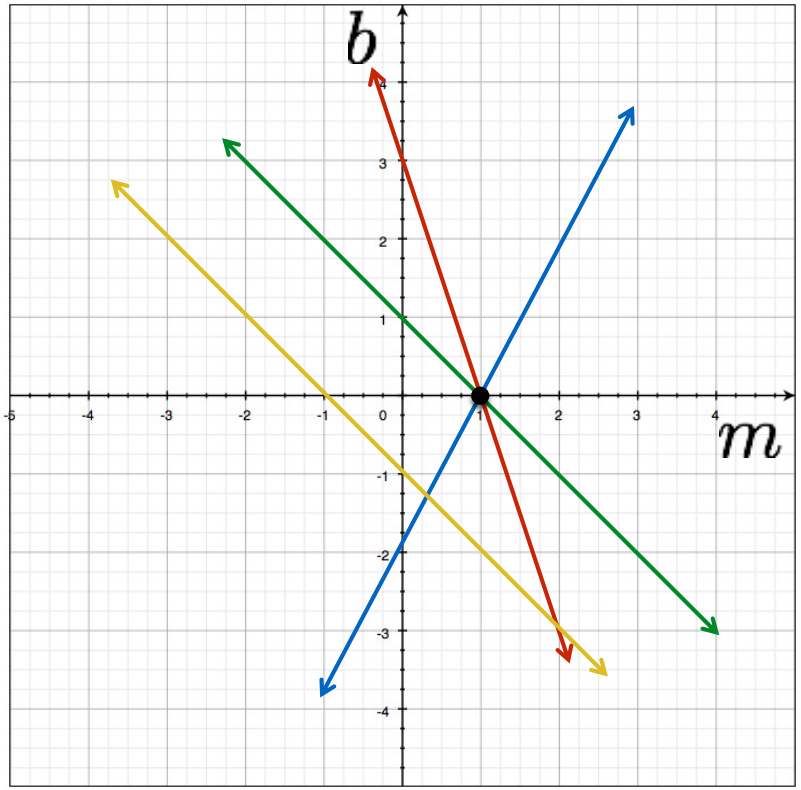


Image space



Parameter space

Is this method robust to measurement noise?

Is this method robust to outliers?

Line Detection by Hough Transform

Algorithm:

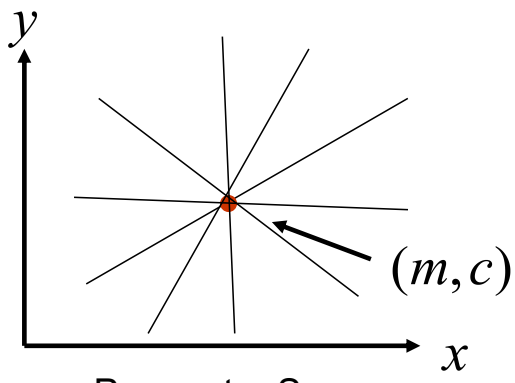
1. Quantize Parameter Space (m,c)

2. Create Accumulator Array $A(m,c)$

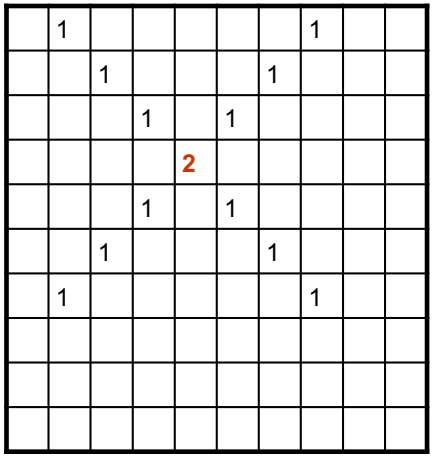
3. Set $A(m,c) = 0 \quad \forall m,c$

4. For each image edge (x_i, y_i)
   For each element in $A(m,c)$
   If (m,c) lies on the line: $c = -x_i m + y_i$
   Increment $A(m,c) = A(m,c) + 1$

5. Find local maxima in $A(m,c)$



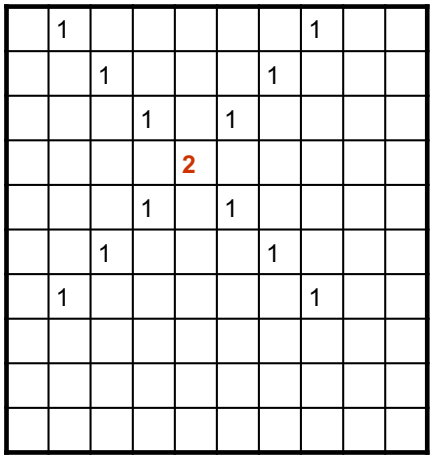
$A(m,c)$



Problems with parameterization

How big does the accumulator need to be for the parameterization (m, c)*?*

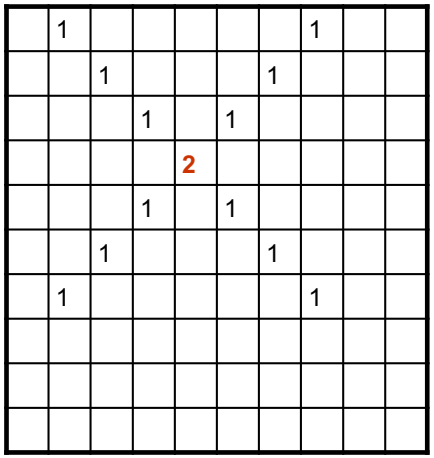
$A(m, c)$



Problems with parameterization

How big does the accumulator need to be for the parameterization (m, c)*?*

$A(m, c)$



The space of m is huge!

$$-\infty \leq m \leq \infty$$

The space of c is huge!

$$-\infty \leq c \leq \infty$$

Better Parameterization

Use normal form:

$$x\cos\theta + y\sin\theta = \rho$$

Given points (x_i, y_i) find (ρ,θ)

Hough Space Sinusoid

$$0 \le \theta \le 2\pi$$

$$0 \le \rho \le \rho_{\max}$$

(Finite Accumulator Array Size)

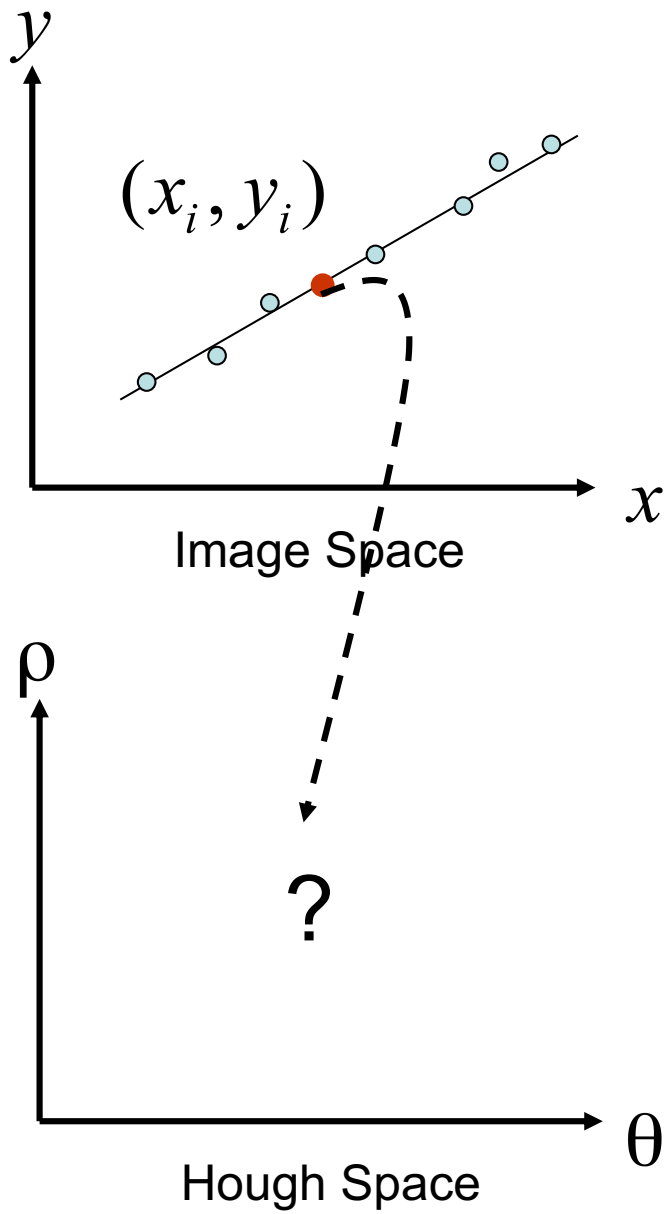


Image and parameter space

variables

$$y = mx + b$$

parameters

parameters

$$x\cos\theta + y\sin\theta = \rho$$

variables

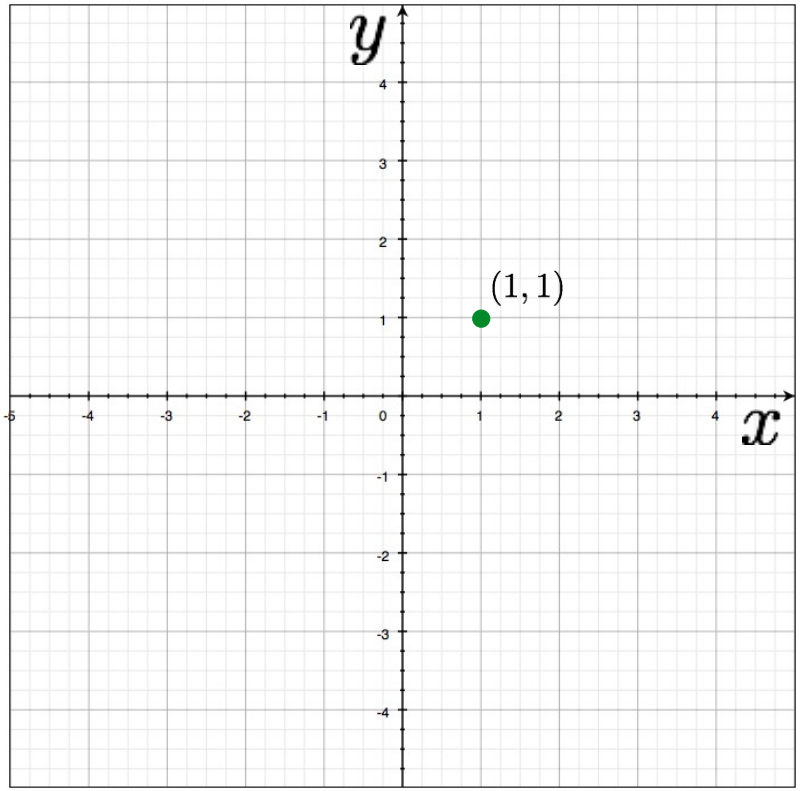
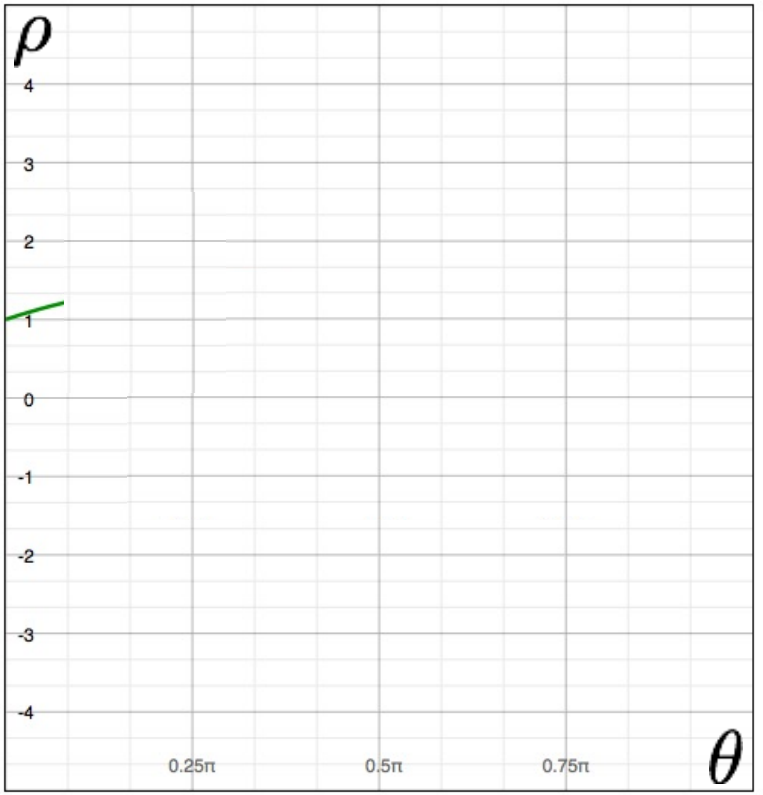


Image space

a point becomes?



