Computer Vision

18AI742

Dr. rer. nat. D. Antony Louis Piriyakumar,
Dean (R&D),
Cambridge Institute of Technology.

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Grouping and Model Fitting

Dr. D. Antony Louis Piriyakumar,
Dean (Research & Development)
Registered Indian patent agent (IN/PA 3041)

www.cambridge.edu.in

Grouping and model fitting

Contents

$$\nabla \cdot \mathbf{E} = \rho/\varepsilon_0$$

$$\nabla \cdot \mathbf{B} = 0$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$$

$$\nabla \times \mathbf{B} = \mu_0 \varepsilon_0 \frac{\partial \mathbf{E}}{\partial t} + \mu_0 \boldsymbol{j}_c$$

where

$$\nabla = \hat{\mathbf{i}} \frac{\partial}{\partial x} + \hat{\mathbf{j}} \frac{\partial}{\partial y} + \hat{\mathbf{k}} \frac{\partial}{\partial z}$$

Grouping and model fitting

- To collect together pixels, tokens, or whatever conforming to some model
- Clustering methods and various ways of measuring similarity – local view
- Results - local structure, but will not necessarily have a global structure
- find all the lines represented by a set of tokens – fitting/grouping

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

The Hough transform

- grouping points that lie on lines
- Take each point and vote for all lines that could go through it
- Lines passing through many points and so have many votes
- Instead of x,y do it in m (slope), c (intercept)

The Hough transform

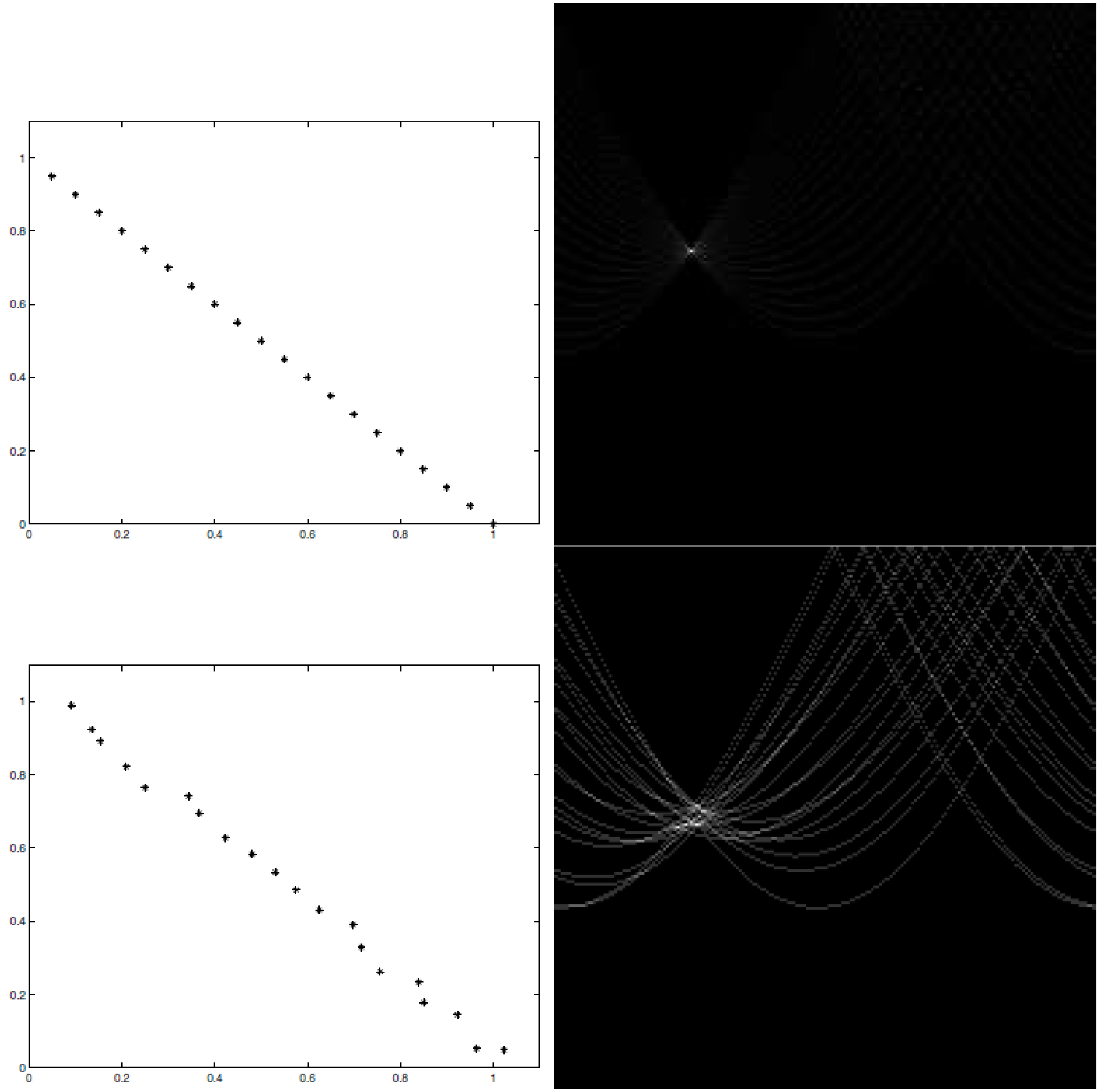
$$x \cos\theta + y \sin\theta + r = 0.$$

Now any pair of (θ, r) represents a unique line, where $r \geq 0$ is the perpendicular distance from the line to the origin and $0 \leq \theta < 2\pi$. We call the set of pairs (θ, r) *line space*; the space can be visualized as a half-infinite cylinder. There is a family

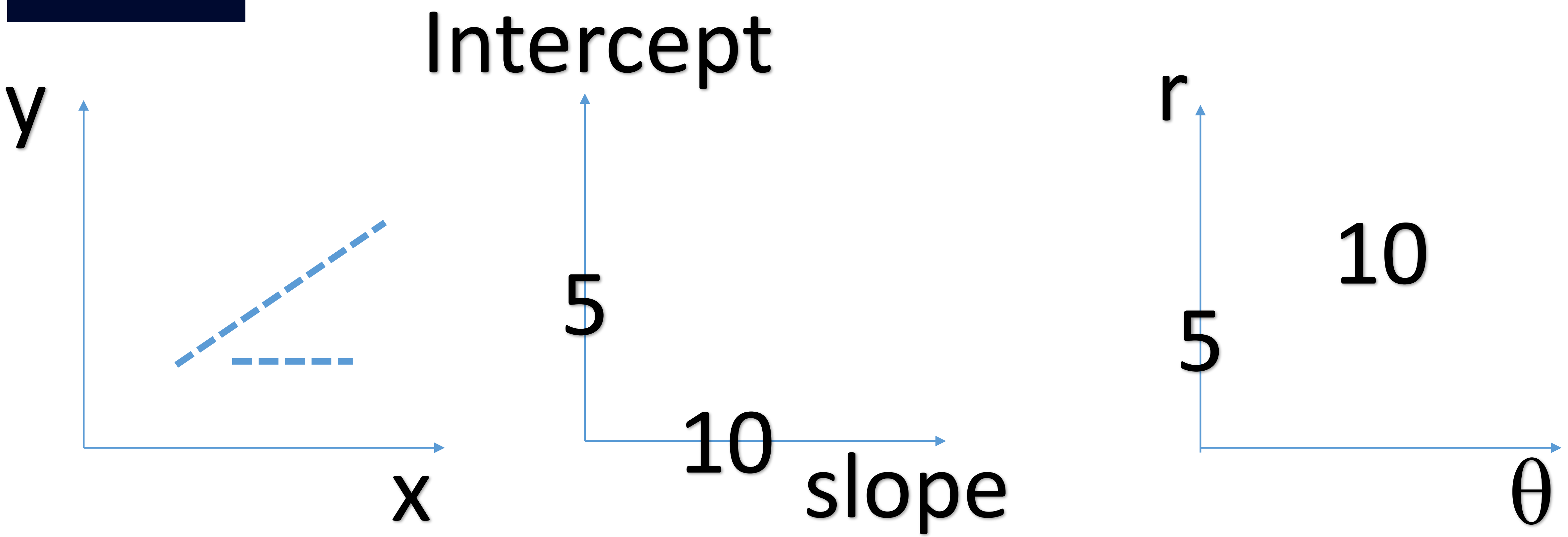
the curve *in line space* given by $r = -x_0 \cos\theta + y_0 \sin\theta$ all pass through the point token at (x_0, y_0).

Grid of buckets is referred to as the accumulator array.

The Hough transform



The Hough transform



CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Several sources of difficulty in using HT

- Grid dimension of accumulator array – 2 for lines, 3 for circle
- high-dimensional accumulator arrays, which take unmanageable amounts of storage.
- Quantization errors in appropriate grid size
- coarse a grid can lead to large values of the vote being obtained falsely
- Noise -connects widely separated tokens that lie close to some structure

Natural application of the Hough transform is in object recognition

- detect objects by first detecting parts
- allowing each detected part to vote for location
- find instances on which many part detectors agree
- parts might move around a bit – ok?

