

---

# Computer Vision – 4b

## Segmentation by Fitting

MAP-I Doctoral Programme

*Miguel Tavares Coimbra*

---

# Outline

---

- The Hough Transform
- Fitting Lines
- Fitting Curves
- Fitting as a Probabilistic Inference Problem

Acknowledgements: These slides follow Forsyth and Ponce's "Computer Vision: A Modern Approach", Chapter 15.

---

# Topic: The Hough Transform

---

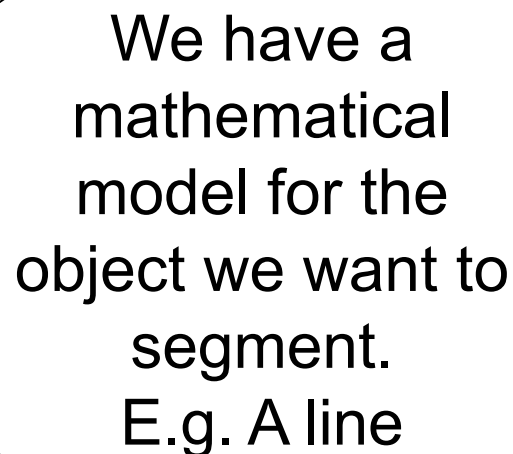
- The Hough Transform
- Fitting Lines
- Fitting Curves
- Fitting as a Probabilistic Inference Problem

---

# Fitting and Clustering

---

- Another definition for segmentation:
  - Pixels belong together because they conform to some model.
- Sounds like “Segmentation by Clustering” ...
- Key difference:
  - The model is now **explicit**.



We have a mathematical model for the object we want to segment.  
E.g. A line

---

# Hough Transform

---

- Elegant method for direct object recognition
- Edges need not be connected
- Complete object need not be visible
- Key Idea: Edges **VOTE** for the possible model

# Image and Parameter Spaces

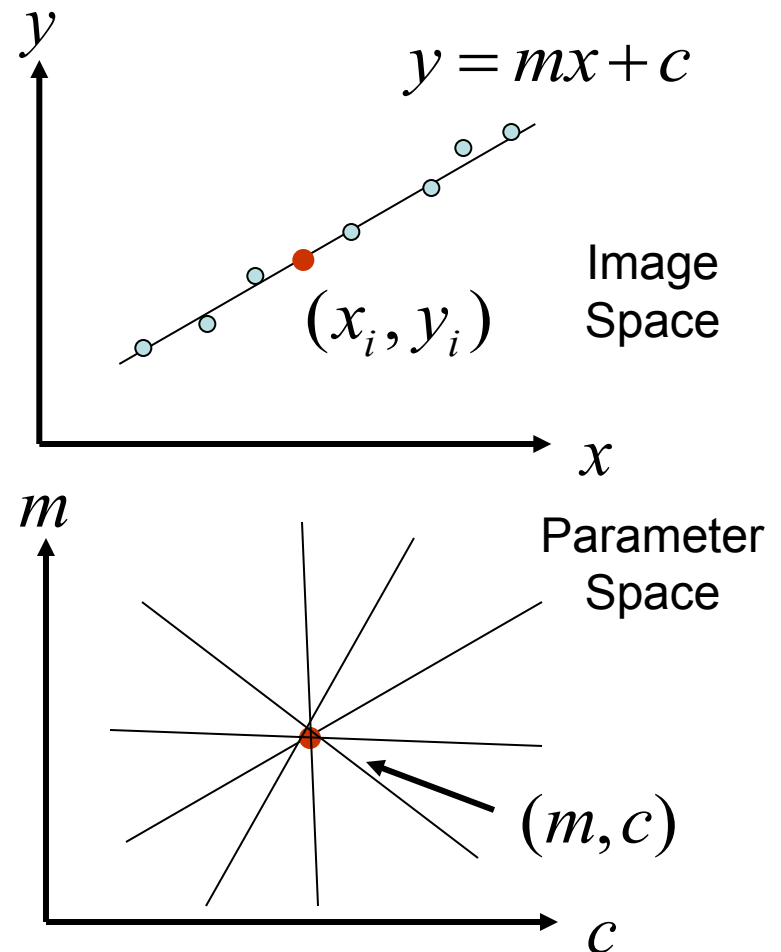
Equation of Line:  $y = mx + c$

Find:  $(m, c)$

Consider point:  $(x_i, y_i)$

$$y_i = mx_i + c \quad \text{or} \quad c = -x_i m + y_i$$

Parameter space also called Hough Space





# Better Parameterization

NOTE:  $-\infty \leq m \leq \infty$   
Large Accumulator

More memory and computations

Improvement: (Finite Accumulator Array Size)

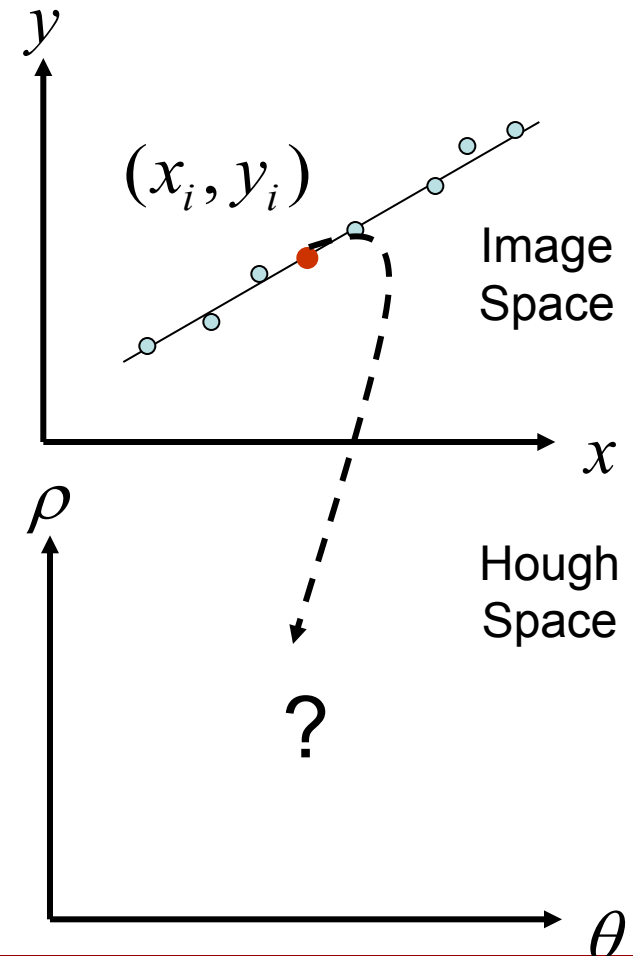
Line equation:  $\rho = -x \cos \theta + y \sin \theta$