Parameter space

Image and parameter space

variables

$$y = mx + b$$

parameters

parameters

$$x\cos\theta + y\sin\theta = \rho$$

variables

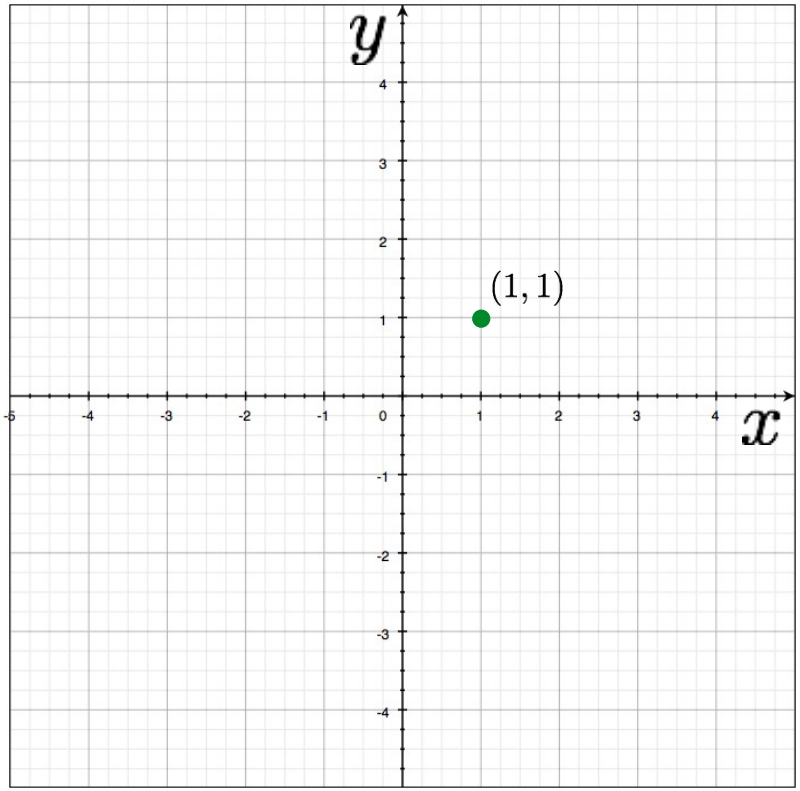
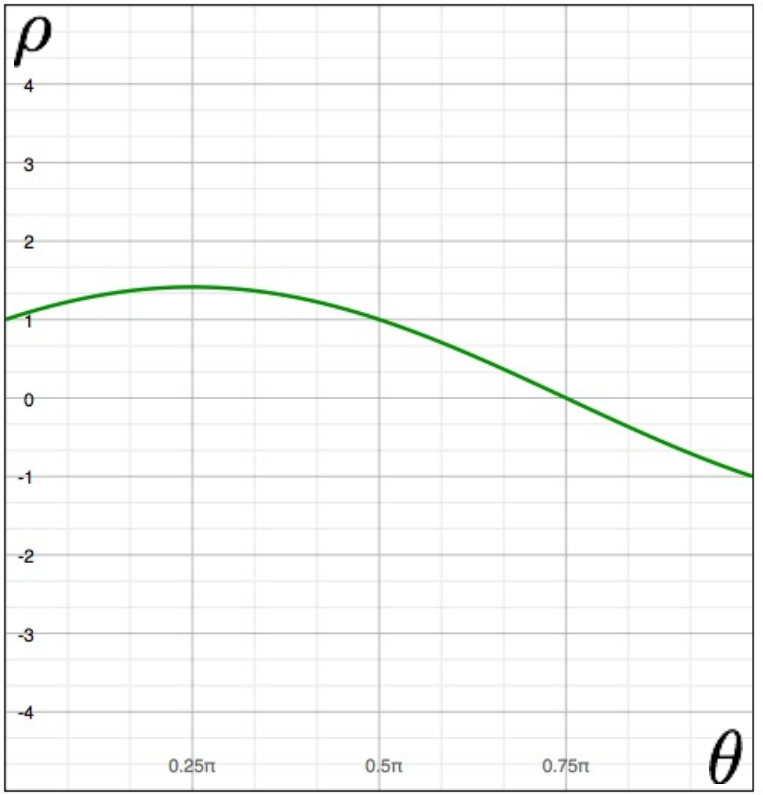


Image space

a point becomes a wave



Parameter space

Image and parameter space

variables

$$y = mx + b$$

parameters

$$x\cos\theta + y\sin\theta = \rho$$

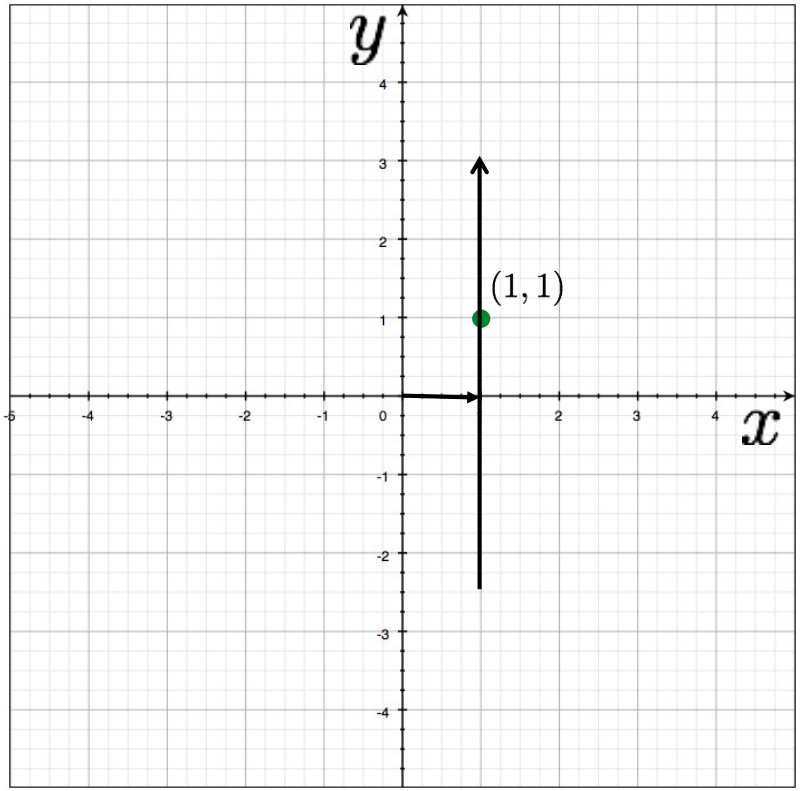
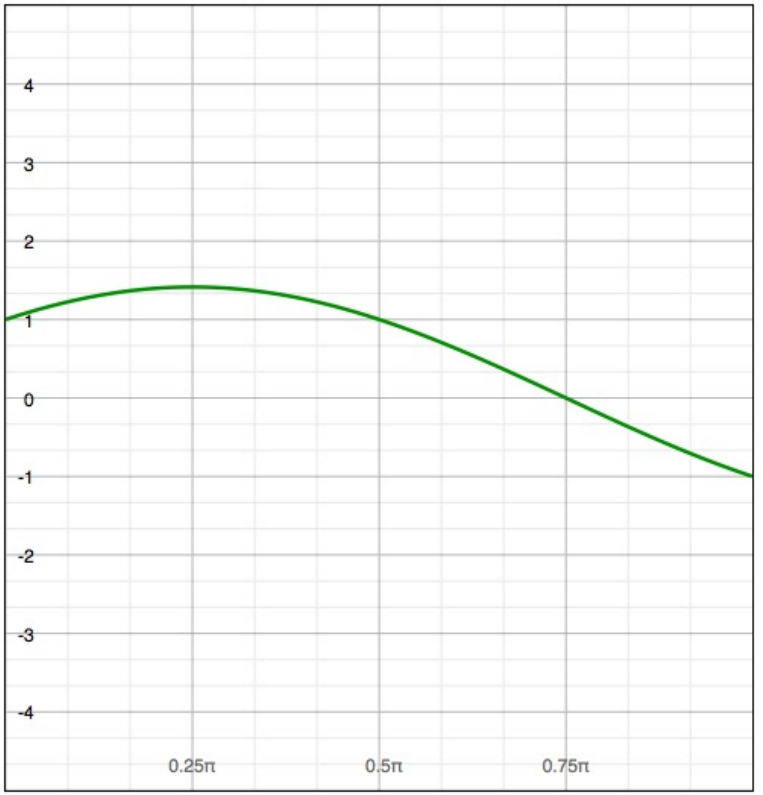


Image space

a line becomes?



Parameter space

Image and parameter space

variables

$$y = mx + b$$

parameters

$$x\cos\theta + y\sin\theta = \rho$$

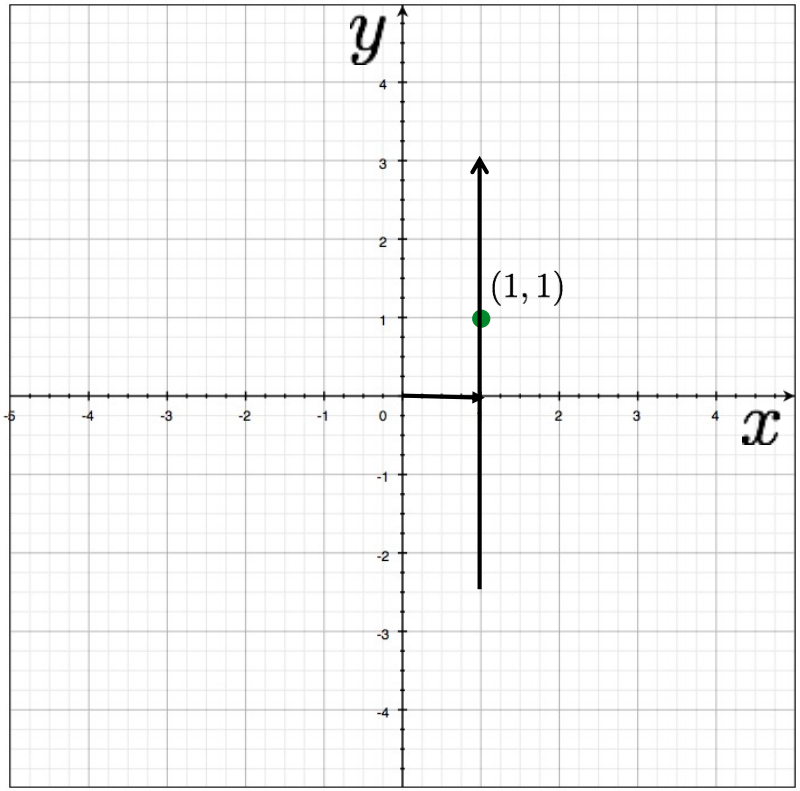
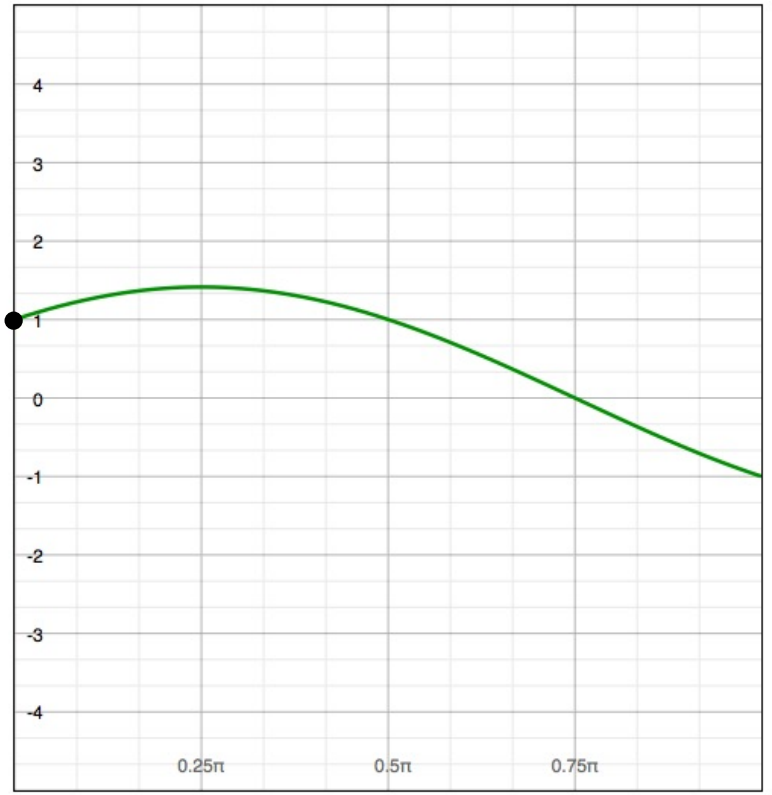


Image space

a line becomes a point



Parameter space

Image and parameter space

variables

$$y = mx + b$$

parameters

$$x\cos\theta + y\sin\theta = \rho$$

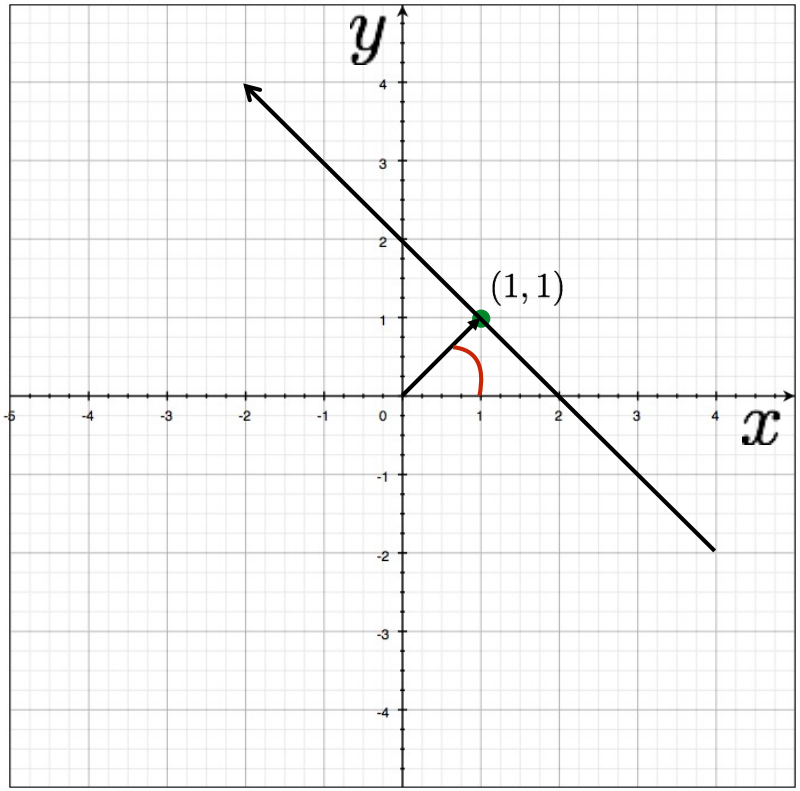
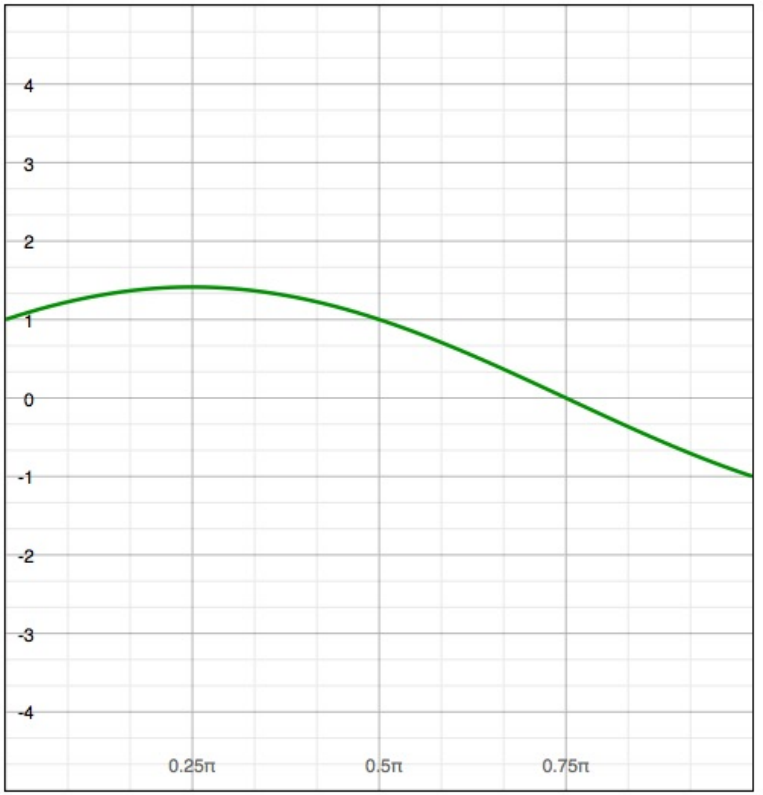


Image space

a line becomes?