Grouping and model fitting

Contents

$$\nabla \cdot \mathbf{E} = \rho / \varepsilon_0$$

$$\nabla \cdot \mathbf{B} = 0$$

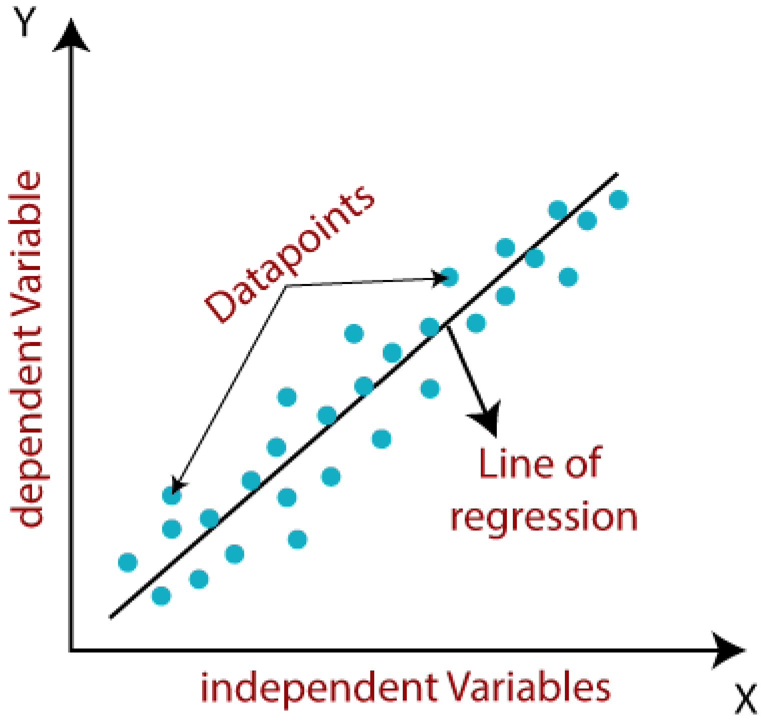
$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$$

$$\nabla \times \mathbf{B} = \mu_0 \varepsilon_0 \frac{\partial \mathbf{E}}{\partial t} + \mu_0 \boldsymbol{j}_c$$

where

$$\nabla = \hat{\mathbf{i}} \frac{\partial}{\partial x} + \hat{\mathbf{j}} \frac{\partial}{\partial y} + \hat{\mathbf{k}} \frac{\partial}{\partial z}$$

Fitting a single line

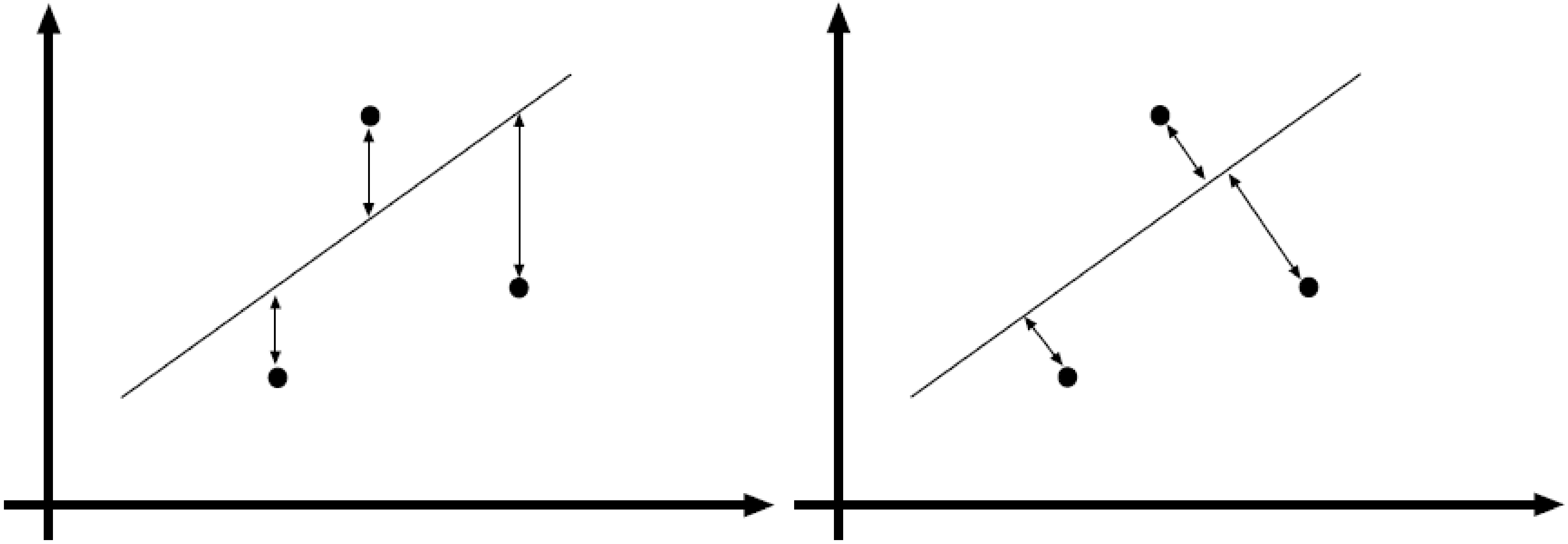


$$\sum_i (y_i - ax_i - b)^2.$$

By differentiation, the line is given by the solution to the problem

$$\begin{pmatrix} \overline{y^2} \\ \overline{y} \end{pmatrix} = \begin{pmatrix} \overline{x^2} & \overline{x} \\ \overline{x} & 1 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix}.$$

Total least squares



Total least squares

$$\sum_i (ax_i + by_i + c)^2,$$

subject to $a^2 + b^2 = 1$. Now using a Lagrange multiplier λ, we have a solution if

$$\begin{pmatrix} \overline{x^2} & \overline{xy} & \overline{x} \\ \overline{xy} & \overline{y^2} & \overline{y} \\ \overline{x} & \overline{y} & 1 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \lambda \begin{pmatrix} 2a \\ 2b \\ 0 \end{pmatrix}.$$

This means that

$$c = -a\overline{x} - b\overline{y},$$

and we can substitute this back to get the eigenvalue problem

$$\begin{pmatrix} \overline{x^2} - \overline{x}\,\overline{x} & \overline{xy} - \overline{x}\,\overline{y} \\ \overline{xy} - \overline{x}\,\overline{y} & \overline{y^2} - \overline{y}\,\overline{y} \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \mu \begin{pmatrix} a \\ b \end{pmatrix}.$$

Fitting Multiple Lines

```
Put all points on curve list, in order along the curve
Empty the line point list
Empty the line list
Until there are too 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(s) back to curve
   Refit line
   Attach line to line list
end
```

Algorithm 10.1: Incremental Line Fitting.

K-means Line Fitting

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
end

Algorithm 10.2: K-means Line Fitting.

Grouping and model fitting

Contents

$$\nabla \cdot \mathbf{E} = \rho/\varepsilon_0$$

$$\nabla \cdot \mathbf{B} = 0$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$$

$$\nabla \times \mathbf{B} = \mu_0 \varepsilon_0 \frac{\partial \mathbf{E}}{\partial t} + \mu_0 \boldsymbol{j}_c$$

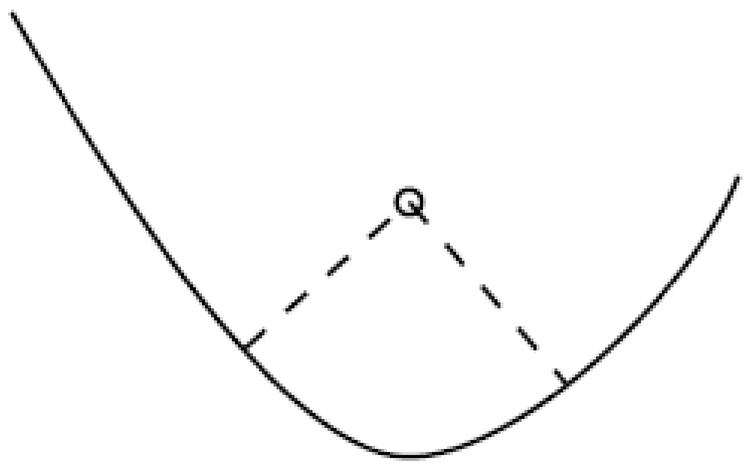
where

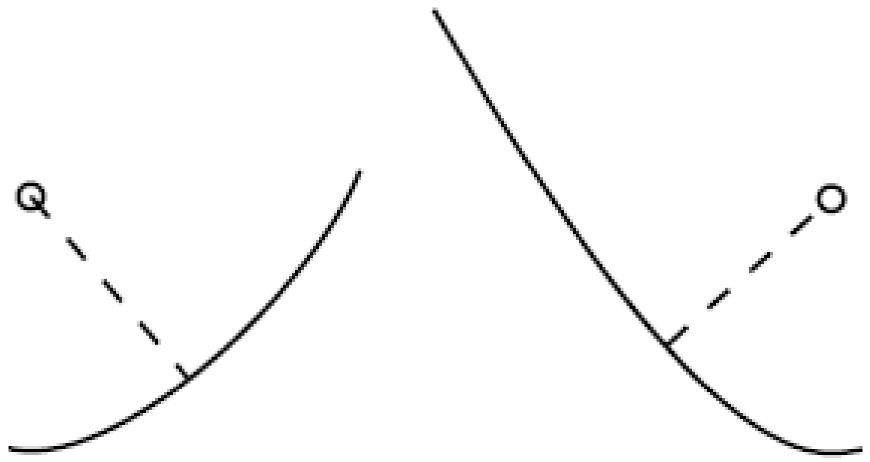
$$\nabla = \hat{\mathbf{i}}\frac{\partial}{\partial x} + \hat{\mathbf{j}}\frac{\partial}{\partial y} + \hat{\mathbf{k}}\frac{\partial}{\partial z}$$

Fitting curved structures

- Assume that the curve is implicit, and so has the form $\phi(x, y) = 0$.
- (u, v) is a point on the curve, which means that $\phi(u, v) = 0$.
- $s = (dx, dy) - (u, v)$ is normal to the curve.
- Given all s, length of the shortest is the distance from the data point to the curve.