Here  $0 \leq \theta \leq 2\pi$

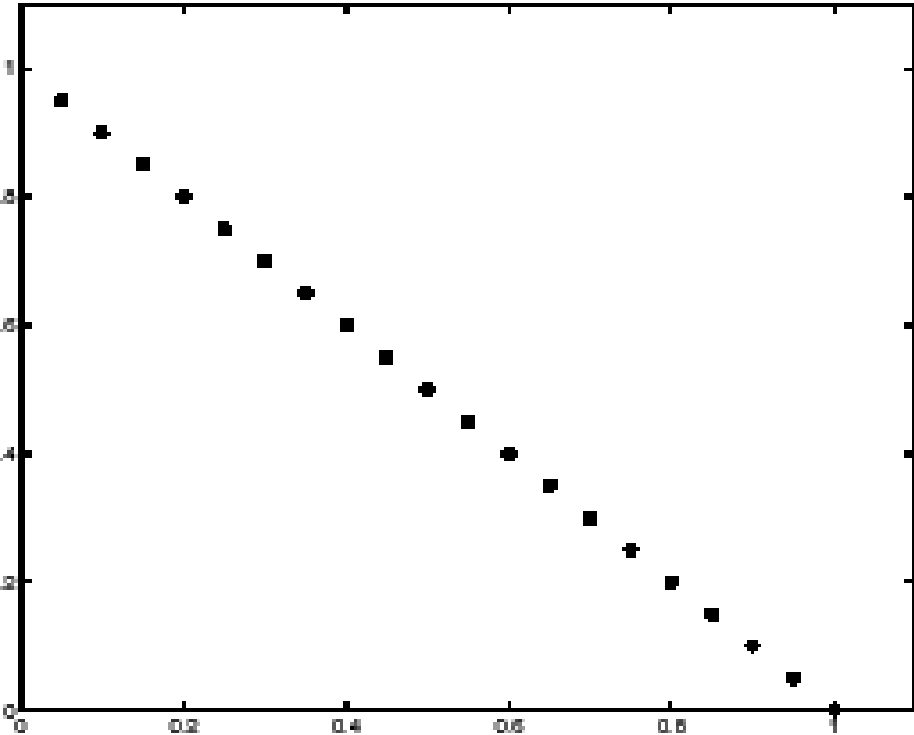
$0 \leq \rho \leq \rho_{\max}$

Given points  $(x_i, y_i)$  find  $(\rho, \theta)$

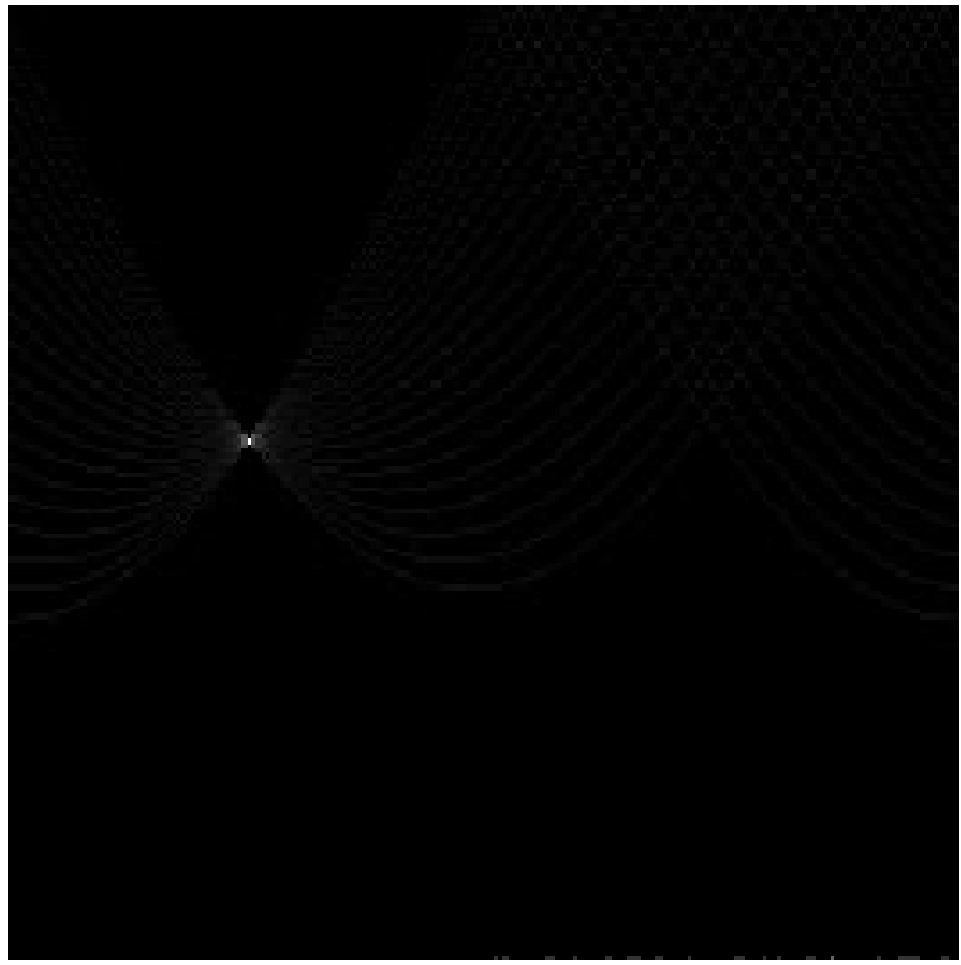
Hough Space Sinusoid





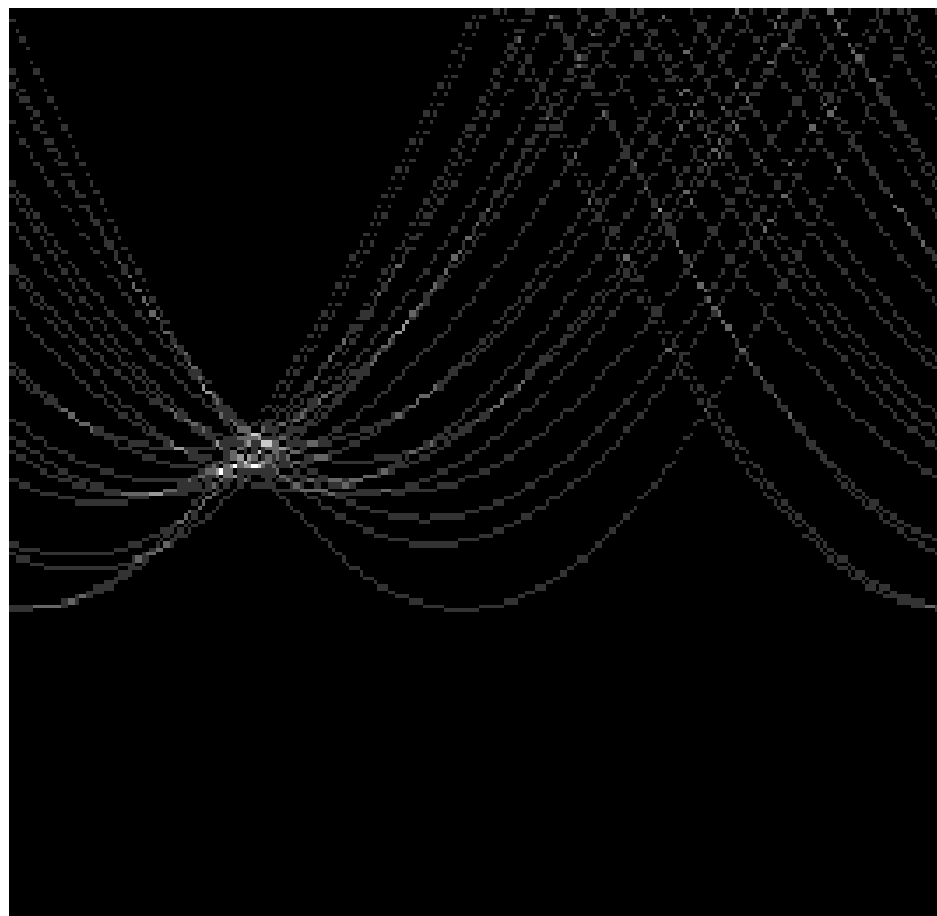
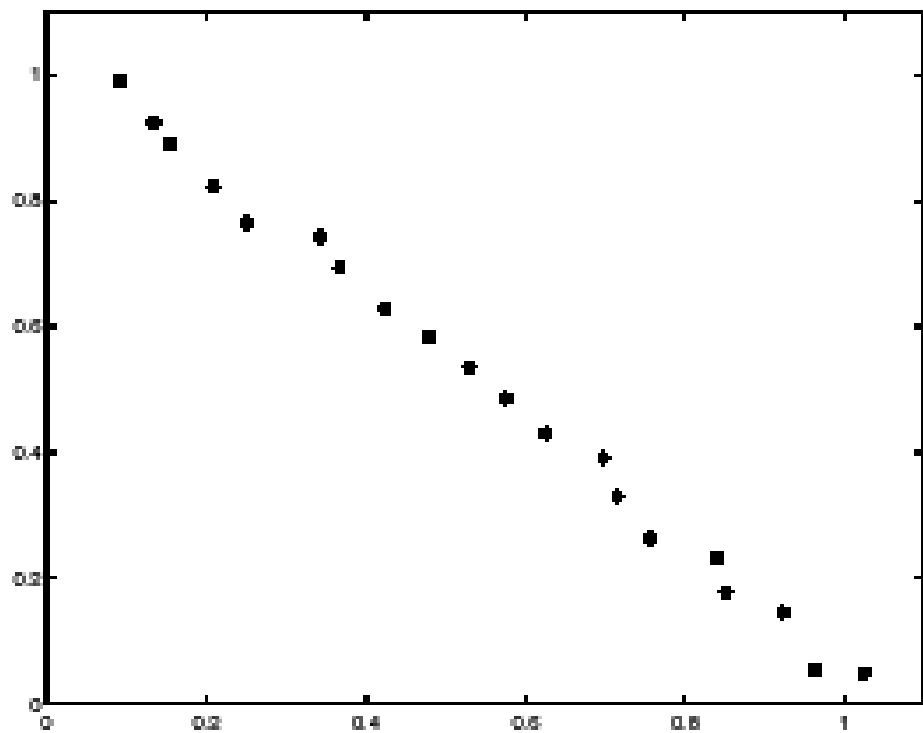