Parameter space

Image and parameter space

variables

$$y = mx + b$$

parameters

$$x\cos\theta + y\sin\theta = \rho$$

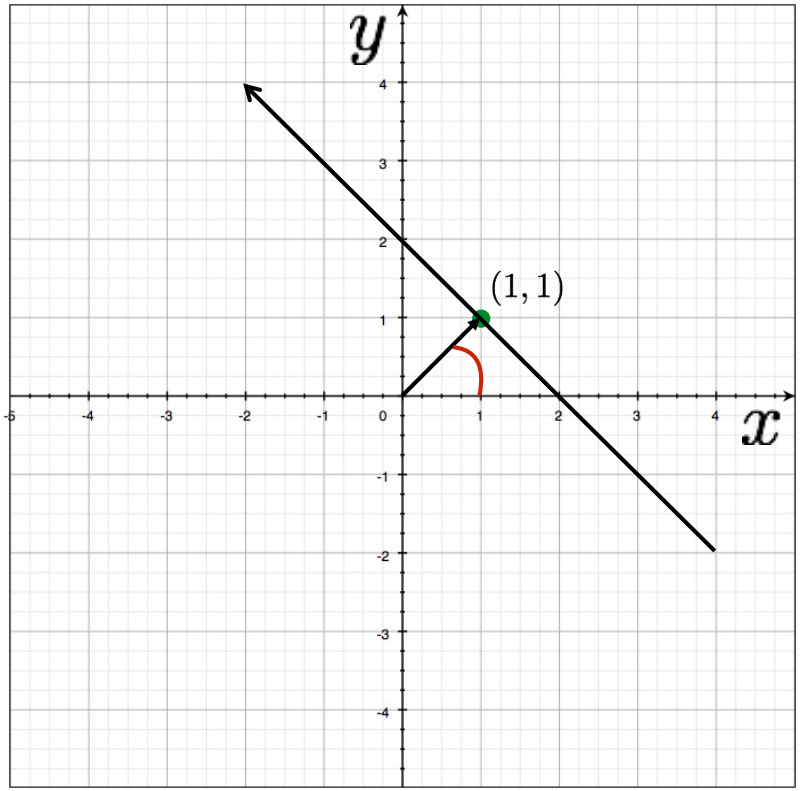
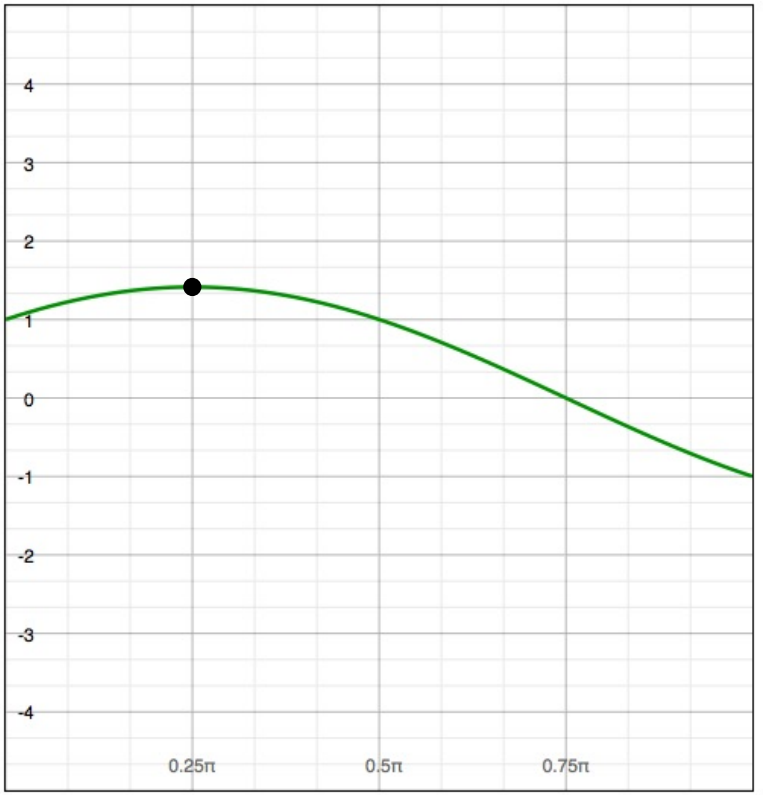


Image space

a line becomes a point



Parameter space

Image and parameter space

variables

$$y = mx + b$$

parameters

$$x\cos\theta + y\sin\theta = \rho$$

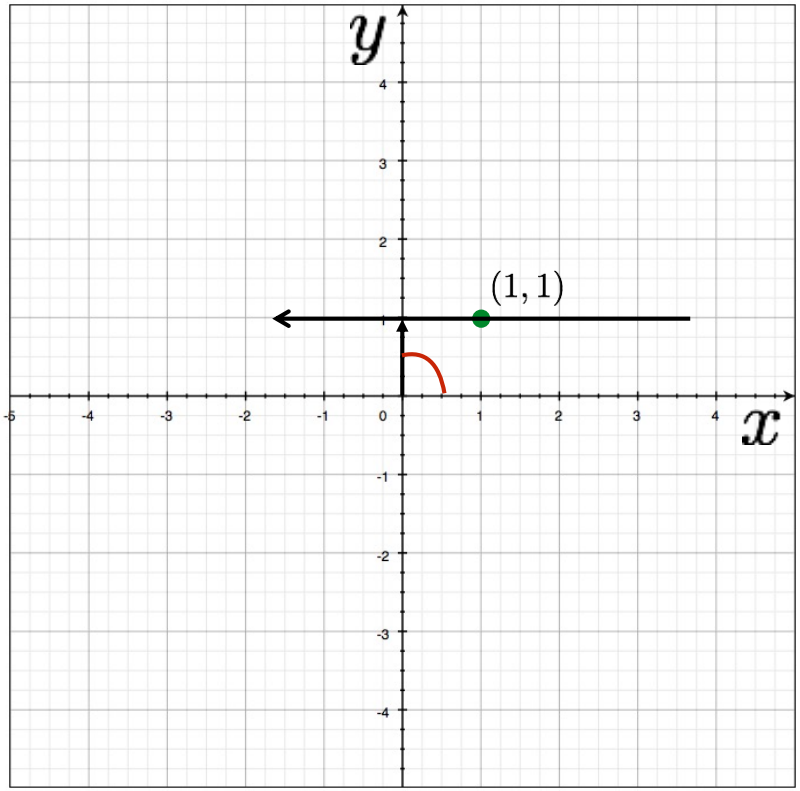
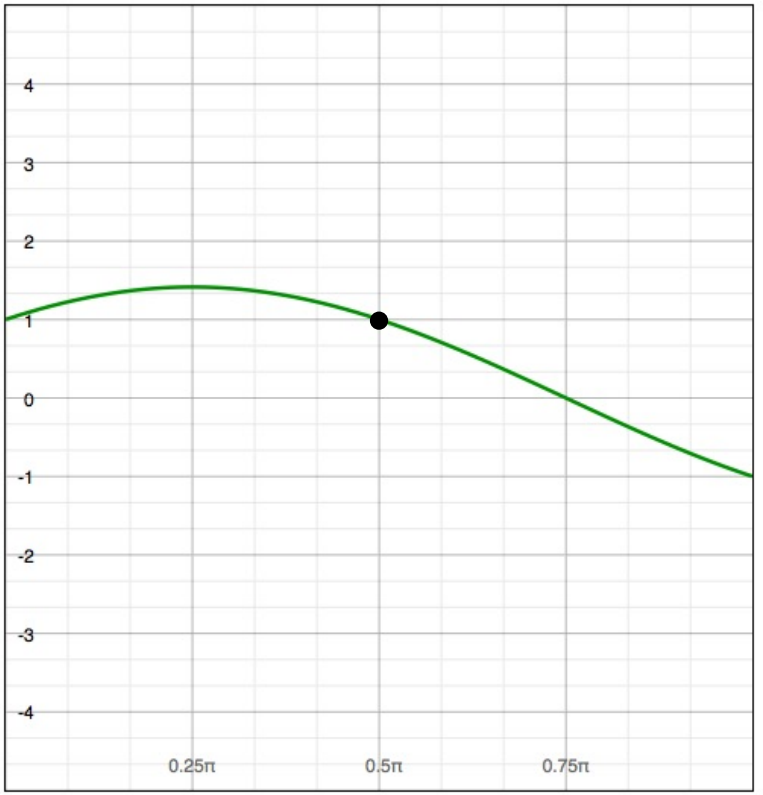


Image space

a line becomes a point



Parameter space

Image and parameter space

variables

$$y = mx + b$$

parameters

$$x\cos\theta + y\sin\theta = \rho$$

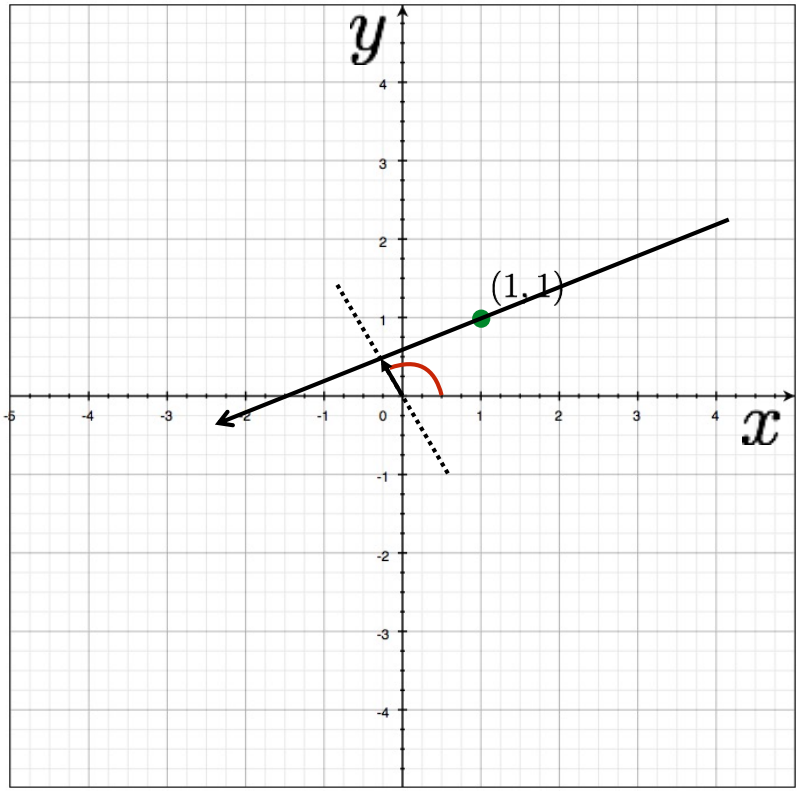
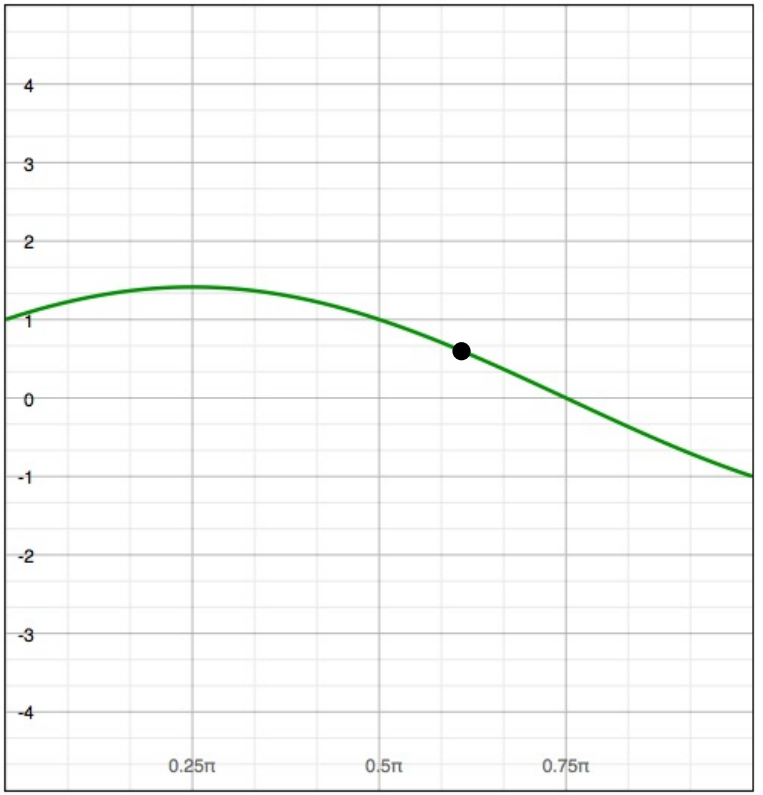