There can be more than one point on a curve that looks locally as if it is closest to a token.





s = (dx, dy)−(u, v) is normal to the curve

normal at a point (u, v) is

$$\left(\frac{\partial \phi}{\partial x}, \frac{\partial \phi}{\partial y}\right),$$

evaluated at (u, v). If the tangent to the curve is $\boldsymbol{T}$, then we must have $\boldsymbol{T}.\boldsymbol{s} = 0$. Because we are working in 2D, we can determine the tangent from the normal, so that we must have

$$\psi(u, v; d_x, d_y) = \frac{\partial \phi}{\partial y}(u, v)\left\{d_x - u\right\} - \frac{\partial \phi}{\partial x}(u, v)\left\{d_y - v\right\} = 0$$

s = (dx, dy)–(u, v) is normal to the curve

the curve. This means that $s(\tau) = (d_x, d_y) - (x(\tau), y(\tau))$ is normal to the tangent vector, so that $s(\tau).\boldsymbol{T} = 0$. The tangent vector is

$$\left(\frac{dx}{dt}(\tau), \frac{dy}{dt}(\tau)\right),$$

which means that τ must satisfy the equation

$$\frac{dx}{dt}(\tau)\left\{d_x - x(\tau)\right\} + \frac{dy}{dt}(\tau)\left\{d_y - y(\tau)\right\} = 0.$$

Grouping and model fitting

Contents

$$\nabla \cdot \mathbf{E} = \rho / \varepsilon_0$$

$$\nabla \cdot \mathbf{B} = 0$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$$

$$\nabla \times \mathbf{B} = \mu_0 \varepsilon_0 \frac{\partial \mathbf{E}}{\partial t} + \mu_0 j_c$$

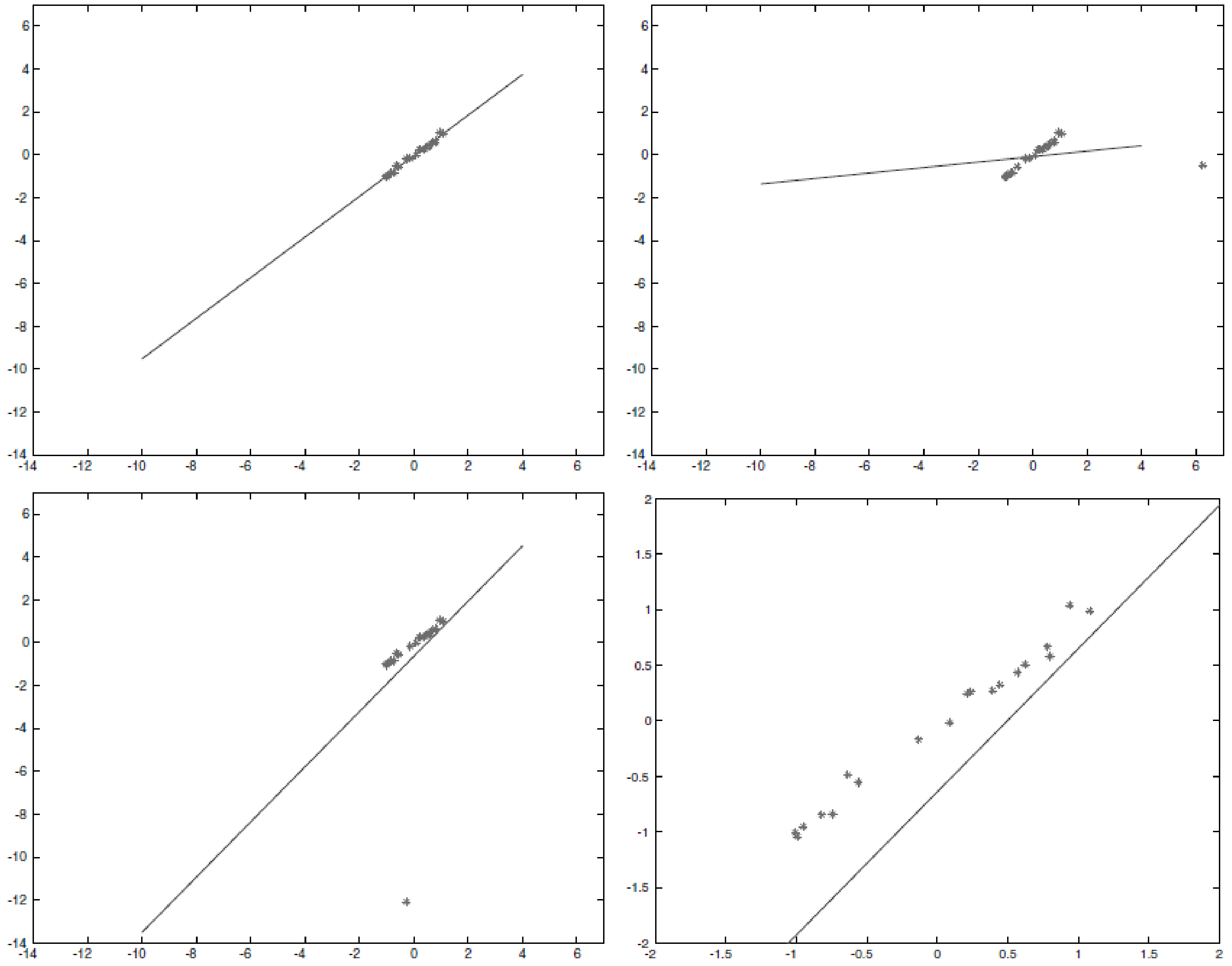
where

$$\nabla = \hat{\mathbf{i}} \frac{\partial}{\partial x} + \hat{\mathbf{j}} \frac{\partial}{\partial y} + \hat{\mathbf{k}} \frac{\partial}{\partial z}$$

Robustness

- single wildly inappropriate data point might give errors that dominate
- these errors could mresult in a substantial bias in the fitting process
- difficult to avoid such data points—usually called outliers
- Practical vision problems usually involve outliers.

Line fitting with a squared error is extremely sensitive to outliers



M-estimator estimates parameters by replacing the squared error term with

$$\sum_i \rho(r_i(\boldsymbol{x}_i, \theta); \sigma),$$

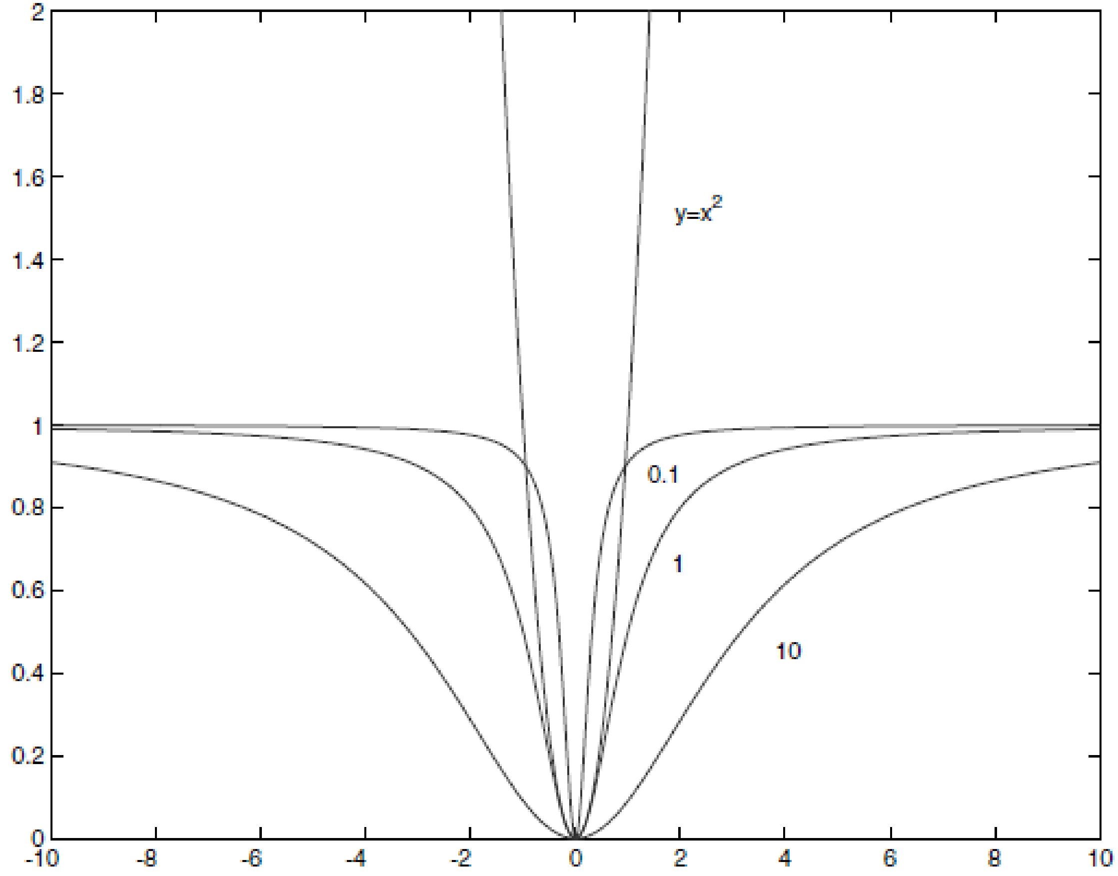
where θ are the parameters of the model being fitted (for example, in the case of the line, we might have the orientation and the y intercept), and $r_i(\boldsymbol{x}_i, \theta)$ is the residual error of the model on the ith data point. Using this notation, our least