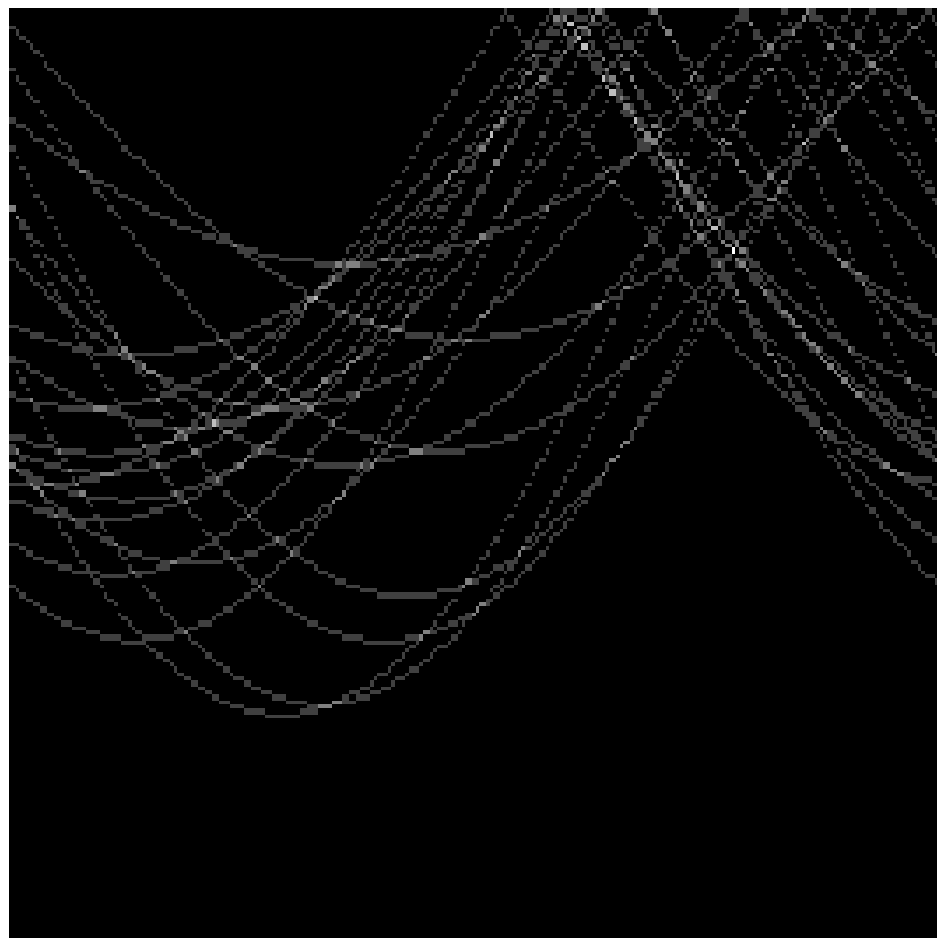
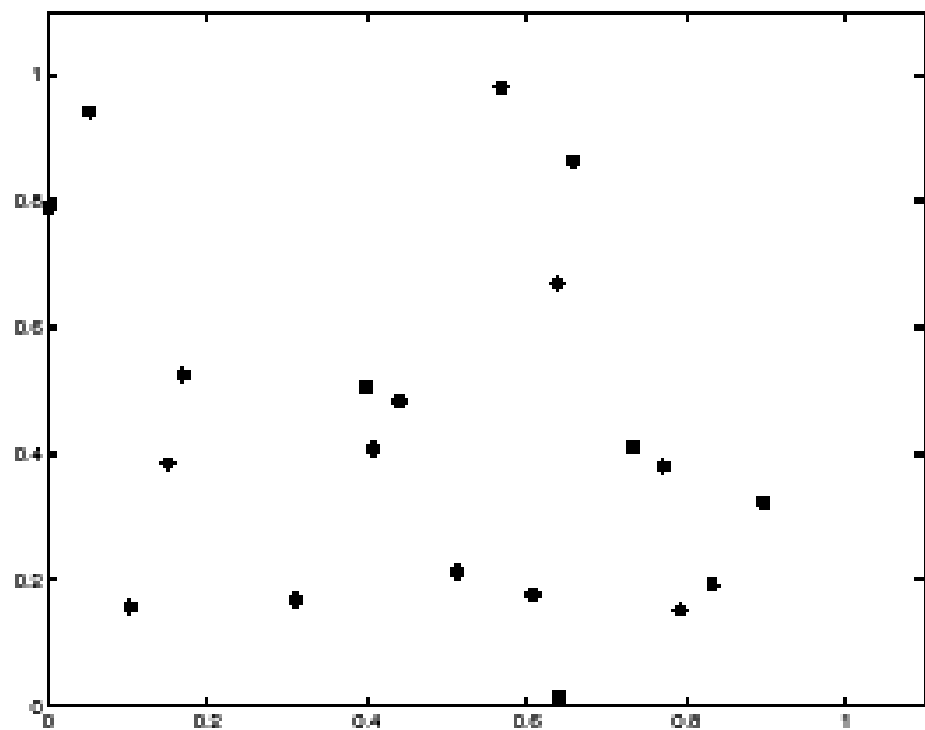
**Image space**



**Votes**

Horizontal axis is  $\theta$ ,  
vertical is  $\rho$ .





---

# Mechanics of the Hough Transform

---

- **Difficulties**

- how big should the cells be? (too big, and we merge quite different lines; too small, and noise causes lines to be missed)

- **How many lines?**

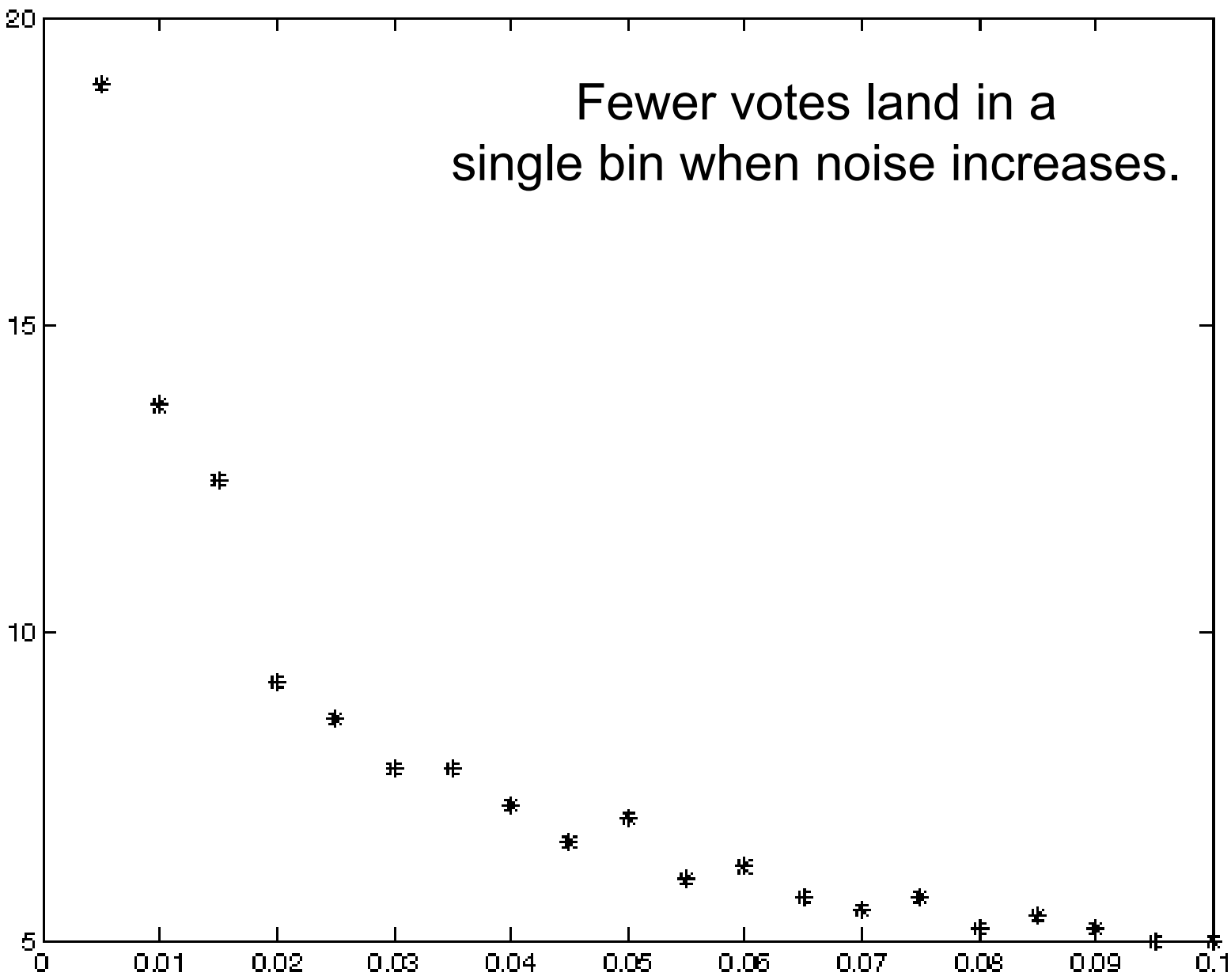
- Count the peaks in the Hough array
- Treat adjacent peaks as a single peak