Image space

a line becomes a point



Parameter space

Image and parameter space

variables

$$y = mx + b$$

parameters

$$x\cos\theta + y\sin\theta = \rho$$

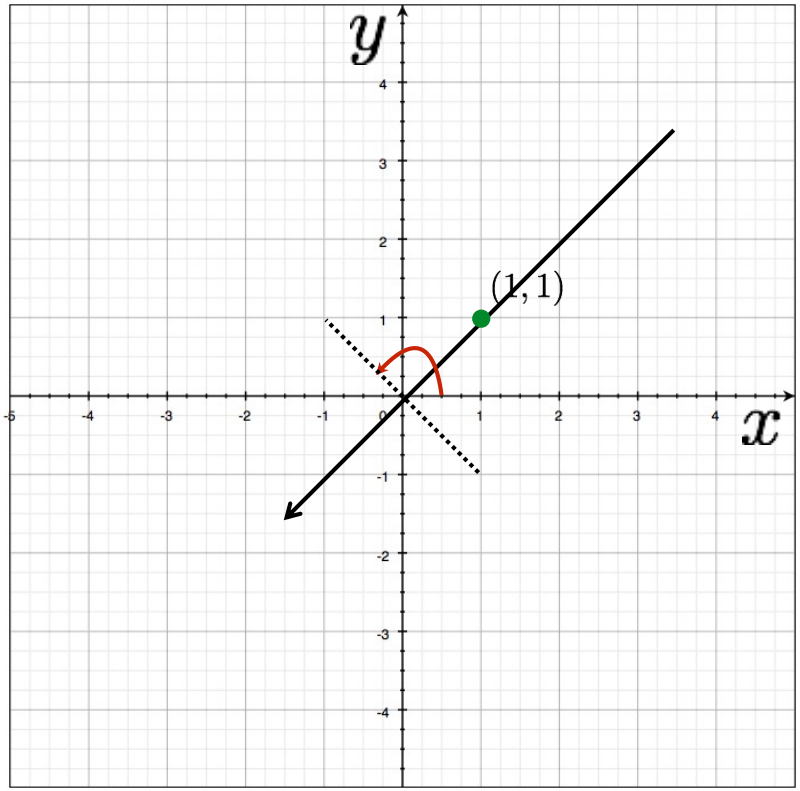
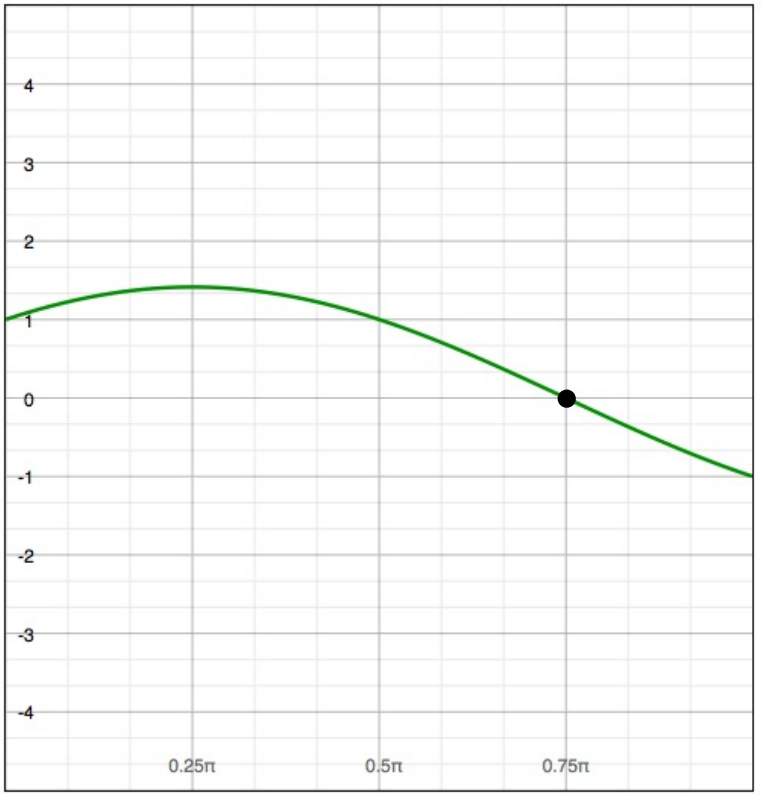


Image space

a line becomes a point



Parameter space

Image and parameter space

variables

$$y = mx + b$$

parameters

$$x\cos\theta + y\sin\theta = \rho$$

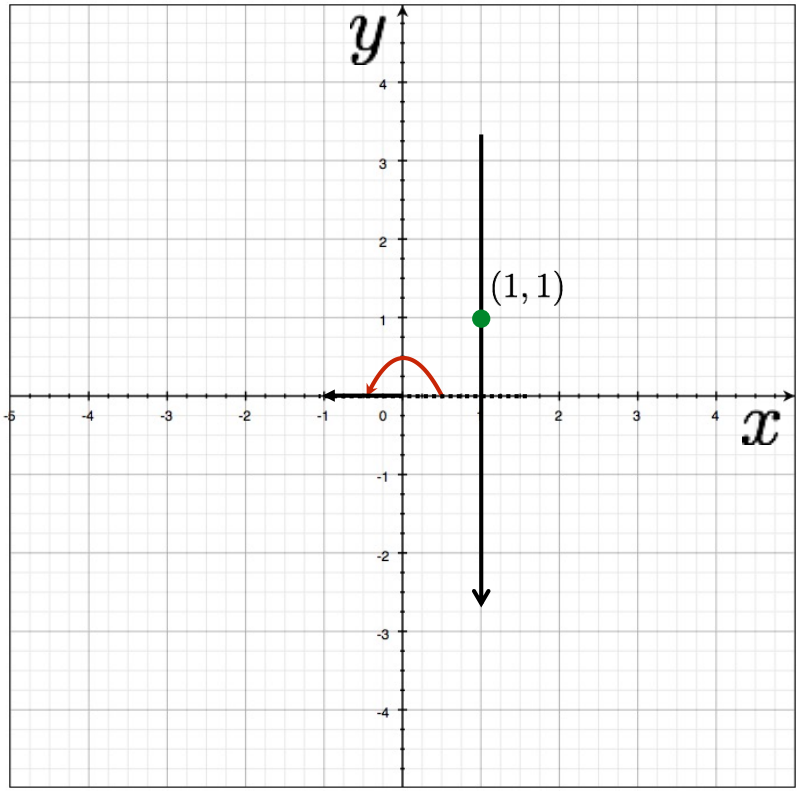
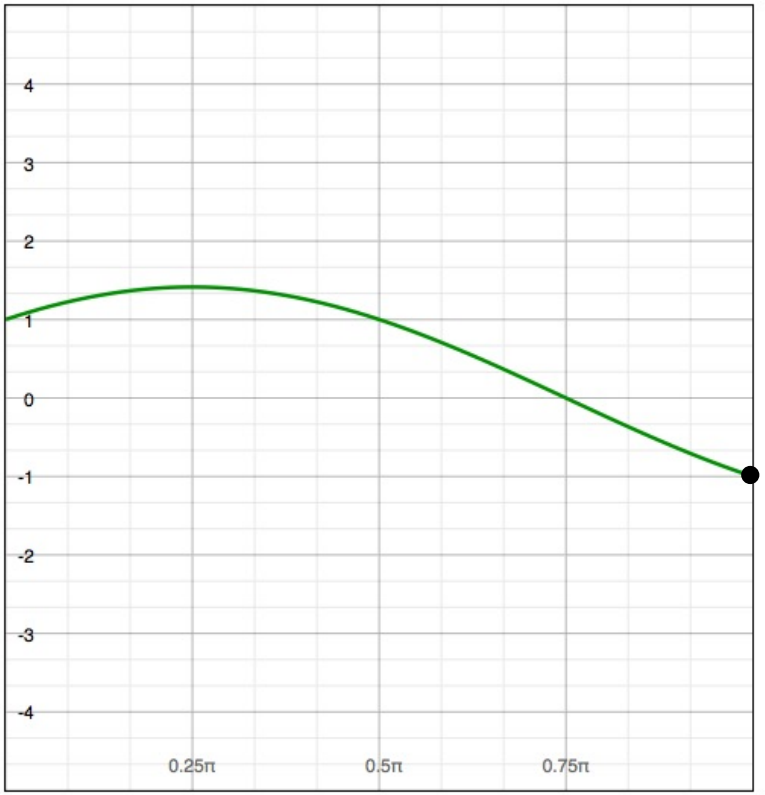


Image space

a line becomes a point



Parameter space

Image and parameter space

variables

$$y = mx + b$$

parameters

$$x\cos\theta + y\sin\theta = \rho$$

Wait …why is rho negative?

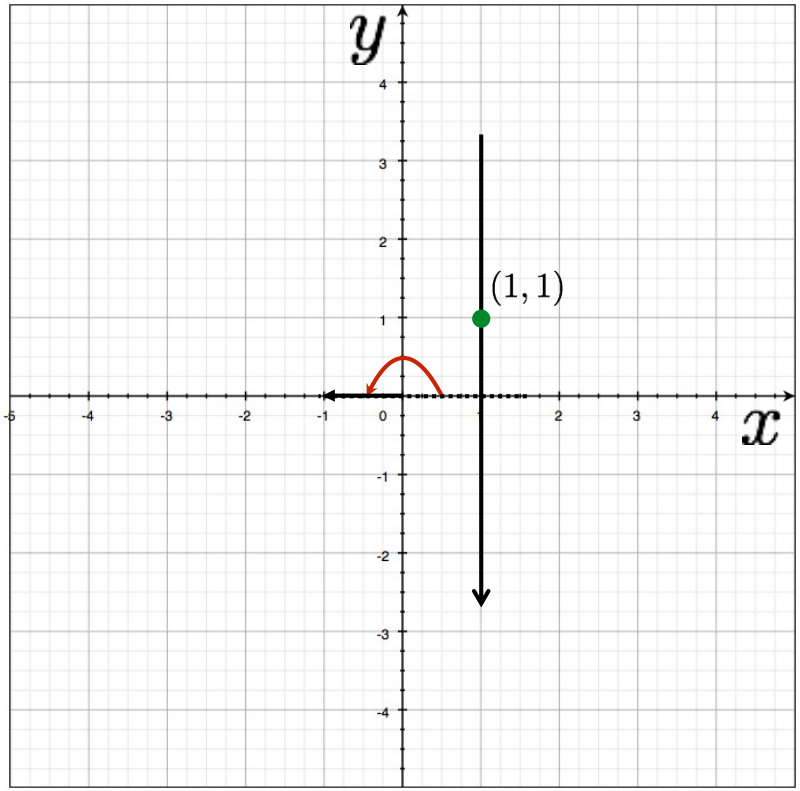
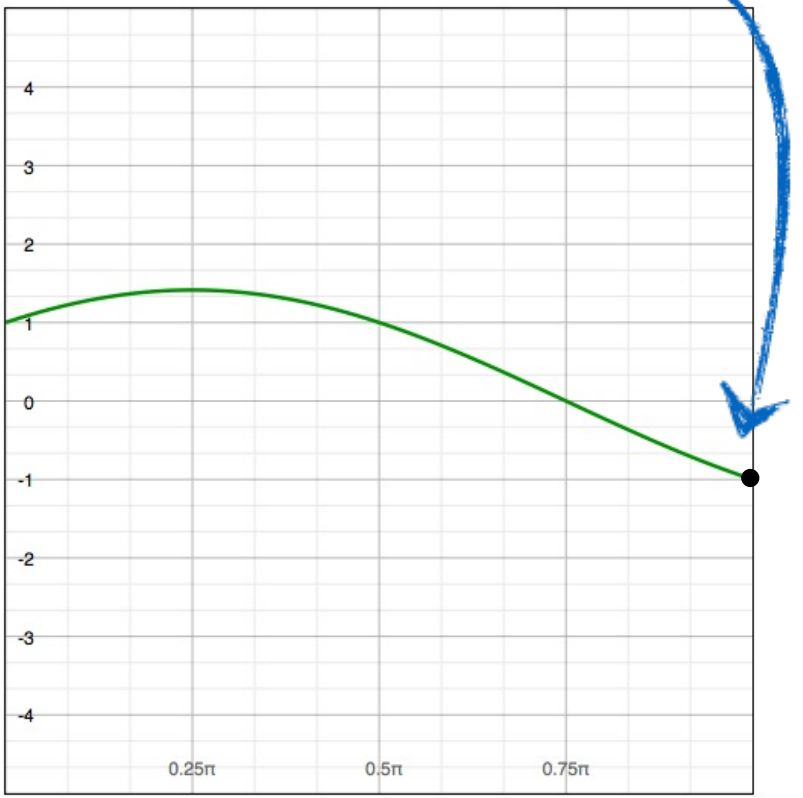