$\rho(u; \sigma)$ look like u^2 for part of its range and then flattens out; we expect that $\rho(u; \sigma)$ increases monotonically, and is close to a constant value for large u. A common choice is

$$\rho(u; \sigma) = \frac{u^2}{\sigma^2 + u^2}.$$

The parameter σ controls the point at which the function flattens out, and we have

The function $\rho(x; \sigma) = x^2/(\sigma^2+x^2)$, plotted for $\sigma^2 = 0.1, 1,$ and 10, with a plot of $y = x^2$ for comparison



M-estimators

$$\rho(u;\sigma) = \frac{u^2}{\sigma^2 + u^2}.$$

available. Typically, they are discussed in terms of their *influence function*, which is defined as

$$\frac{\partial \rho}{\partial \theta}.$$

This is natural because our minimization criterion yields

$$\sum_i \rho(r_i(x_i, \theta); \sigma) \frac{\partial \rho}{\partial \theta} = 0$$

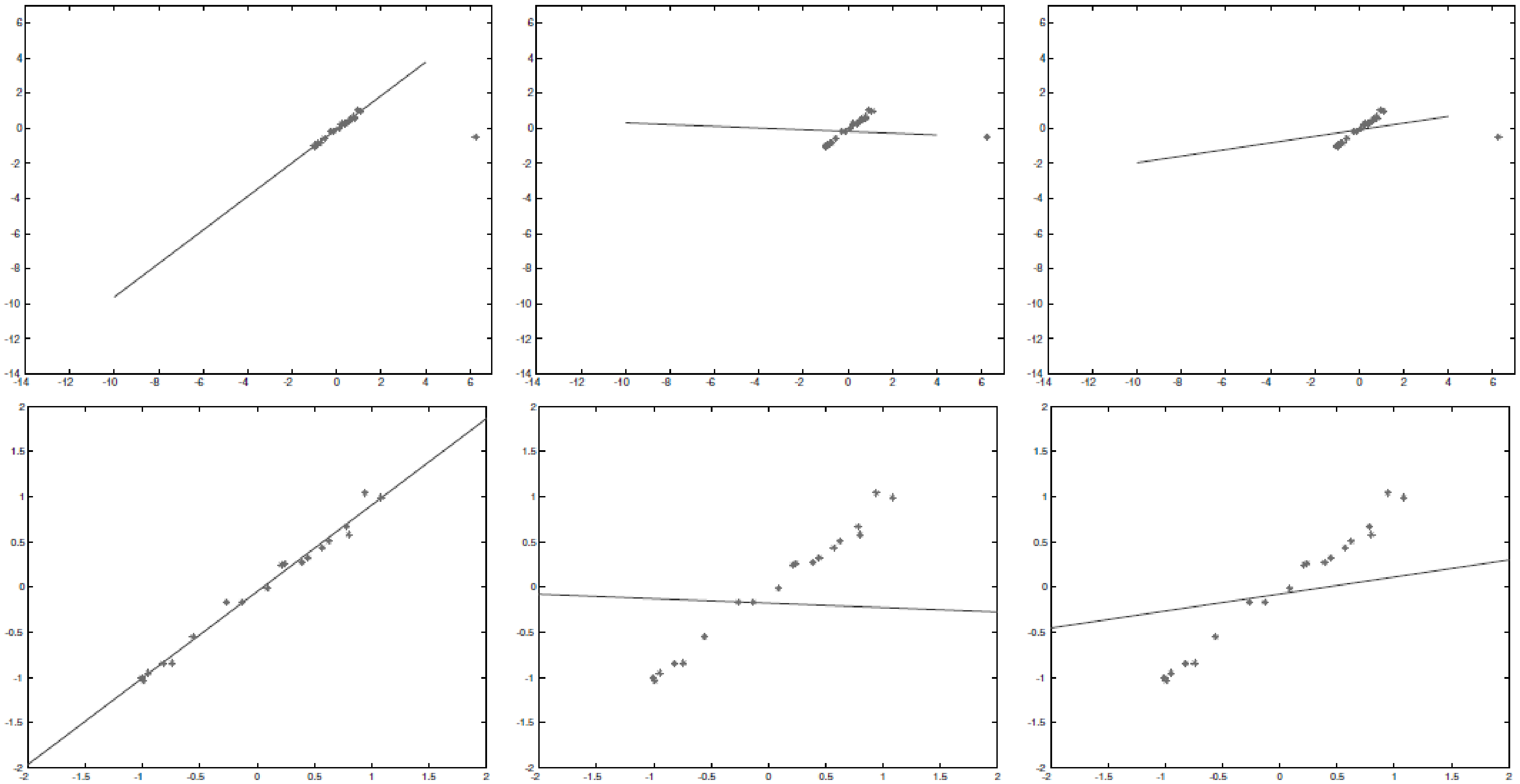
$$\sigma^{(n)} = 1.4826 \text{ median}_i \left| r_i^{(n)}(x_i; \theta^{(n-1)}) \right| .$$

Using an M-Estimator to Fit a Least Squares Model

For $s = 1$ to $s = k$
  Draw a subset of r distinct points, chosen uniformly at random
  Fit to this set of points using least squares to obtain an initial
    set of parameters θ_s^0
  Estimate σ_s^0 using θ_s^0
  Until convergence (usually $|\theta_s^n - \theta_s^{n-1}|$ is small):
    Take a minimizing step using θ_s^{n-1}, σ_s^{n-1}
      to get θ_s^n
    Now compute σ_s^n
  end
end
Report the best fit of this set of k trials, using the median of the residuals
  as a criterion

Algorithm 10.3: Using an M-Estimator to Fit a Least Squares Model.

Weighting function that deemphasizes the contribution of distant points



RANSAC: Searching for Good Points

- Search the collection of data points for good points.
- Choose a small subset of points and fit to that subset
- see how many other points fit to the resulting object
- continue this process until we have a high probability of finding the structure
- RANdom SAmple Consensus

RANSAC: Fitting Structures Using Random Sample (

Determine:

n—the smallest number of points required (e.g., for lines, $n = 2$, for circles, $n = 3$)
k—the number of iterations required
t—the threshold used to identify a point that fits well
d—the number of nearby points required to assert a model fits well

Until k iterations have occurred
    Draw a sample of n points from the data uniformly and at random
    Fit to that set of n points
    For each data point outside the sample
        Test the distance from the point to the structure against t; if the distance from the point to the structure is less than t, the point is close
    end
    If there are d or more points close to the structure then there is a good fit. Refit the structure using all these points. Add the result to a collection of good fits.
end
Use the best fit from this collection, using the fitting error as a criterion

The Number of Samples Required

that we need to draw n data points, and that w is the fraction of these points that are good (we need only a reasonable estimate of this number). Now the expected value of the number of draws k required to get one point is given by

$$\begin{aligned}
\mathrm{E}[k] &= 1P(\text{one good sample in one draw}) + \\
&\quad 2P(\text{one good sample in two draws}) + \ldots \\
&= w^n + 2(1-w^n)w^n + 3(1-w^n)^2 w^n + \ldots \\
&= w^{-n}
\end{aligned}$$

The Number of Samples Required

number. The standard deviation of k can be obtained as

$$SD(k) = \frac{\sqrt{1 - w^n}}{w^n}.$$

An alternative approach to this problem is to look at a number of samples that guarantees a low probability z of seeing only bad samples. In this case, we have

$$(1 - w^n)^k = z,$$

which means that

$$k = \frac{\log(z)}{\log(1 - w^n)}.$$

Grouping and model fitting

Contents

$$\nabla \cdot \mathbf{E} = \rho / \varepsilon_0$$

$$\nabla \cdot \mathbf{B} = 0$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$$

$$\nabla \times \mathbf{B} = \mu_0 \varepsilon_0 \frac{\partial \mathbf{E}}{\partial t} + \mu_0 \boldsymbol{j}_c$$

where

$$\nabla = \hat{\mathbf{i}} \frac{\partial}{\partial x} + \hat{\mathbf{j}} \frac{\partial}{\partial y} + \hat{\mathbf{k}} \frac{\partial}{\partial z}$$

Fitting using probabilistic models

- Key is to view observed data as having been produced by a generative model
- Straightforward to build probabilistic models from the fitting procedures
- Estimate the unknown parameters of the model in a straightforward way
- Generative model specifies how each data point was produced

CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Line fitting with least squares

- Producing data, x coordinate is uniformly distributed
- y coordinate is generated by (a) finding the point $ax_i + b$ on the line
- (b) adding a zero mean normally distributed random variable
- our observed data as having been produced by a generative model

Line fitting with least squares

random variable. Now write $x \sim p$ to mean that x is a sample from the probability distribution p; write $U(R)$ for the uniform distribution over some range of values R; and write $N(\mu, \sigma^2)$ for the normal distribution with mean mu and variance σ^2. With our notation, we can write:

$$
\begin{aligned}
x_i &\sim U(R) \\
y_i &\sim N(ax_i + b, \sigma^2).
\end{aligned}
$$