- **Which points belong to each line?**

- Search for points close to the line
- Solve again for line and iterate

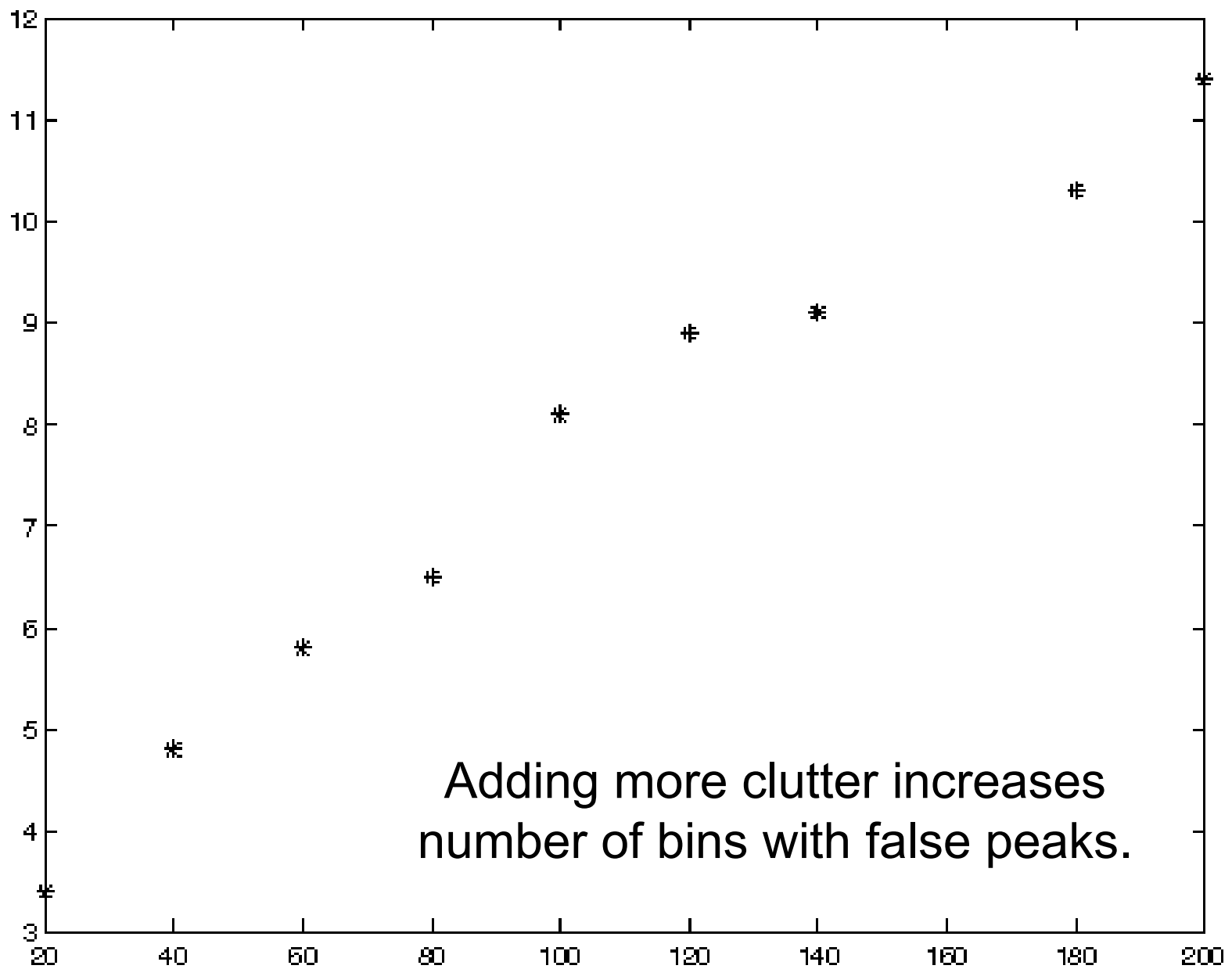
Fewer votes land in a single bin when noise increases.

Maximum number of votes



Noise level

Maximum number of votes



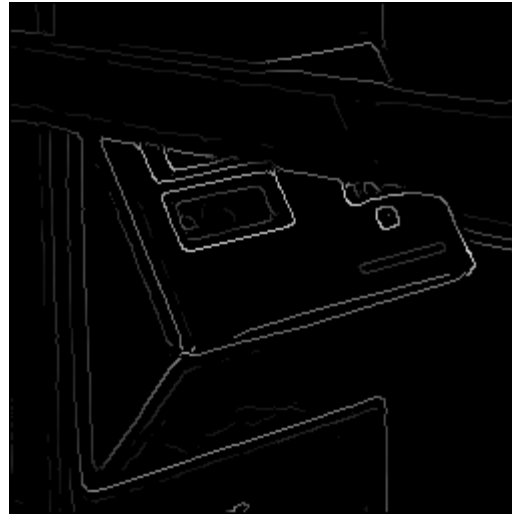
Adding more clutter increases number of bins with false peaks.