Image space

a line becomes a point



Parameter space

Image and parameter space

variables

$$y = mx + b$$

parameters

$$x\cos\theta + y\sin\theta = \rho$$

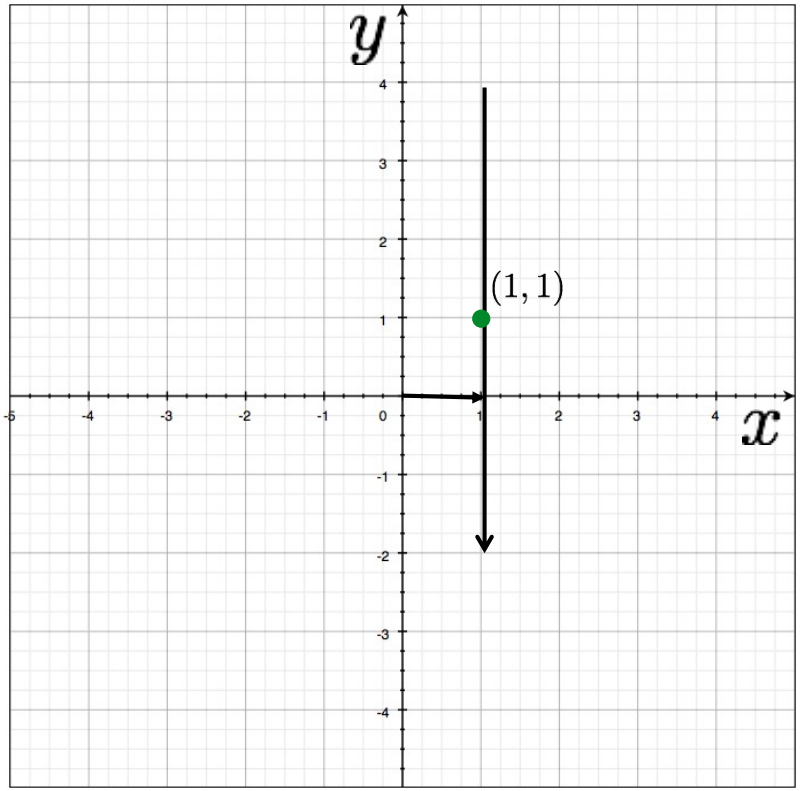
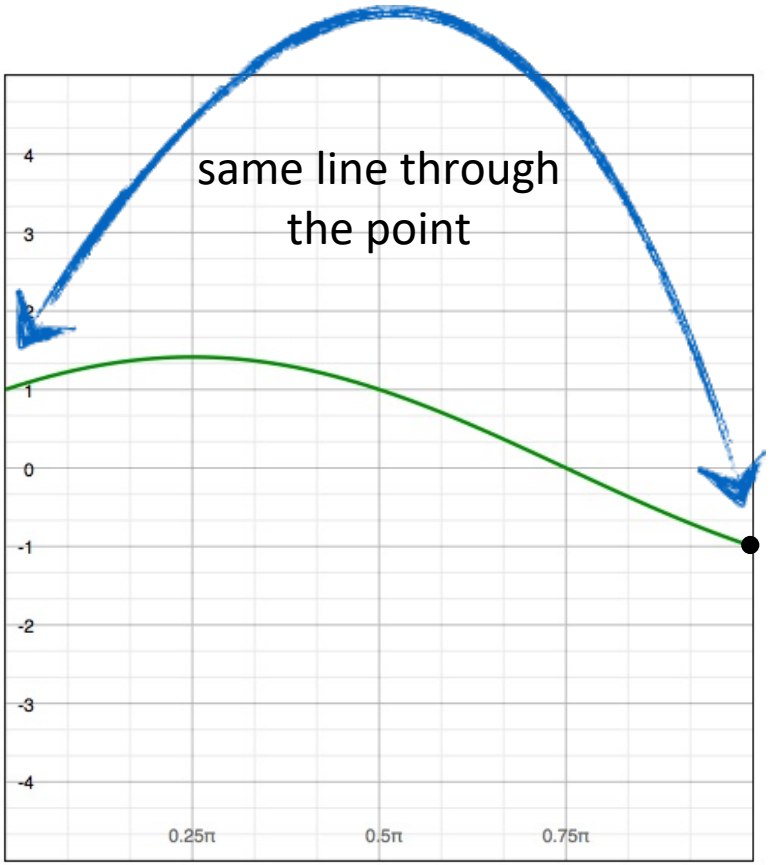


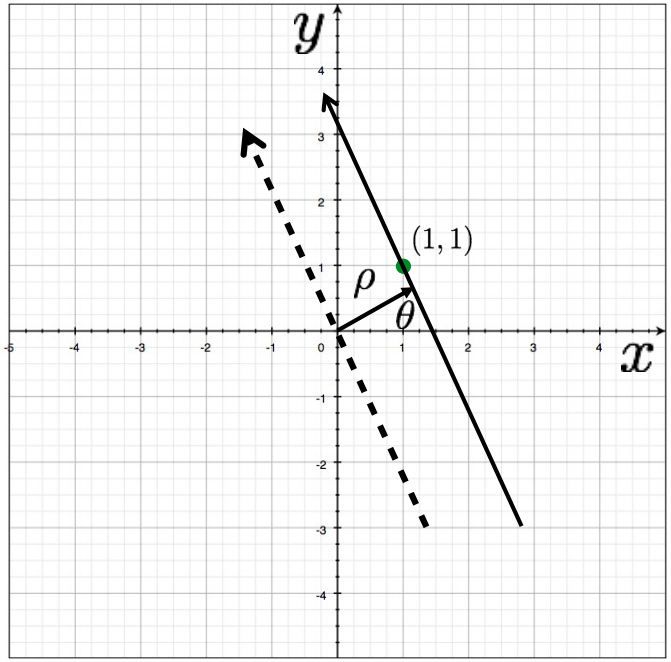
Image space

a line becomes a point



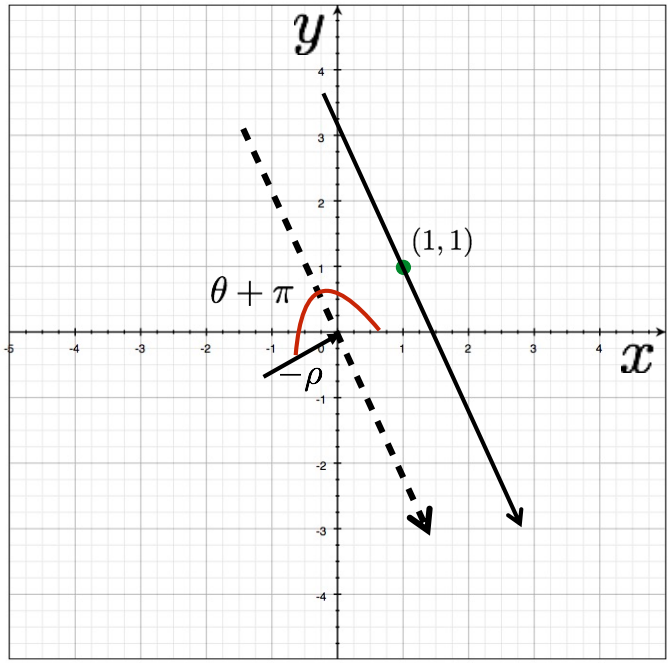
Parameter space

There are two ways to write the same line:



Positive rho version:

$$x\cos\theta + y\sin\theta = \rho$$



Negative rho version:

$$x\cos(\theta+\pi) + y\sin(\theta+\pi) = -\rho$$

Recall:

$$\sin(\theta) = -\sin(\theta+\pi)$$
$$\cos(\theta) = -\cos(\theta+\pi)$$

Image and parameter space

variables

$$y = mx + b$$

parameters

$$x\cos\theta + y\sin\theta = \rho$$

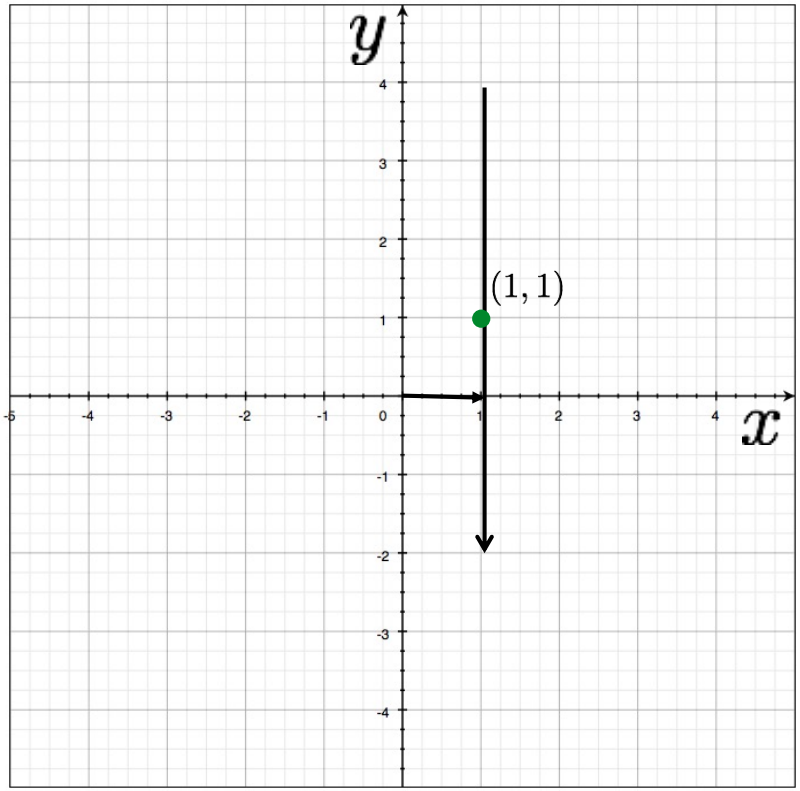
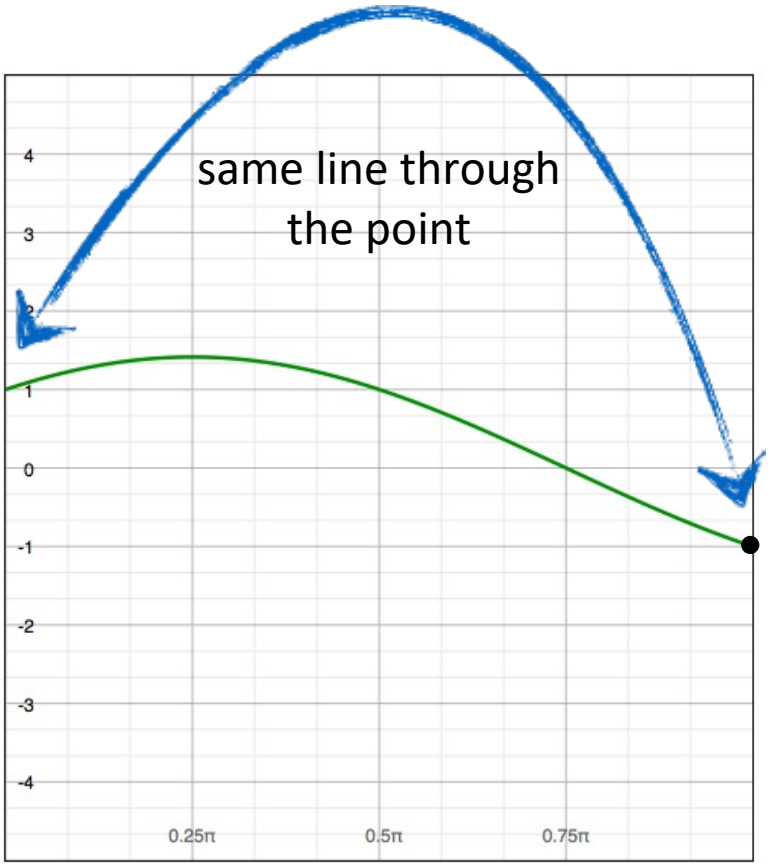


Image space

a line becomes a point



Parameter space

Image and parameter space

variables

$$y = mx + b$$

parameters

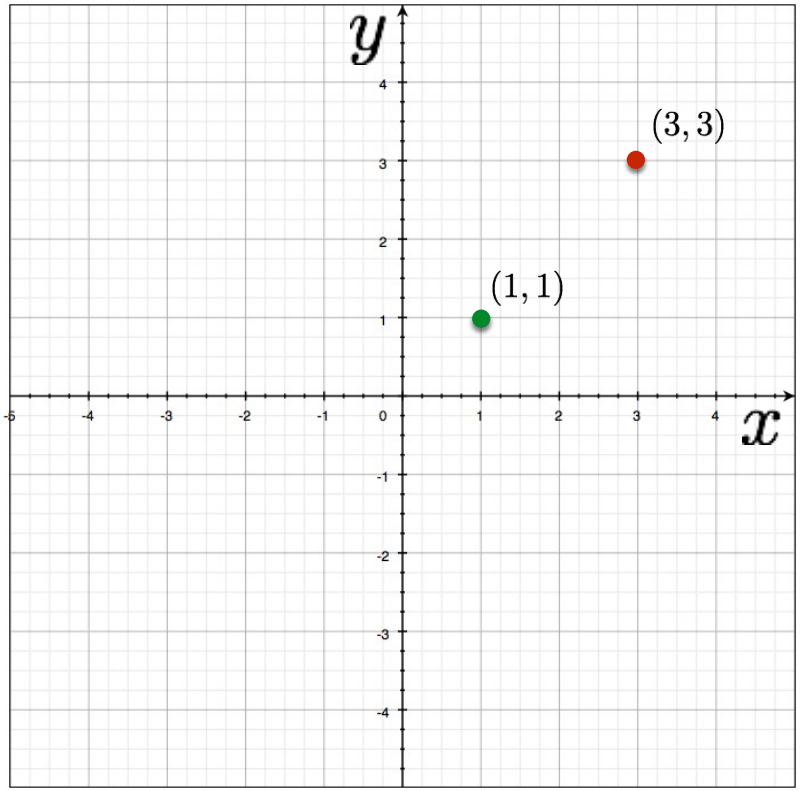
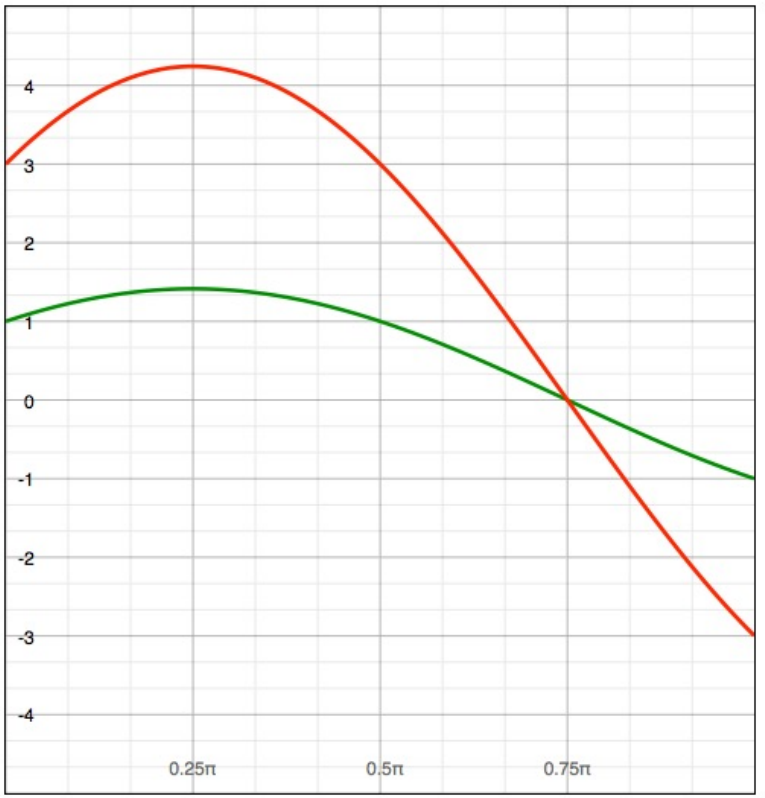


Image space

two points become ?