Line fitting with least squares

usual way to estimate parameters in a probabilistic model is to maximize the likelihood of the data, typically by working with the negative log-likelihood and minimizing that. In this case, the log-likelihood of the data is

$$
\begin{aligned}
\mathcal{L}(a, b, \sigma) &= \sum_{i \in \text{data}} \log P(x_i, y_i | a, b, \sigma) \\
&= \sum_{i \in \text{data}} \log P(y_i | x_i, a, b, \sigma) + \log P(x_i) \\
&= \sum_{i \in \text{data}} -\frac{(y_i - (ax_i + b))^2}{2\sigma^2} - \frac{1}{2} \log 2\pi\sigma^2 + K_b
\end{aligned}
$$

where K_b is a constant representing $\log P(x_i)$. Now, to minimize the negative log-likelihood as a function of a and b we could minimize $\sum_{i \in \text{data}} (y_i - (ax_i + b))^2$ as a function of a and b (which is what we did for least-squares line fitting in

Total least-squares line fitting

case, to generate a data point (x_i, y_i), we generate a point (u_i, v_i) uniformly at random along the line (or rather, along a finite length segment of the line likely to be of interest to us), then sample a distance ξ_i (where $\xi_i \sim N(0, \sigma^2)$, and move the point (u_i, v_i) perpendicular to the line by that distance. If the line is $ax + by + c = 0$ and if $a^2 + b^2 = 1$, we have that $(x_i, y_i) = (u_i, v_i) + \xi_i(a, b)$. We can write the

Total least-squares line fitting

log-likelihood of the data under this model as

$$\begin{aligned}\mathcal{L}(a,b,c,\sigma) &= \sum_{i\in\text{data}} \log P(x_i, y_i|a,b,c,\sigma) \\ &= \sum_{i\in\text{data}} \log P(\xi_i|\sigma) + \log P(u_i, v_i|a,b,c).\end{aligned}$$

But $P(u_i, v_i|a,b,c)$ is some constant, because this point is distributed uniformly along the line. Since ξ_i is the perpendicular distance from (x_i, y_i) to the line (which is $\|(ax_i + by_i + c)\|$ as long as $a^2 + b^2 = 1$), we must maximize

$$\begin{aligned}\sum_{i\in\text{data}} \log P(\xi_i|\sigma) &= \sum_{i\in\text{data}} -\frac{\xi_i^2}{2\sigma^2} - \frac{1}{2}\log 2\pi\sigma^2 \\ &= \sum_{i\in\text{data}} -\frac{(ax_i + by_i + c)^2}{2\sigma^2} - \frac{1}{2}\log 2\pi\sigma^2\end{aligned}$$

(again, subject to $a^2 + b^2 = 1$). For fixed (but perhaps unknown) σ this yields the problem we were working with in Section 10.2.1. So far, generative models have

Missing data problems

- a statistical problem where some data is missing
- some terms in a data vector are missing for some instances and present for others
- by rewriting it using some variables whose values are unknown
- take an expectation over the missing data

Example: Outliers and Line Fitting

- to fit a line to a set of tokens that are at $x_i = (x_i, y_i)$. P(token comes from line) = α.
- $P(x_i | a, b, c, \alpha) = P(x_i, \text{line} | a, b, c, \alpha) + P(x_i, \text{outlier} | a, b, c, \alpha)$
- $= P(x_i | \text{line}, a, b, c)P(\text{line}) + P(x_i | \text{outlier}, a, b, c)P(\text{outlier})$
- $= P(x_i | \text{line}, a, b, c)\, \alpha + P(x_i | \text{outlier}, a, b, c)(1 - \alpha)$.

Example: Image segmentation

- At each pixel in an image, we compute a d-dimensional feature vector x
- Let image contains g segments, and each pixel is produced by one of these segments
- lth segment is chosen with probability αl, known covariance Σ and unknown mean θl = (μl)
- probability of generating a pixel vector x as p(x|Θ) = $\sum_i p(\boldsymbol{x}|\theta_l)$ αl.

Mixture Models and Hidden Variables

$$\delta_{ij} = \begin{cases} 1 & \text{if item } i \text{ came from component } j \\ 0 & \text{otherwise} \end{cases}$$

the *complete data log-likelihood*,

$$\mathcal{L}_c(\Theta) = \sum_{i \in \text{observations}} \log P(\boldsymbol{x}_i, \delta_i | \Theta),$$

$$\begin{aligned}
\mathcal{L}_c(\Theta) &= \sum_{i \in \text{observations}} \log P(\boldsymbol{x}_i, \delta_i | \Theta) \\
&= \sum_{i \in \text{observations}} \log \prod_{j \in \text{components}} [p_j(\boldsymbol{x}_i | \theta_j) \pi_j]^{\delta_{ij}} \\
&= \sum_{i \in \text{observations}} \left(\sum_{j \in \text{components}} [(\log p_j(\boldsymbol{x}_i | \theta_j) \log \pi_j) \, \delta_{ij}] \right)
\end{aligned}$$

EM Algorithm for Mixture Models

1. Obtain some estimate of the missing data using a guess at the parameters.
2. Form a maximum likelihood estimate of the free parameters using the estimate of the missing data.

We would iterate this procedure until (hopefully!) it converged. In the case of line fitting, the algorithm would look like this:

1. Obtain some estimate of which points lie on the line and which are off lines, using an estimate of the line.
2. Form a revised estimate of the line, using this information.

For image segmentation, this would look like the following:

1. Obtain some estimate of the component from which each pixel's feature vector came, using an estimate of the θ_l.
2. Update the θ_l and the mixing weights, using this estimate.

Expectation-maximization (EM) algorithm

formally, given $\Theta^{(s)}$, we form $\Theta^{(s+1)}$ by:

1. Computing an expected value for the *complete* data log-likelihood using the incomplete data and the current value of the parameters. That is, we compute

$$Q(\Theta; \Theta^{(s)}) = E_{\delta|\boldsymbol{x},\Theta^{(s)}} \mathcal{L}_c(\Theta).$$

   Notice that this object is a *function* of Θ, obtained by taking an expectation of a function of Θ and δ; the expectation is with respect to $P(\delta|\boldsymbol{x}, \Theta^{(s)})$. This is referred to as the *E-step*.

2. Maximizing this object as a function of Θ. That is, we compute

$$\Theta^{(s+1)} = \arg\max_{\Theta} Q(\Theta; \Theta^{(s)}).$$

   This is known as the *M-step*.

Equivalent to allocating each data point to the j'th model with weight αij

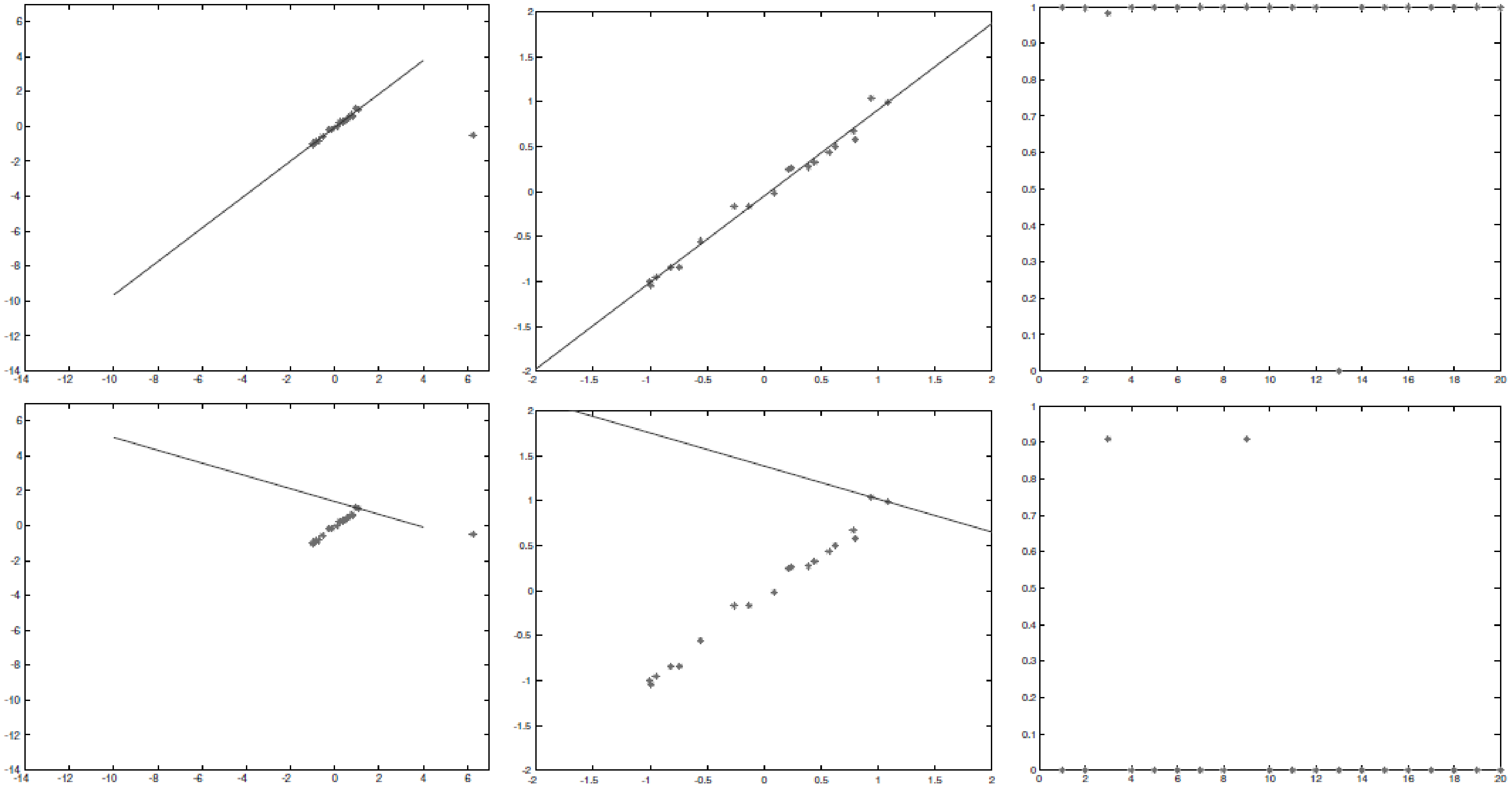
The E-Step For each i, j, compute the soft weights

$$\alpha_{ij} = P(\delta_{ij} = 1 | \boldsymbol{x}_i, \Theta^{(s)}) = \frac{p_j(\boldsymbol{x}_i | \Theta^{(s)}) \pi_j}{\sum_l p_l(\boldsymbol{x}_i | \Theta^{(s)}) \pi_l}.$$