Number of noise points

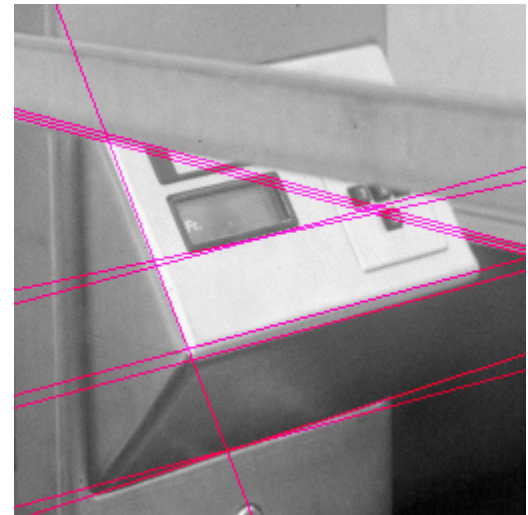
# Real World Example



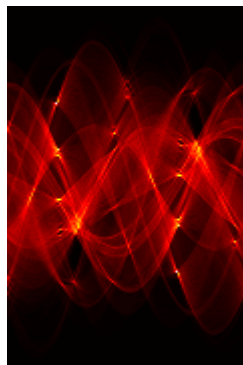
Original



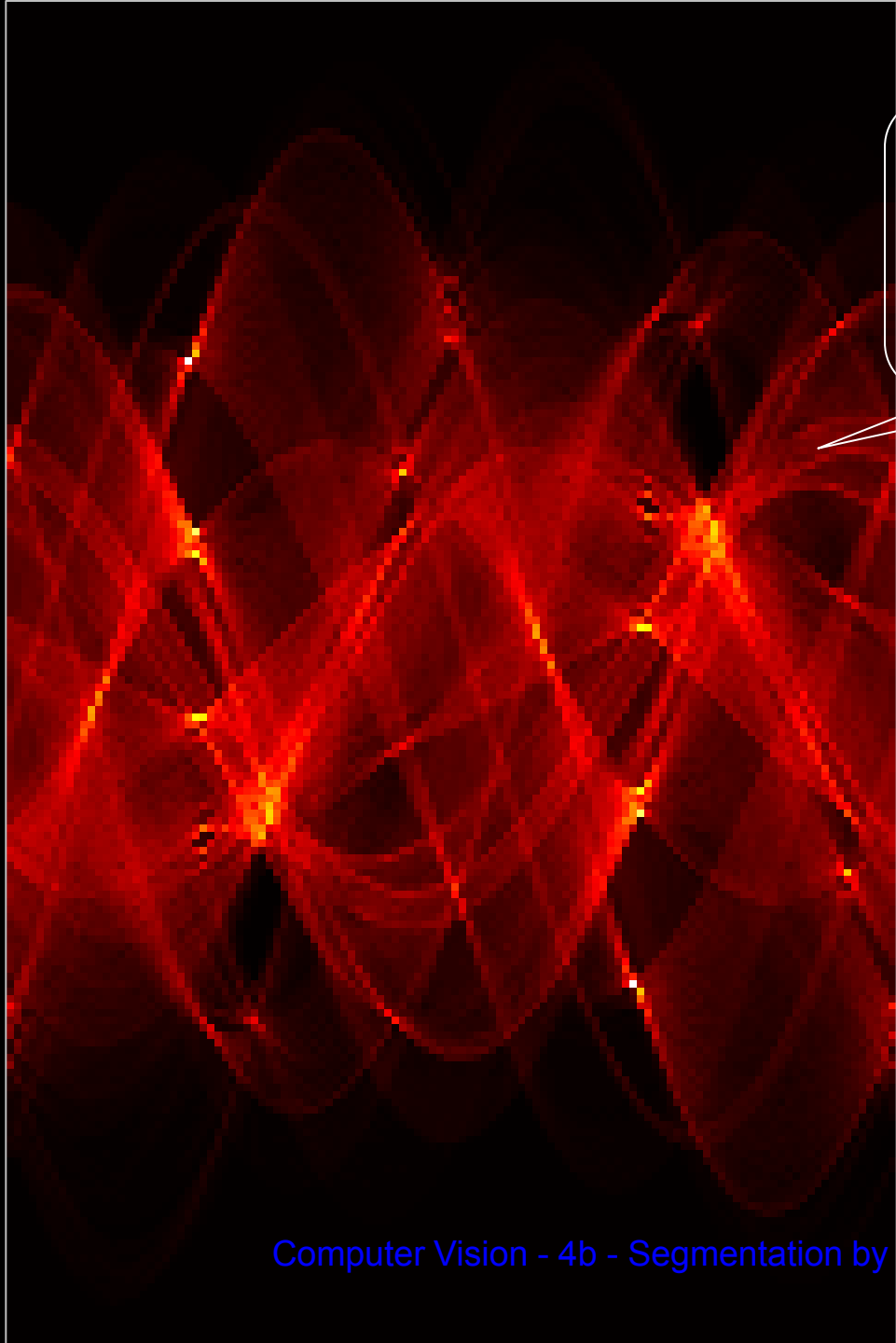
Edge  
Detection



Found Lines



Parameter Space

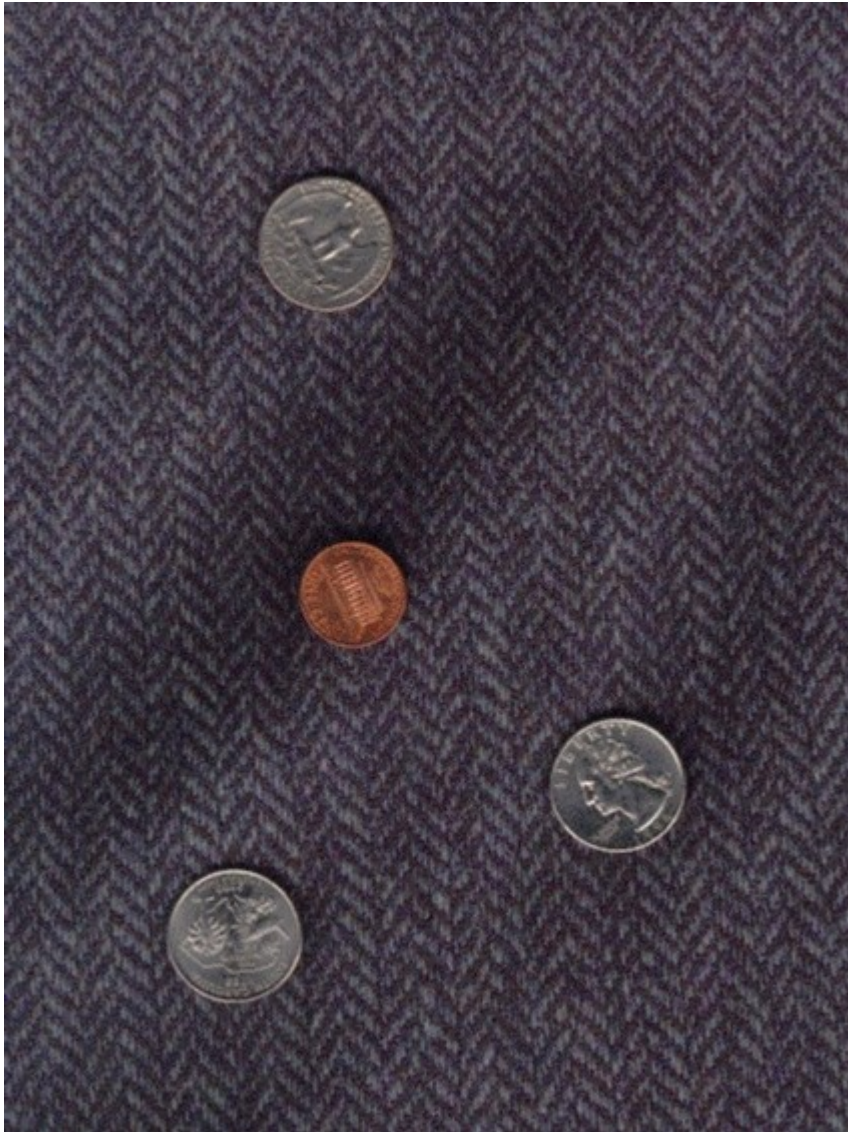


Where are the  
lines?

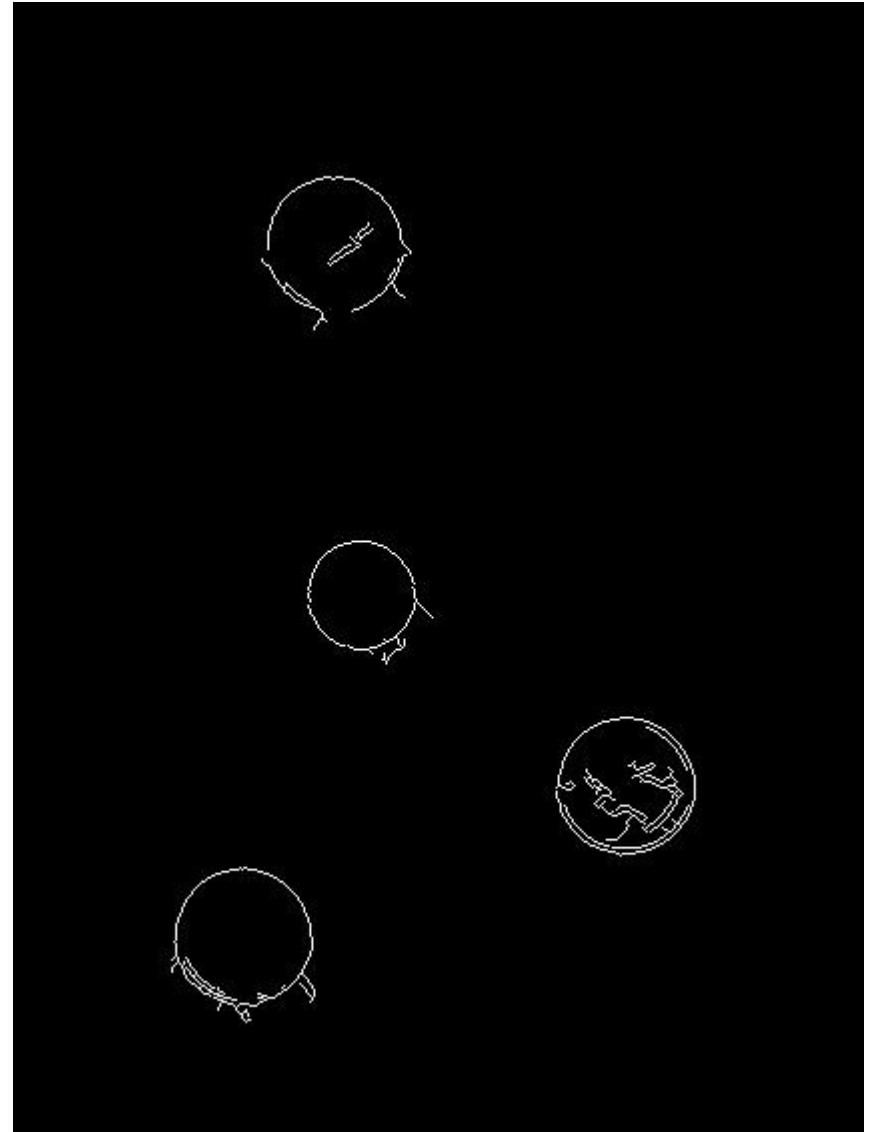


# Other shapes

Original



Edges when using circle model



---

# Topic: Fitting Lines

---

- The Hough Transform
- **Fitting Lines**
- Fitting Curves
- Fitting as a Probabilistic Inference Problem