Parameter space

Image and parameter space

variables

$$y = mx + b$$

parameters

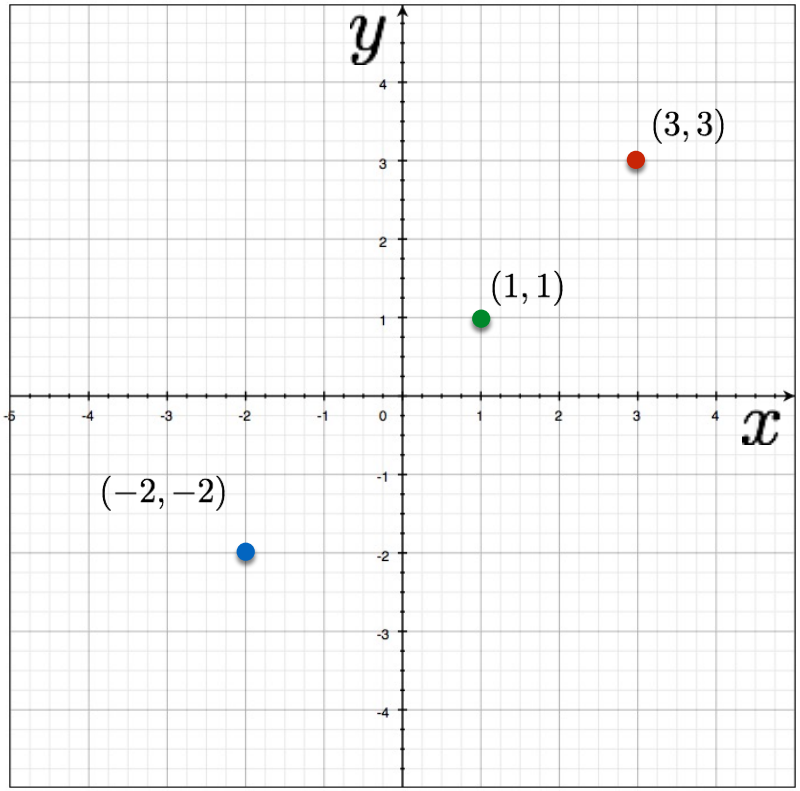
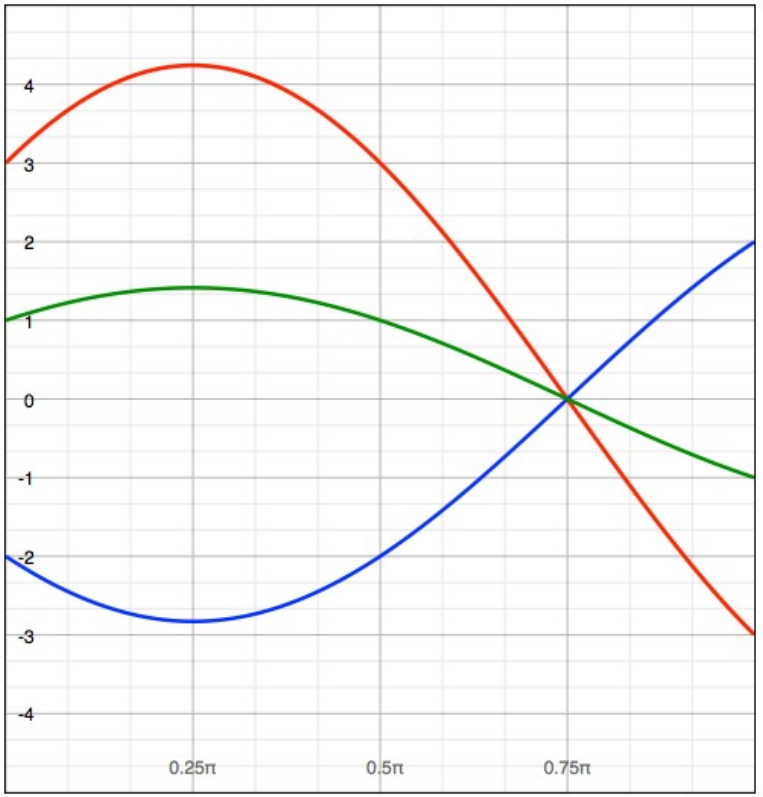


Image space

three points become ?



Parameter space

Image and parameter space

variables

$$y = mx + b$$

parameters

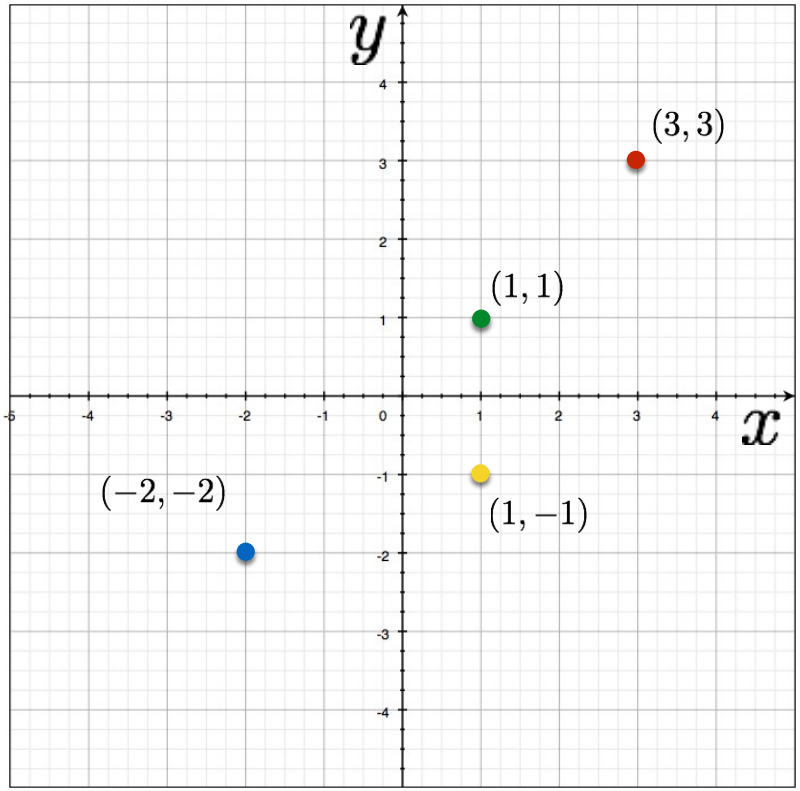
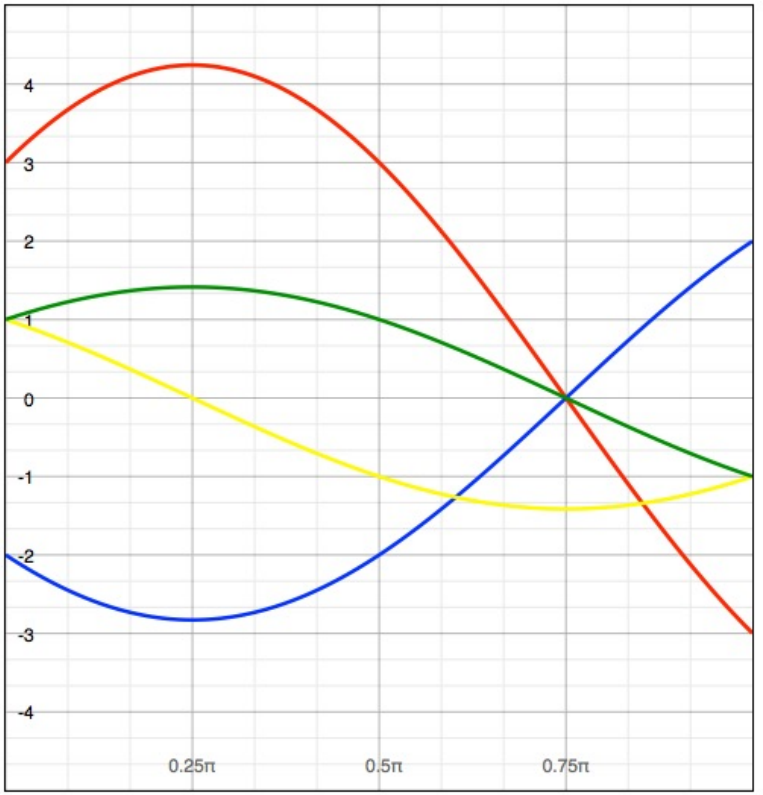


Image space

four points become ?



Parameter space

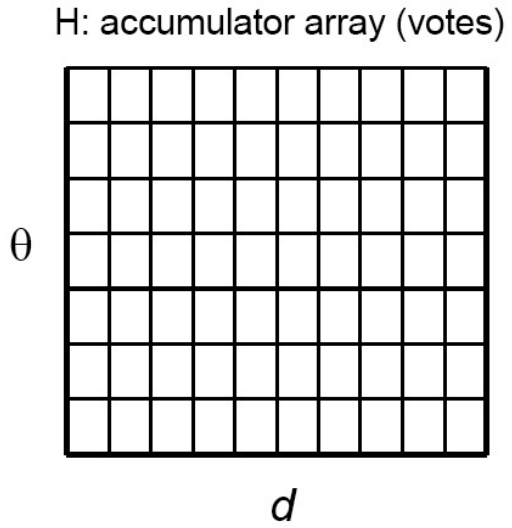
Implementation

1. Initialize accumulator H to all zeros

2. For each edge point (x,y) in the image
   For θ = 0 to 180
   ρ = x cos θ + y sin θ
   H(θ, ρ) = H(θ, ρ) + 1
   end
   end

3. Find the value(s) of (θ, ρ) where H(θ, ρ) is a local maximum

4. The detected line in the image is given by
   ρ = x cos θ + y sin θ



NOTE: Watch your coordinates. Image origin is top left!

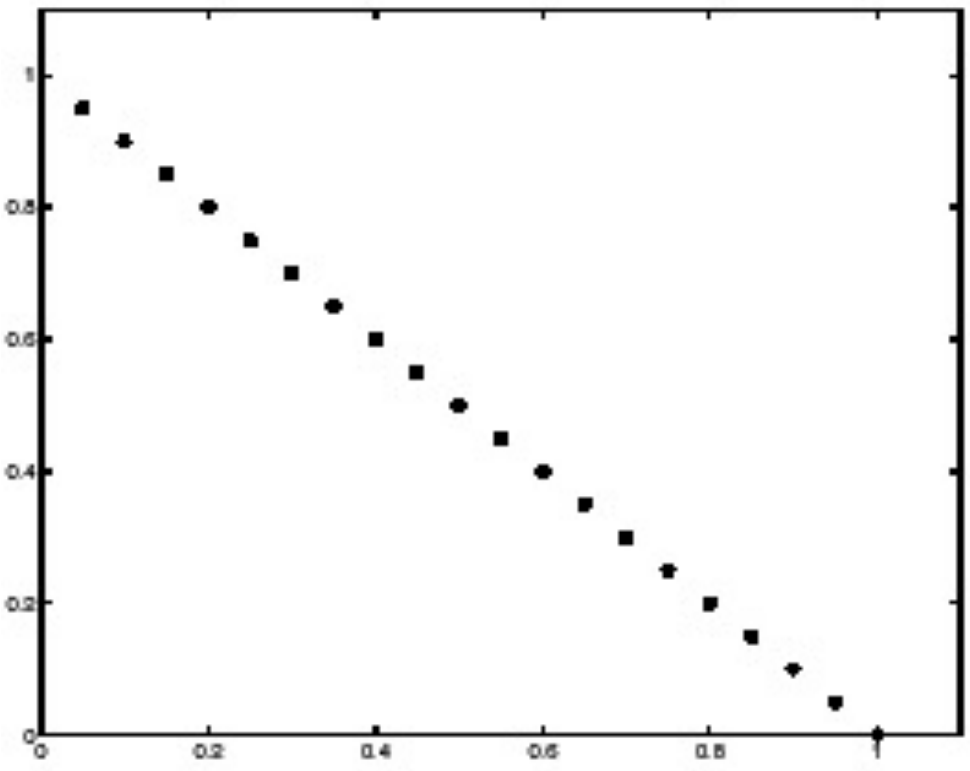
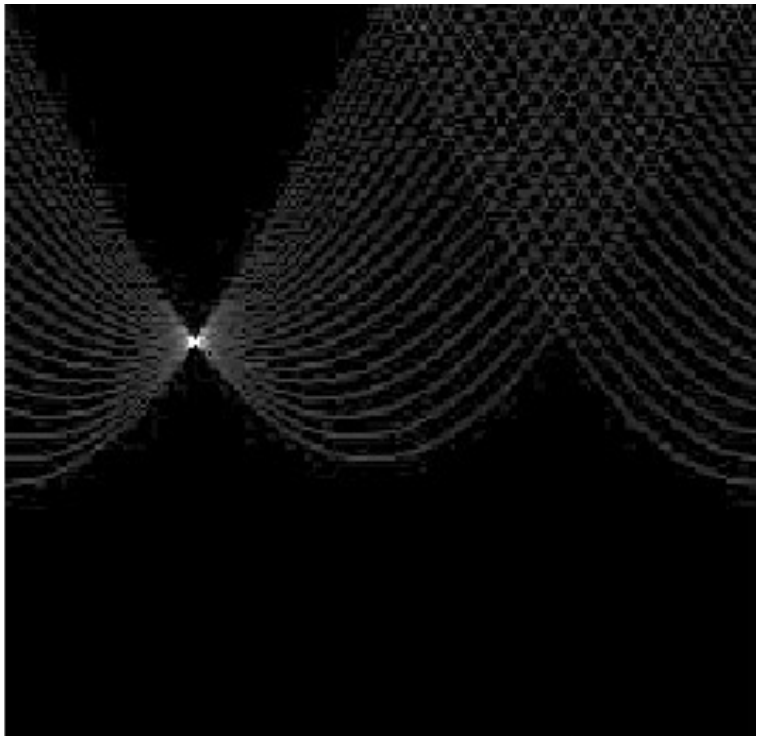