The M-Step We must maximize

$$Q(\Theta; \Theta^{(s)}) = \sum_{i \in \text{observations}} \sum_{j \in \text{components}} \left[\left(\log p_l(\boldsymbol{x}_i | \theta_l) \log \pi_l \right) \alpha_{ij} \right].$$

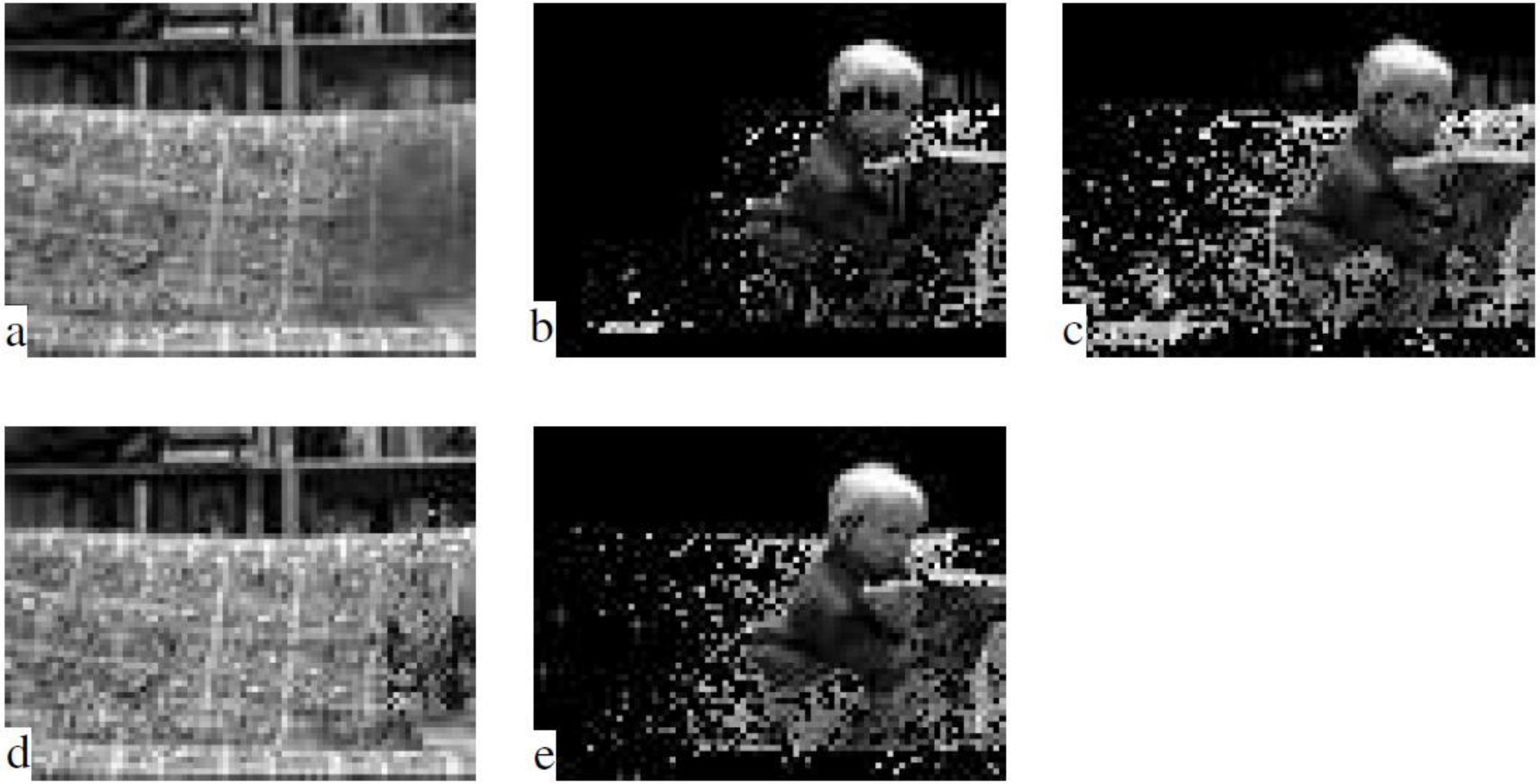
EM can be used to reject outliers



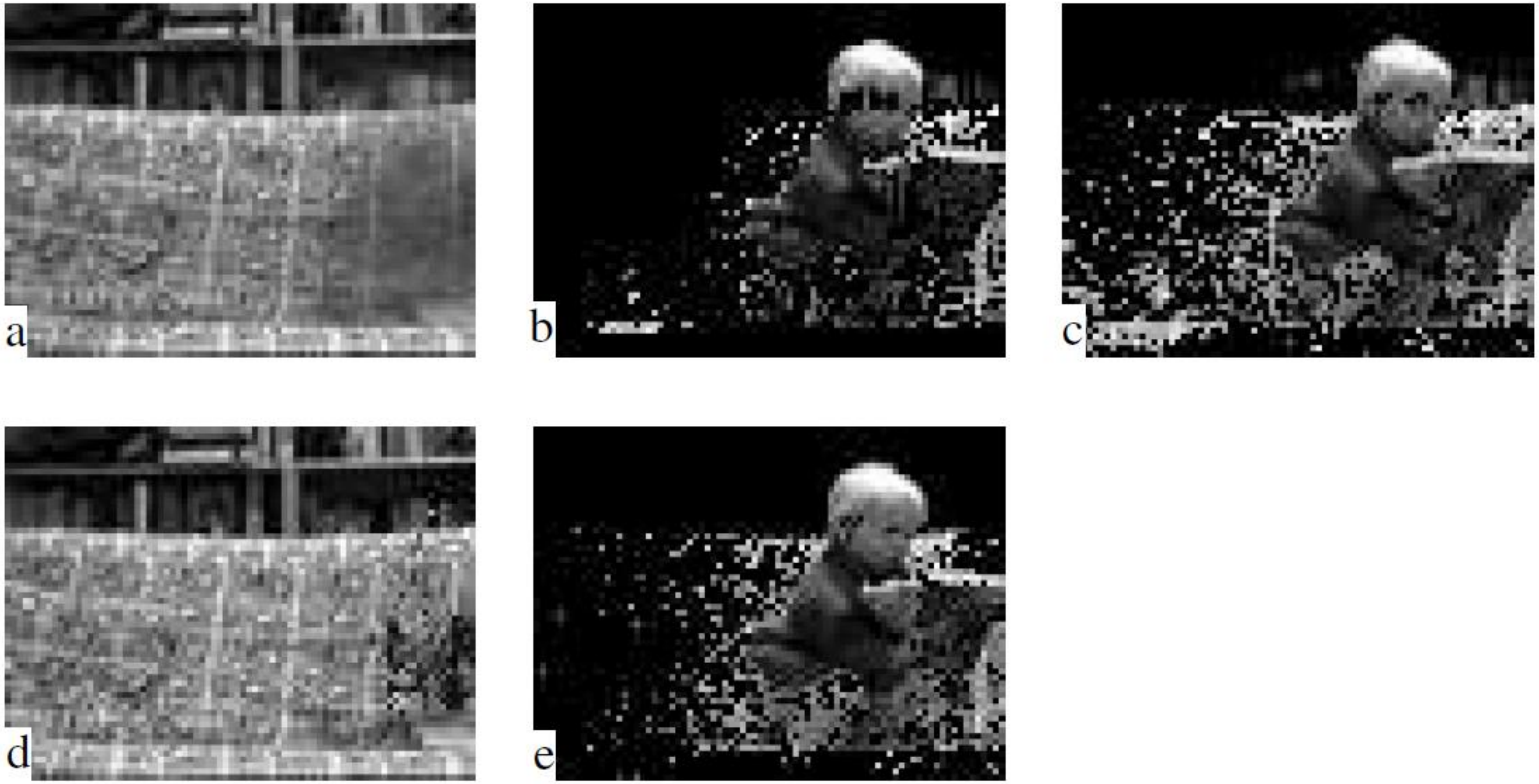
Difficulties with the EM Algorithm

- EM is inclined to get stuck in local minima.
- Some incorrect labelings might be stable
- some points will have extremely small expected weights.
- regard small weights as being equivalent to zero or not

Background subtraction for the sequence of Figure 9.8, using EM



Background subtraction for the sequence of Figure 9.8, using EM (from Figure 9.9, for comparison)



CAMBRIDGE
INSTITUTE OF TECHNOLOGY

Grouping and model fitting

Contents

1) The Hough Transform
2) Fitting Lines and Planes
3) Fitting Curved Structures
4) Robustness
5) Fitting Using Probabilistic Models
6) Motion Segmentation by Parameter Estimation
7) Model Selection: Which Model Is the Best Fit?
8) Conclusion
9) Q&A

$$\nabla \cdot \mathbf{E} = \rho / \varepsilon_0$$

$$\nabla \cdot \mathbf{B} = 0$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$$

$$\nabla \times \mathbf{B} = \mu_0 \varepsilon_0 \frac{\partial \mathbf{E}}{\partial t} + \mu_0 \mathbf{j}_c$$