---

# Least Squares

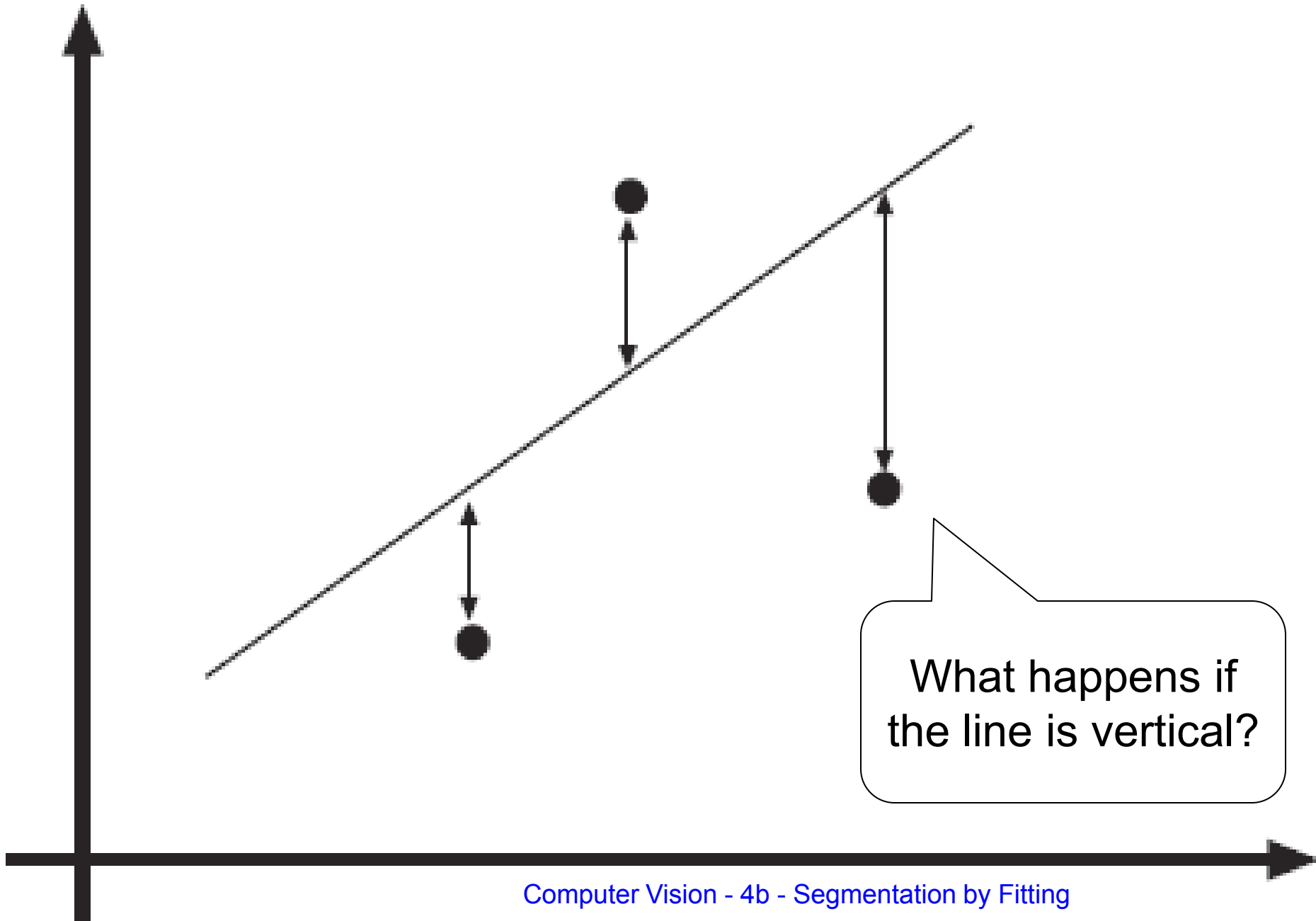
---

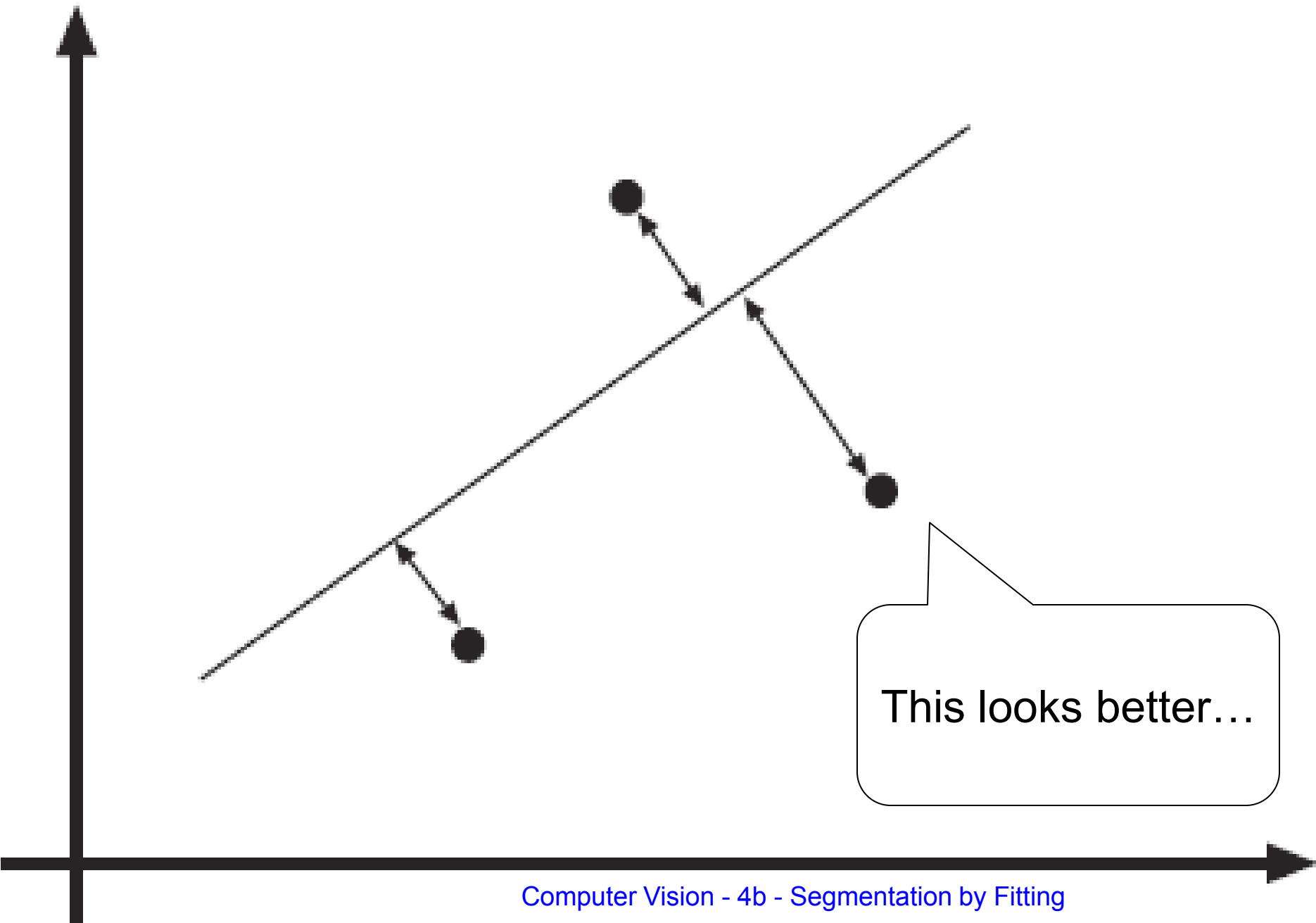
- Popular fitting procedure.
- Simple but biased (why?).
- Consider a line:

$$y = ax + b$$

- What is the line that best predicts all observations  $(x_i, y_i)$ ?