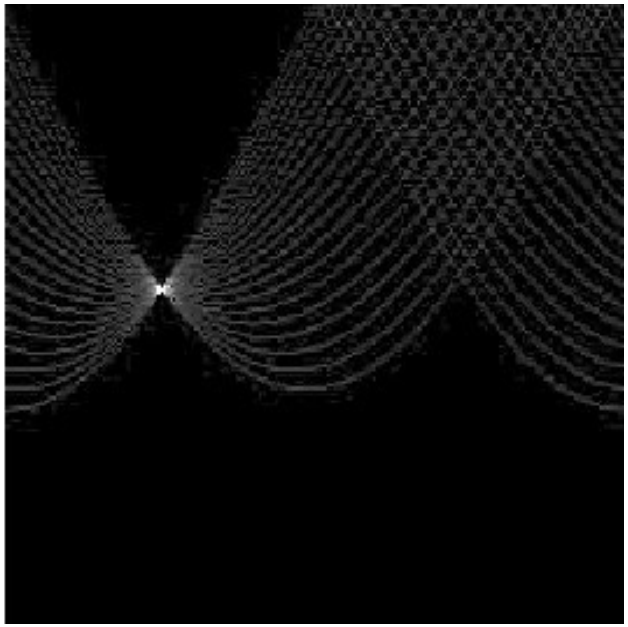
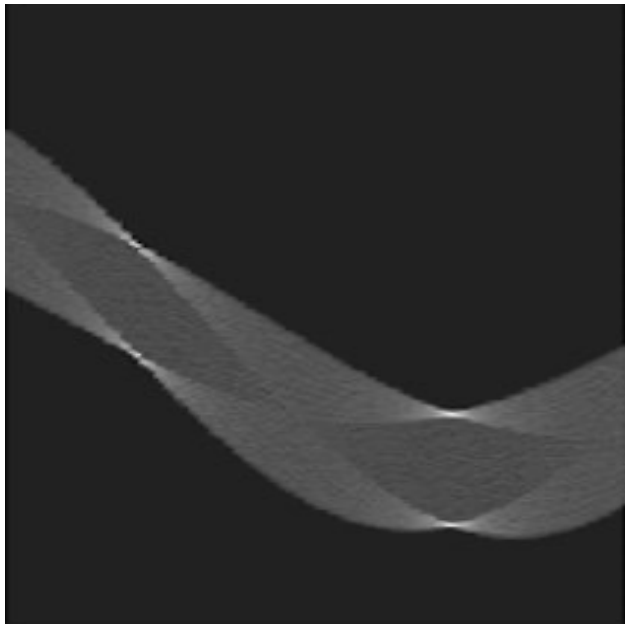
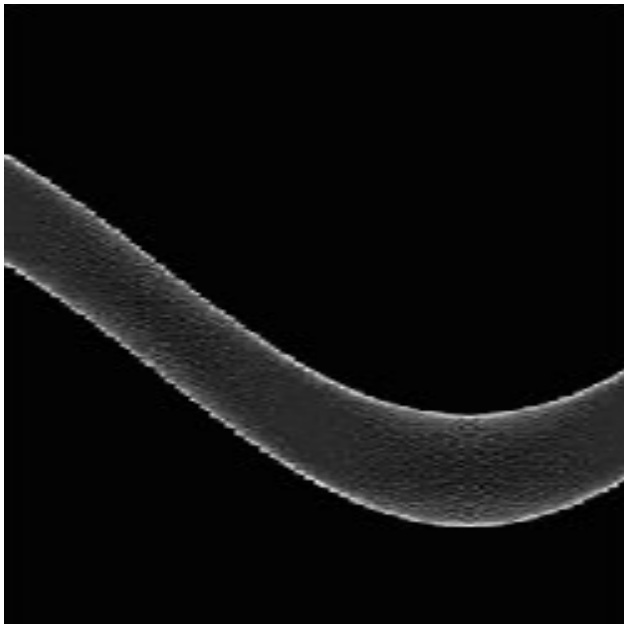
Image space



Votes

Basic shapes

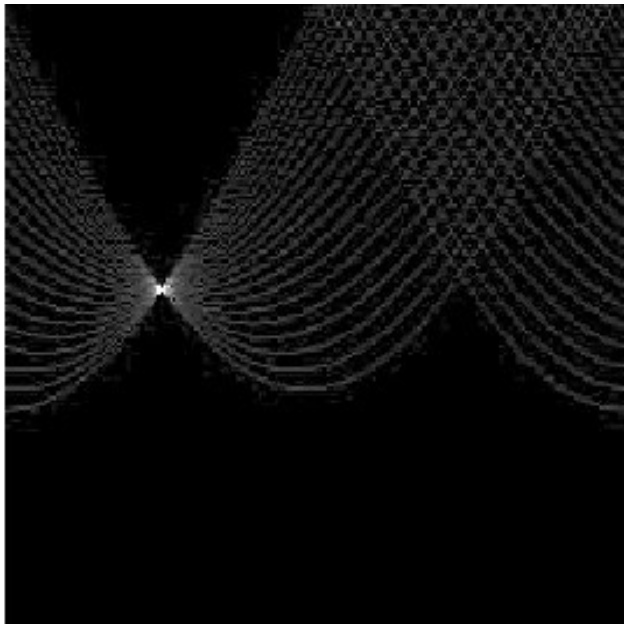
(in parameter space)

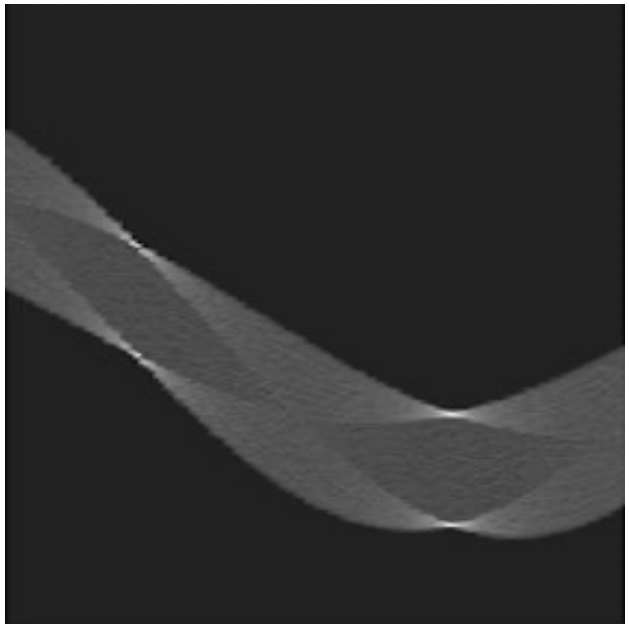
can you guess the shape?

Basic shapes

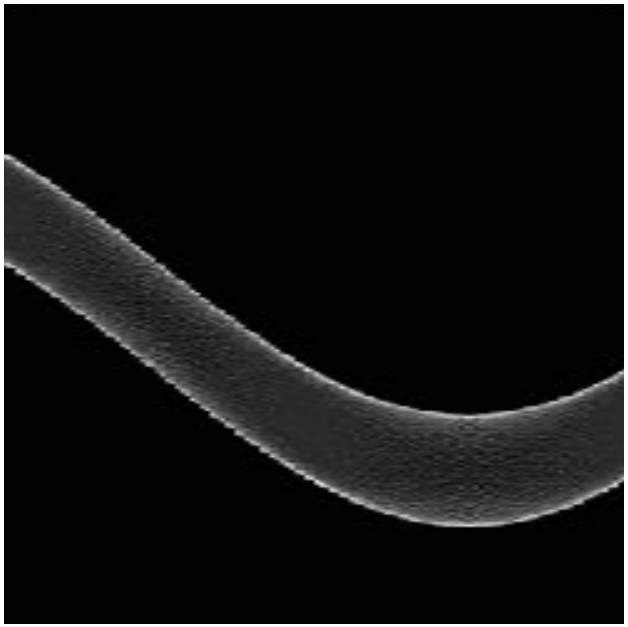
(in parameter space)