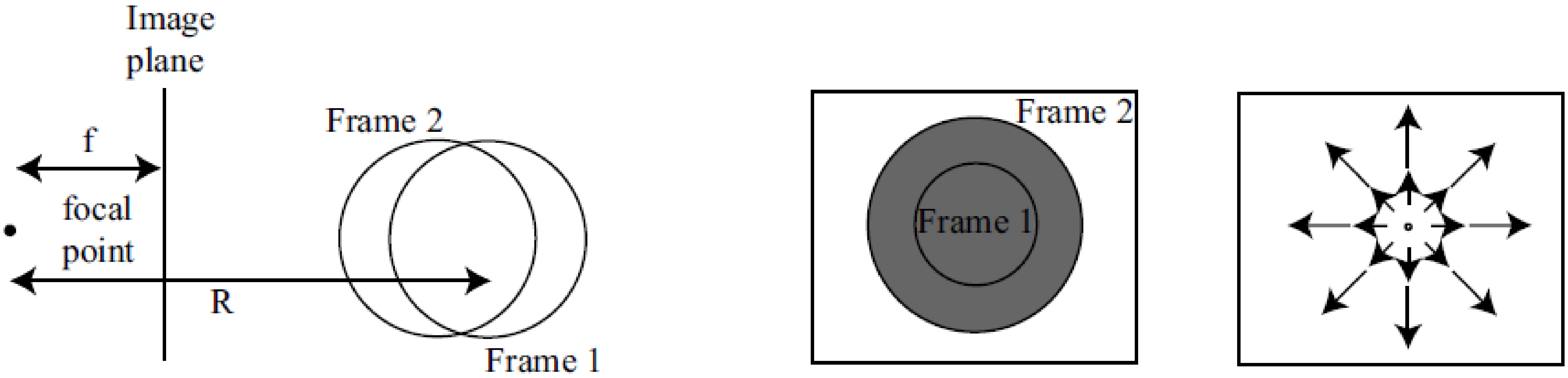
where

$$\nabla = \hat{\mathbf{i}} \frac{\partial}{\partial x} + \hat{\mathbf{j}} \frac{\partial}{\partial y} + \hat{\mathbf{k}} \frac{\partial}{\partial z}$$

Optical flow

- Consider two frames of a motion sequence produced by a moving camera
- For small movement, see relatively few new points, and lose relatively few points
- join each point in 1st frame to its corresponding point on2nd frame (which is overlaid) with an arrow.
- field of arrows can be thought of as the instantaneous movement in the image.
- The arrows are known as the optical flow, a notion originally due to Gibson (1950).

A sphere of radius R approaches a camera along the Z axis, at velocity V



Optical flow and motion

- Flow is particularly informative about relations between the viewer's motion - egomotion
- when viewed from a moving car,
- distant objects have much slower apparent motion than close objects
- flow arrows on distant objects will be shorter than those on nearby objects.

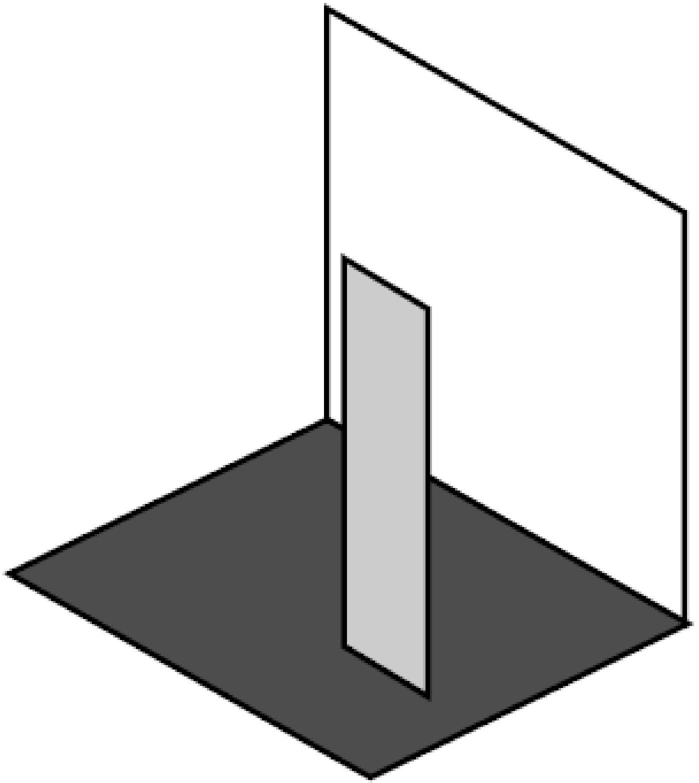
Focus of expansion

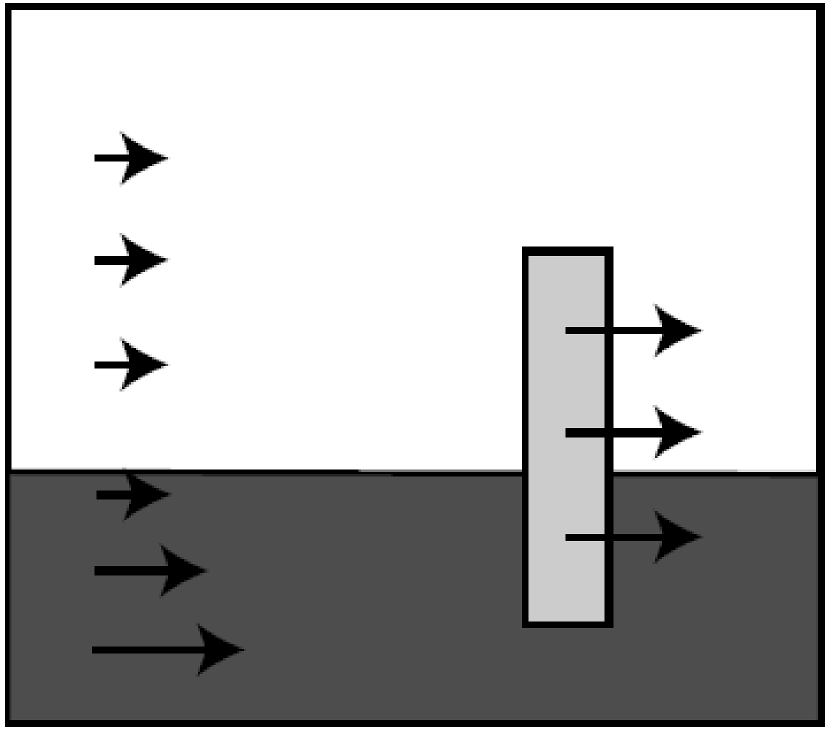
- assume the egomotion is pure translation in some direction
- the image point in that direction, which is known as the focus of expansion
- will not move, and all the optical flow will be away from that point
- a flow field tells us something about how we are moving.

Time to contact - derivation

- A sphere of radius R whose center lies along the direction of motion
- at depth Z will produce a circular image region of radius $r = fR/Z$.
- If it moves down the Z axis with speed $V = dZ/dt$,
- rate of growth of this region in the image will be $dr/dt = -fRV/Z^2$.
- $$\text{time to contact} = -\frac{Z}{V} = \frac{r}{\left(\frac{dr}{dt}\right)}.$$

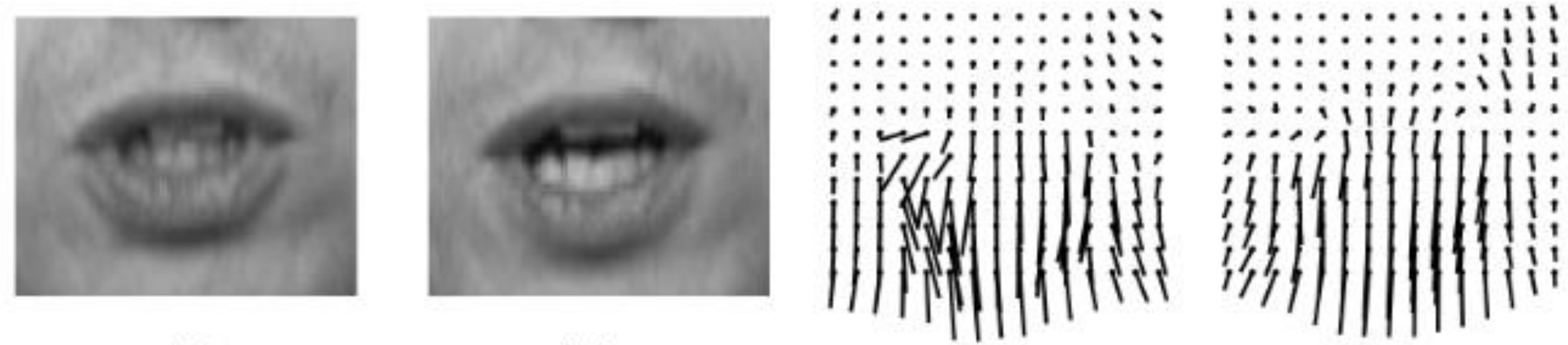
Optic flow fields can be used to structure or segment a scene





Optic flow fields can be used to structure or segment a scene

Optic flow fields can be used to structure or segment a scene



(a) (b) (c) (d)