- Minimize:  $\sum_i (y_i - ax_i - b)^2$





This looks better...

# Total Least Squares

- Works with the actual distance between the point and the line (rather than the vertical distance).
- Lines are represented as a collection of points where:

$$ax + by + c = 0$$

- And:

$$a^2 + b^2 = 1$$

Again... Minimize the error, obtain the line with the 'best fit'.

---

# Point correspondence

---

- We can estimate a line but, **which points are on which line?**
- Usually:
  - We are fitting lines to edge points, so...
  - Edge directions can give us hints!
- What if I only have isolated points?
- Let's look at two options:
  - Incremental fitting.
  - Allocating points to lines with K-means

---

# Incremental Fitting

---

- Start with connected *curves* of edge points
- Fit *lines* to those points in that curve.
- Incremental fitting:
  - Start at one end of the *curve*.
  - Keep fitting all points in that curve to a line.
  - Begin another line when the fitting deteriorates too much.
- Great for closed curves!



```
Put all points on curve list, in order along the curve
empty the line point list
empty the line list
```

```
Until there are two few points on the curve
  Transfer first few points on the curve to the line point list
  fit line to line point list
```

```
while fitted line is good enough
  transfer the next point on the curve
  to the line point list and refit the line
end
```

```
transfer last point back to curve
attach line to line list
```

```
end
```

---

# K-means allocation

---

- What if points carry no hints about which line they lie on?
- Assume there are  $k$  lines for the  $x$  points.
- Minimize: 
$$\sum_{\text{lines}} \sum_{\text{points}} \text{dist}(\text{line}, \text{point})^2$$
- Iteration:
  - Allocate each point to the closest line.
  - Fit the best line to the points allocated to each line.

Hypothesize  $k$  lines (perhaps uniformly at random)  
*or*

hypothesize an assignment of lines to points  
and then fit lines using this assignment

Until convergence

    allocate each point to the closest line  
    refit lines

---

# Topic: Fitting Curves

---

- The Hough Transform
- Fitting Lines
- **Fitting Curves**
- Fitting as a Probabilistic Inference Problem

---

# Fitting Curves

---

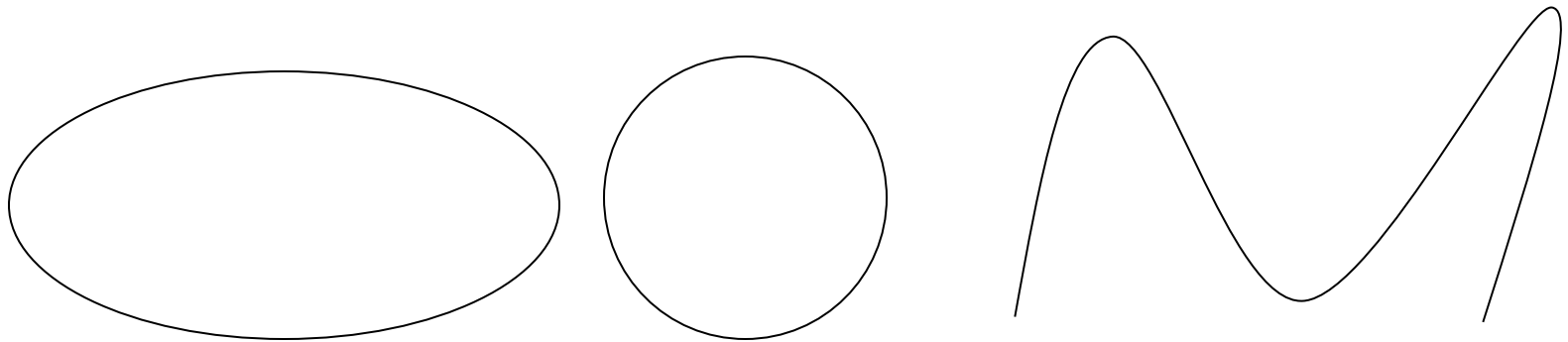
- In principle, **same as fitting lines.**
  - Minimize distances of points to the curve.
- However, how can I tell the **distance between a point and a curve?**
  - Get your geometry book and **solve it.**
  - Get your engineering book and **approximate it.**

---

# Implicit Curves

---

- Coordinates satisfy some parametric equation.
- If the equation is **polynomial** then the curve is said to be **algebraic**.



Curve	equation
Line	$ax + by + c = 0$
Circle, center (a, b) and radius r	$x^2 + y^2 - 2ax - 2by + a^2 + b^2 - r^2 = 0$
Ellipses (including circles)	$ax^2 + bxy + cy^2 + dx + ey + f = 0$ where $b^2 - 4ac < 0$
Hyperbolae	$ax^2 + bxy + cy^2 + dx + ey + f = 0$ where $b^2 - 4ac > 0$
Parabolae	$ax^2 + bxy + cy^2 + dx + ey + f = 0$ where $b^2 - 4ac = 0$
General conic sections	$ax^2 + bxy + cy^2 + dx + ey + f = 0$

---

# Distance to an Implicit Curve

---

- Distance between point  $(x, y)$  and **closest point** of the curve  $(u, v)$ .
- So, this distance vector:
  - Contains the point.
  - Is normal to the curve:  $\phi(u, v)$
- **Therefore:**
  - $\phi(u, v) = 0$  since  $(u, v)$  is on the curve
  - $s = (x, y) - (u, v)$  is normal to the curve



---

# Calculating the distance

---

- We can derive an equation to this distance but solving it is not always trivial.
- Even simple polynomial curves can make this problem rather daunting.
- Several ways to approximate distances.
- Want to dig deeper?
  - Forsyth and Ponce, Chapter 15.3.

# Parametric Curves

- Coordinates are given as **parametric functions** of a **parameter**.

Curves	Parametric form	parameters
Circles centered at the origin	$(r\sin(t), r\cos(t))$	$\theta = r$ $t \in [0, 2\pi)$
Circles	$(r\sin(t) + a, r\cos(t) + b)$	$\theta = (r, a, b)$ $t \in [0, 2\pi)$
Axis aligned ellipses	$(r_1\sin(t) + a, r_2\cos(t) + b)$	$\theta = (r_1, r_2, a, b)$ $t \in [0, 2\pi)$
Ellipses	$(\cos \phi (r_1\sin(t) + a) - \sin \phi (r_2\cos(t) + b),$ $\sin \phi (r_1\sin(t) + a) + \cos \phi (r_2\cos(t) + b))$	$\theta = (r_1, r_2, a, b, \phi)$ $t \in [0, 2\pi)$
cubic segments	$(at^3 + bt^2 + ct + d, et^3 + ft^2 + gt + h)$	$\theta = (a, b, c, d, e, f, g, h)$ $t \in [0, 1]$

---

# Topic: Fitting as a PIP

---

- The Hough Transform
- Fitting Lines
- Fitting Curves
- **Fitting as a Probabilistic Inference Problem**

---

# Error models

---

- Total least squares fitting seems ‘reasonable’ but it’s in fact... arbitrary.
  - It does not take into account the error model of our data.
  - We should thus be asking the following question:
- How did the points come to not lie on a line in the first place?