line

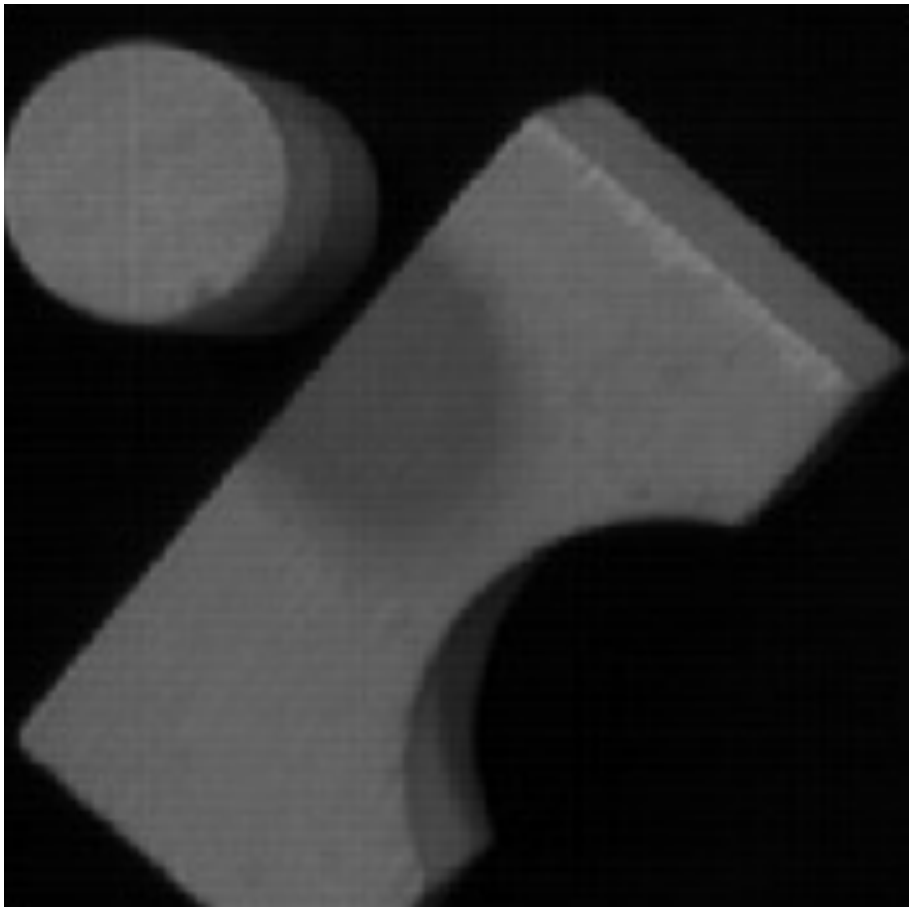


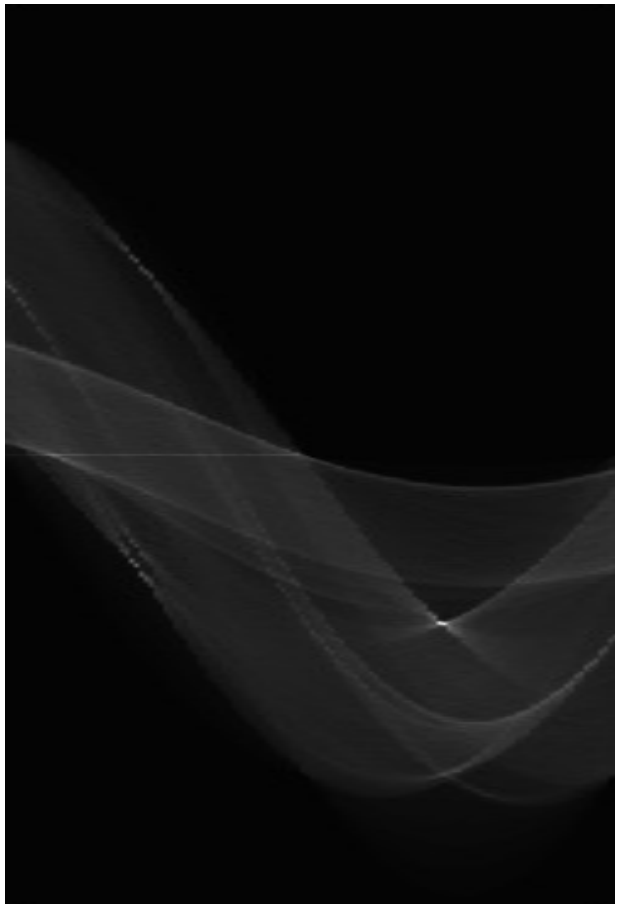
rectangle



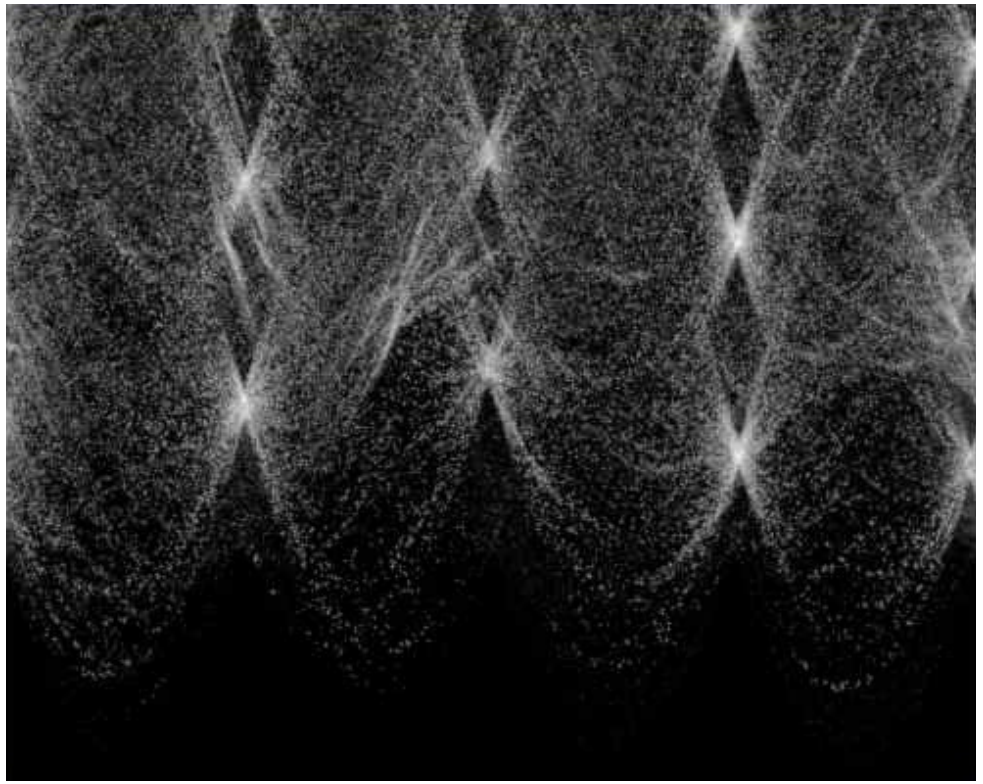
circle

Basic Shapes





More complex image



In practice, measurements are noisy…

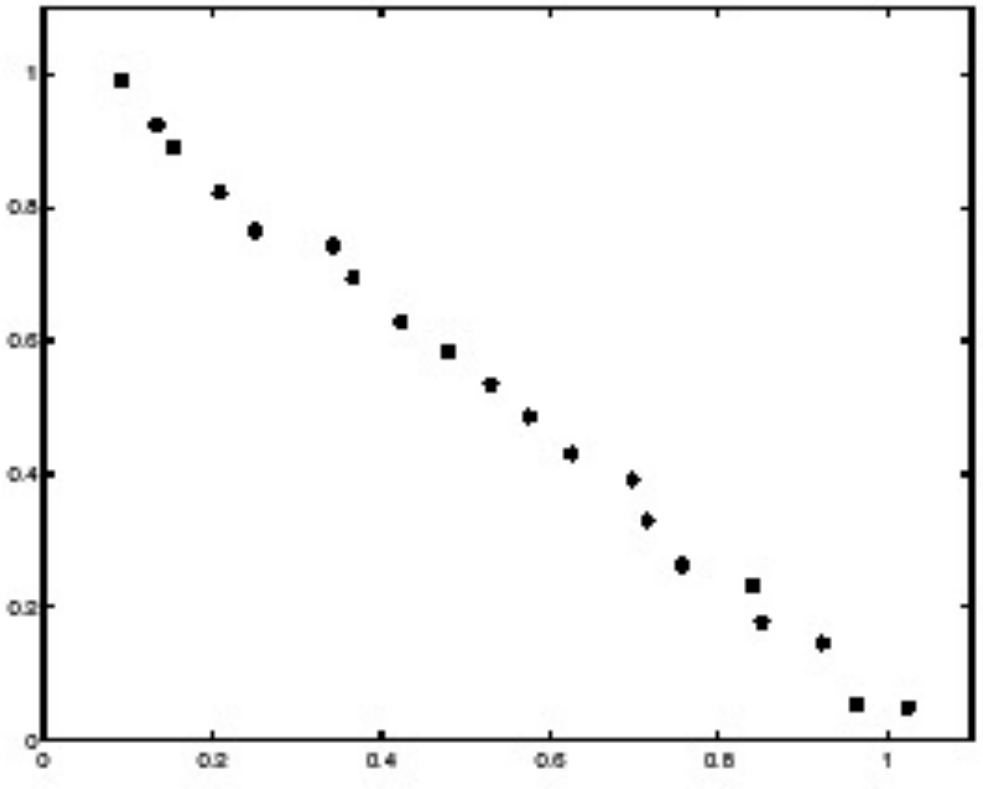
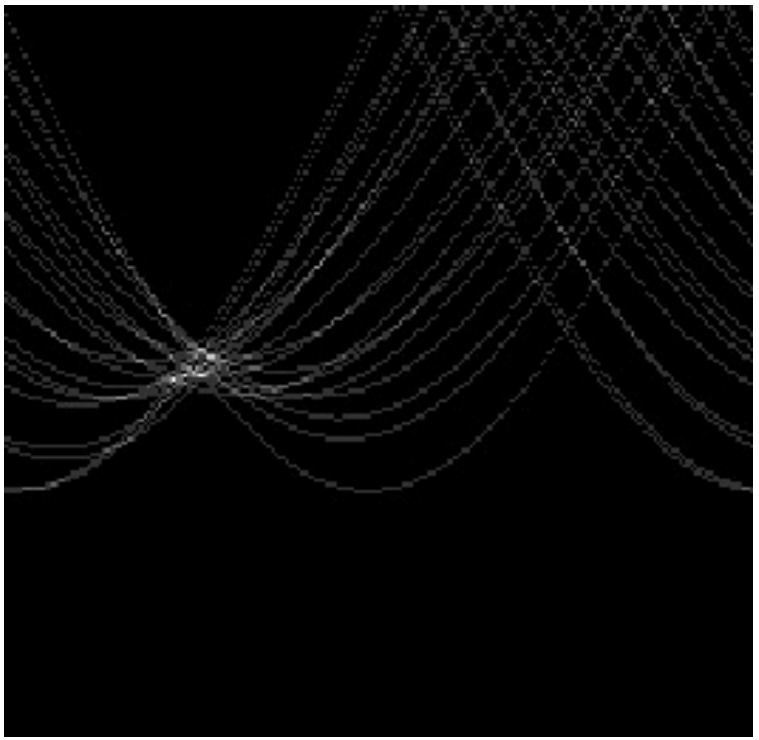


Image space



Votes

Too much noise …

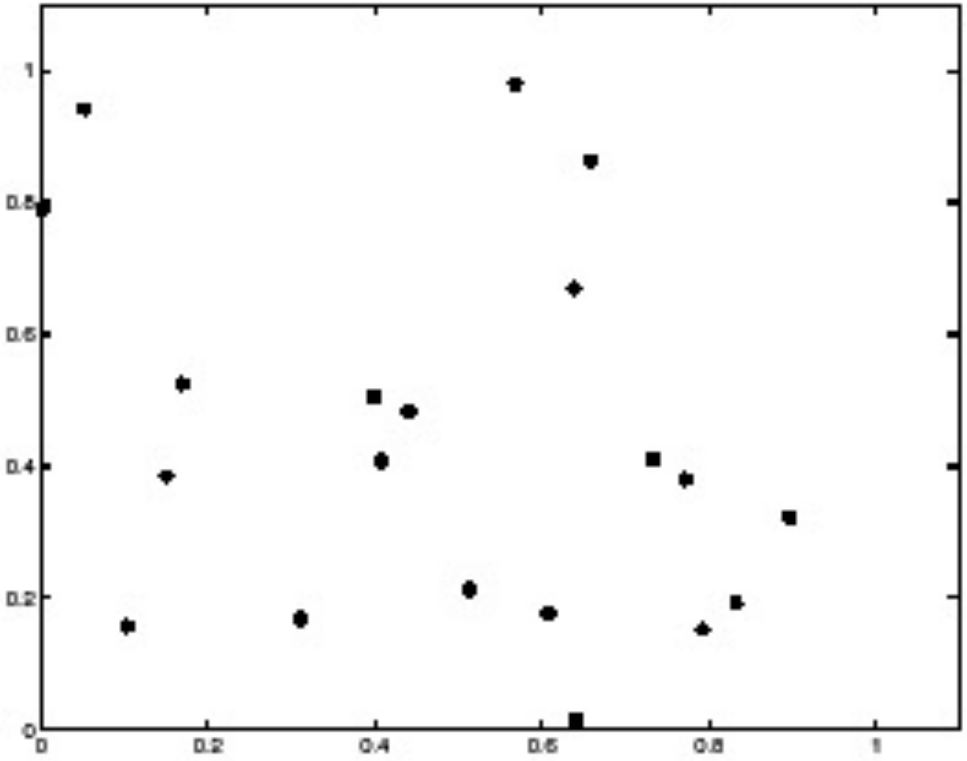
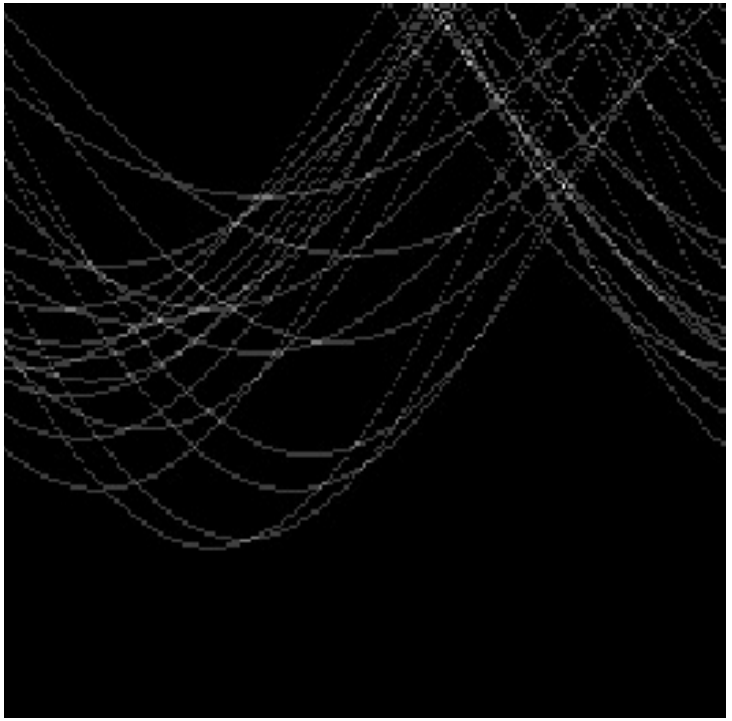


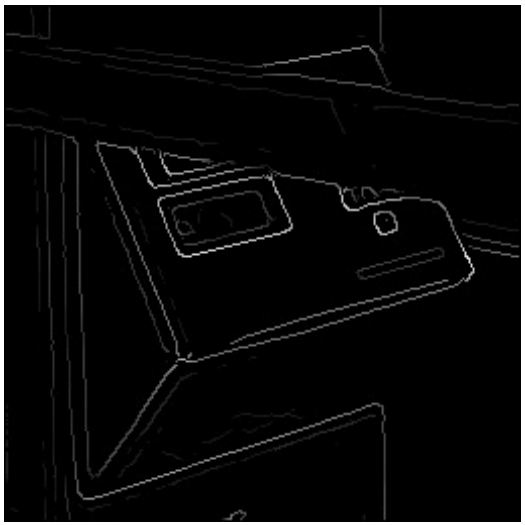
Image space



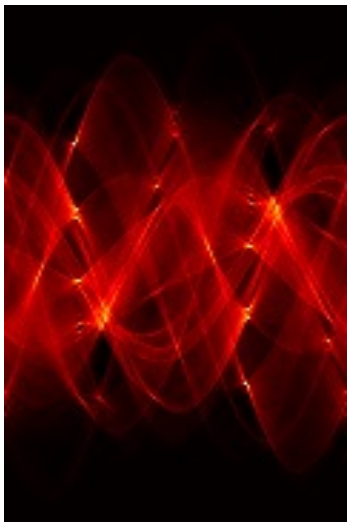
Votes

Real-world example

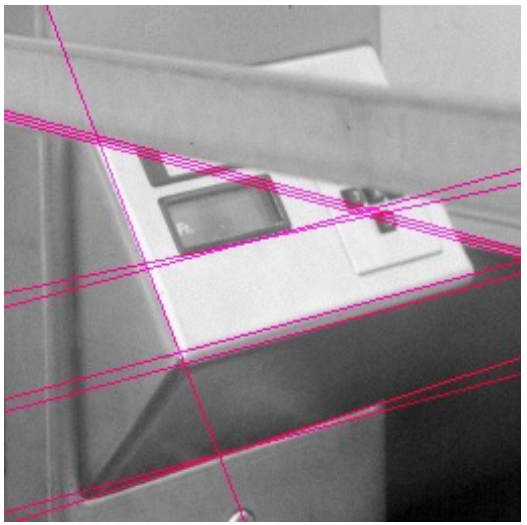
Original



Edges



parameter space



Hough Lines