---

# Maximizing the likelihood

---

- Let's assume our errors are Gaussian noise perturbations to our line.

$$N(0, \sigma)$$

$$au + bv + c = 0, a^2 + b^2 = 1$$

$$P(\text{meas.} \mid a, b, c) = \prod_i P(x_i, y_i \mid a, b, c)$$

- We can thus maximize our log-likelihood:

$$-\frac{1}{2\sigma^2} \sum_i (ax_i + by_i + c)^2$$

---

# Difficulties

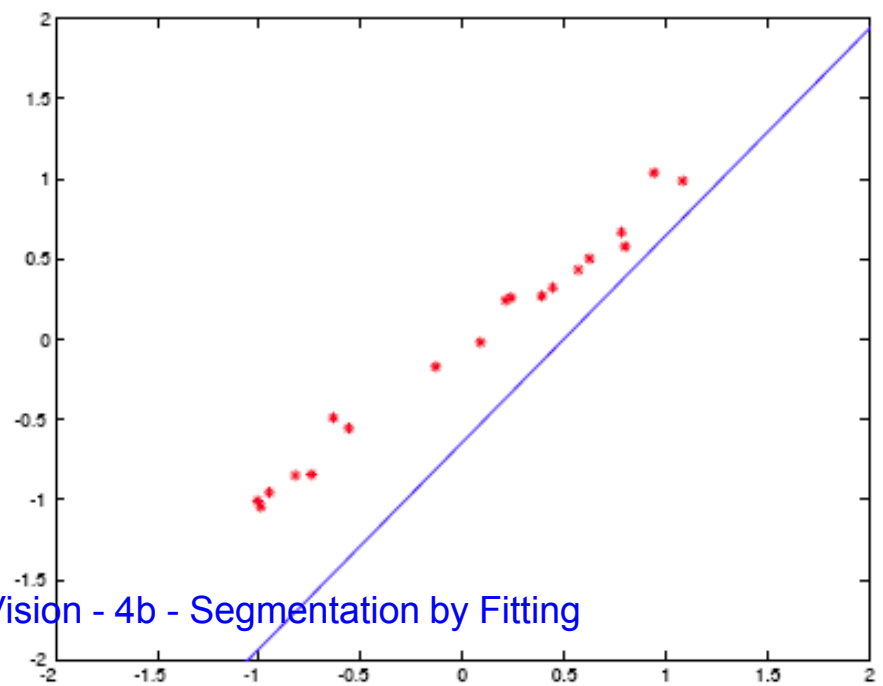
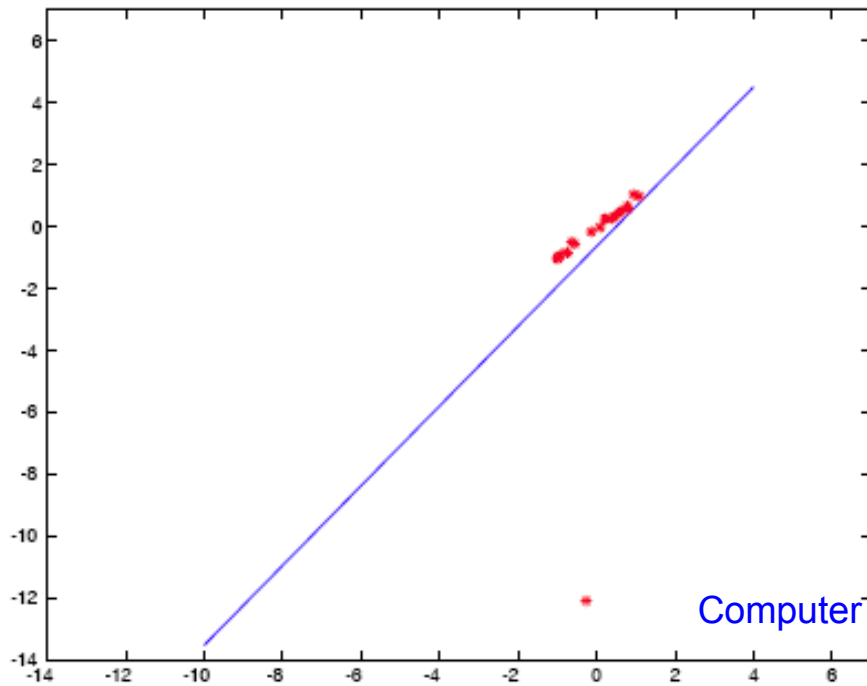
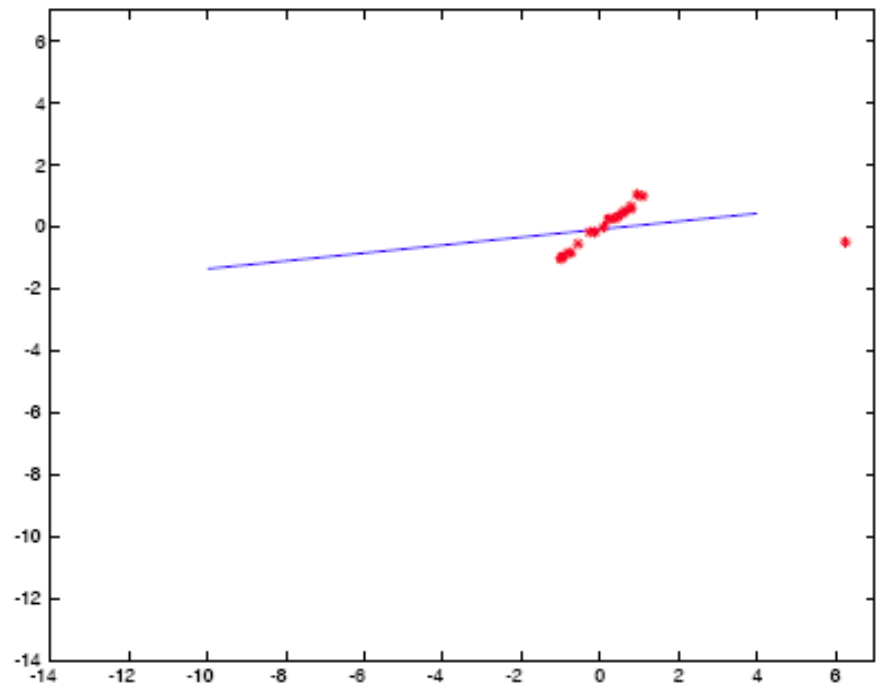
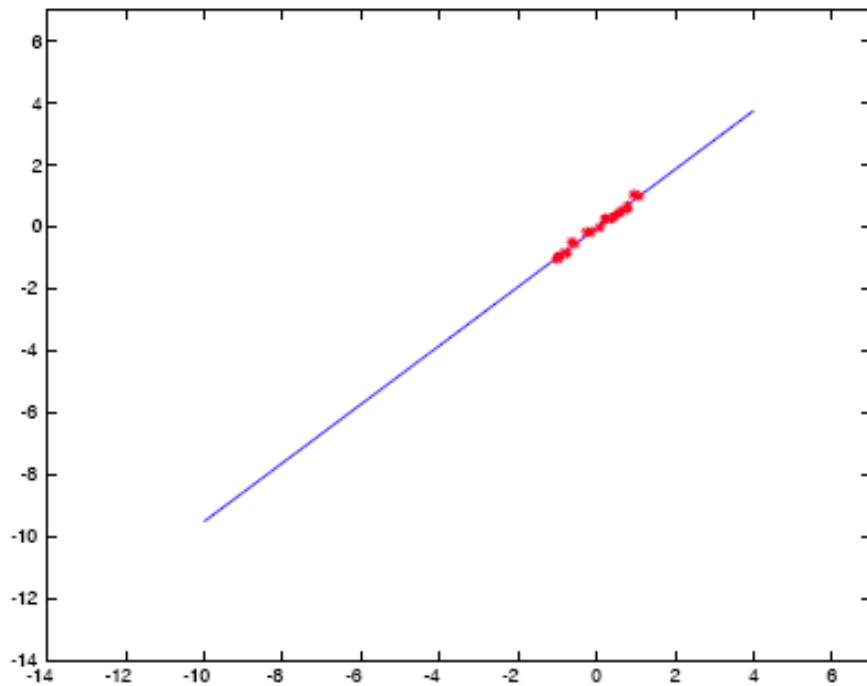
---

- **Robustness**

- TLS places a huge weight on large errors.
- Leads to severe fitting errors.
- Our solution must be **robust** to errors.

- **Missing data**

- Some points are noise.
- Some points come from real lines.
- Distinguishing them is vital for a good fitting!



Computer Vision - 4b - Segmentation by Fitting

---

# Robust fitting

---

- How do we minimize the effect of these **outliers**?
  - **M-estimators** - Give noise 'heavier tails'.
  - Study data points and identify outliers.
  - **RANSAC** – Search for points that 'appear to be good' and use those.
- **Want to know more?**
  - Forsyth and Ponce, Chapter 15 and 16.



---

# Resources

---

- Forsyth and Ponce, Chapter 15