Flow Models - affine motion model

θ_i for the ith component of the parameter vector, $\boldsymbol{F}_i$ for the ith flow basis vector field, and $\boldsymbol{v}(\boldsymbol{x})$ for the flow vector at pixel $\boldsymbol{x}$, one has

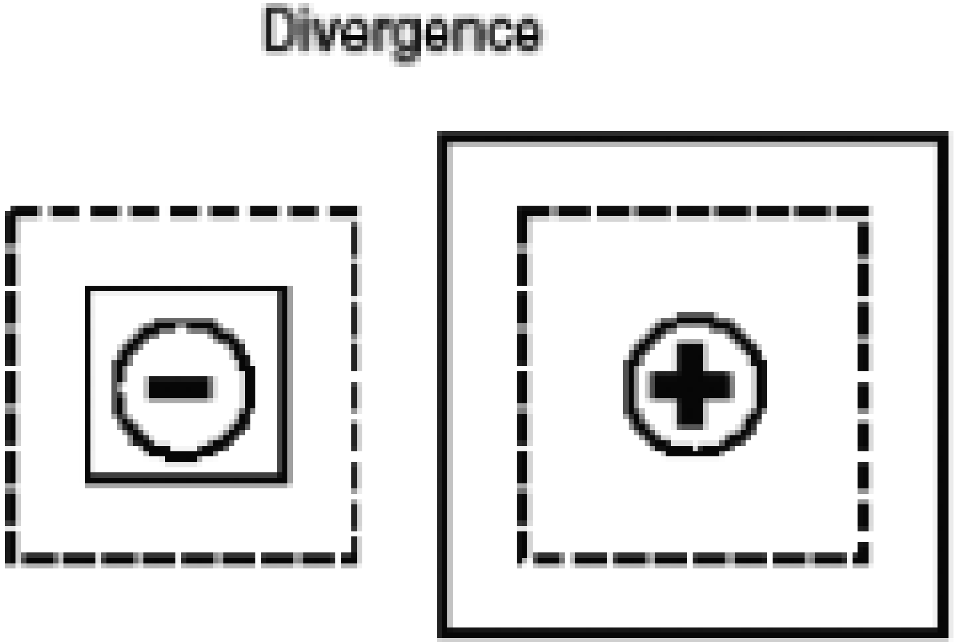
$$\boldsymbol{v}(\boldsymbol{x}) = \sum_i \theta_i \boldsymbol{F}_i$$

In the *affine motion model*, we have

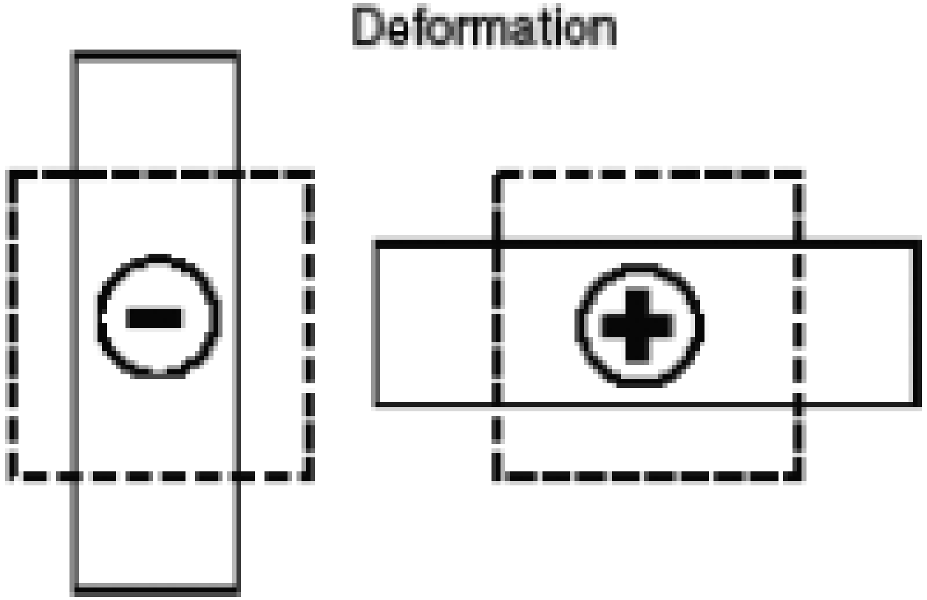
$$\boldsymbol{v}(\boldsymbol{x}) = \begin{pmatrix} 1 & x & y & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & x & y \end{pmatrix} \begin{pmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \\ \theta_5 \\ \theta_6 \end{pmatrix}.$$

$$\boldsymbol{v}(\boldsymbol{x}) = \begin{pmatrix} 1 & x & y & 0 & 0 & 0 & x^2 & xy \\ 0 & 0 & 0 & 1 & x & y & xy & y^2 \end{pmatrix} \begin{pmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \\ \theta_5 \\ \theta_6 \\ \theta_7 \\ \theta_8 \end{pmatrix}.$$

Typical flows generated by the model $(u(x), v(x))^T = (\theta_1 + \theta_2 x + \theta_3 y + \theta_7 x^2 + \theta_8 xy, \theta_4 + \theta_5 x + \theta_6 y + \theta_7 xy + \theta_8 y^2)$.

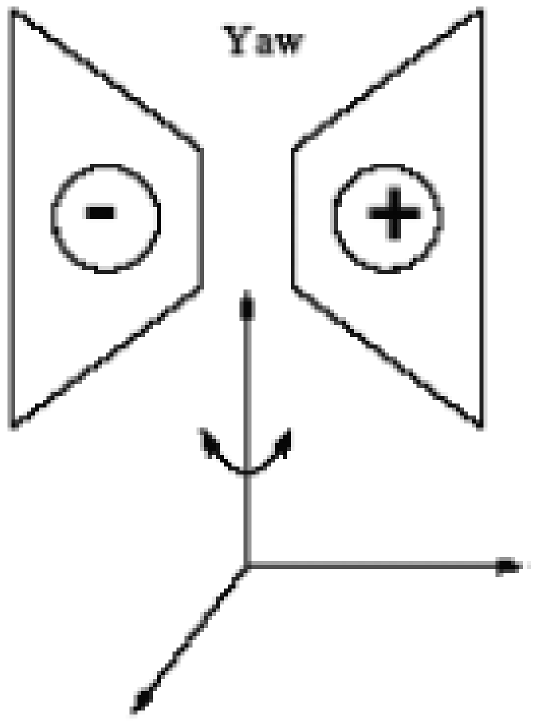


Divergence occurs when the image is scaled;
for example, $\theta = (0,1,0,0,0,1,0,0)$



Deformation occurs when one direction shrinks and another grows (for example, rotation about an axis parallel to the view plane in an orthographic camera);
for example, $\theta = (0,1,0,0,0,-1,0,0)$.

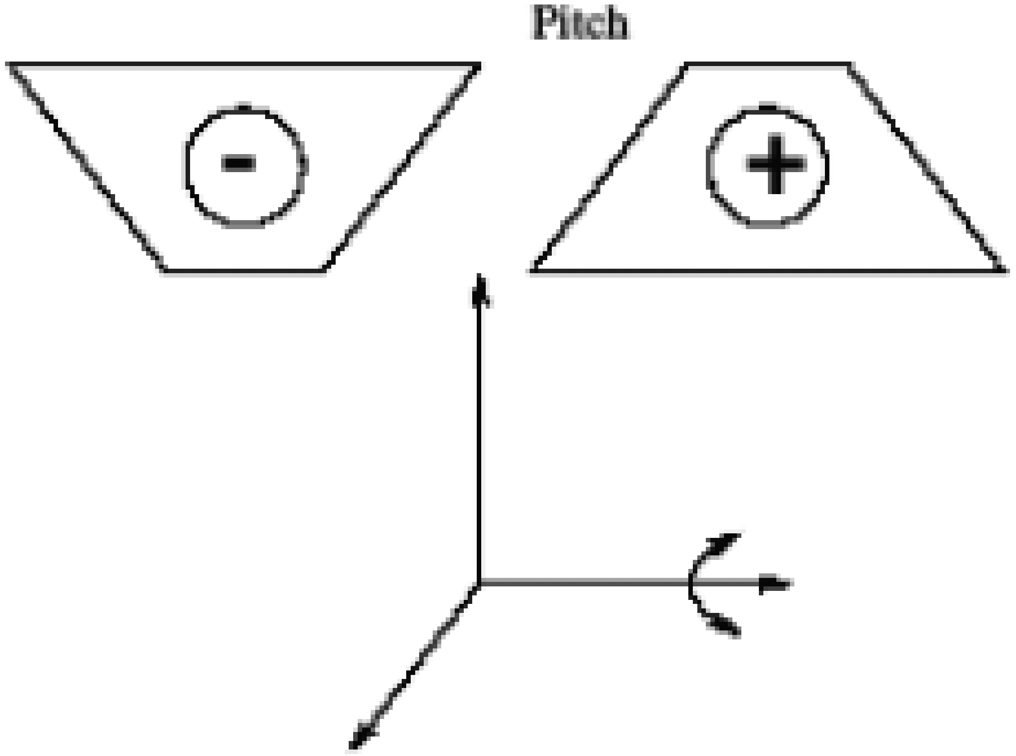
Typical flows generated by the model $(u(x), v(x))^T = (\theta_1 + \theta_2 x + \theta_3 y + \theta_7 x^2 + \theta_8 xy, \theta_4 + \theta_5 x + \theta_6 y + \theta_7 xy + \theta_8 y^2)$.



Yaw models

rotation about a vertical axis in a perspective camera;
for example $\theta = (0,0,0,0,0,1, 0)$.

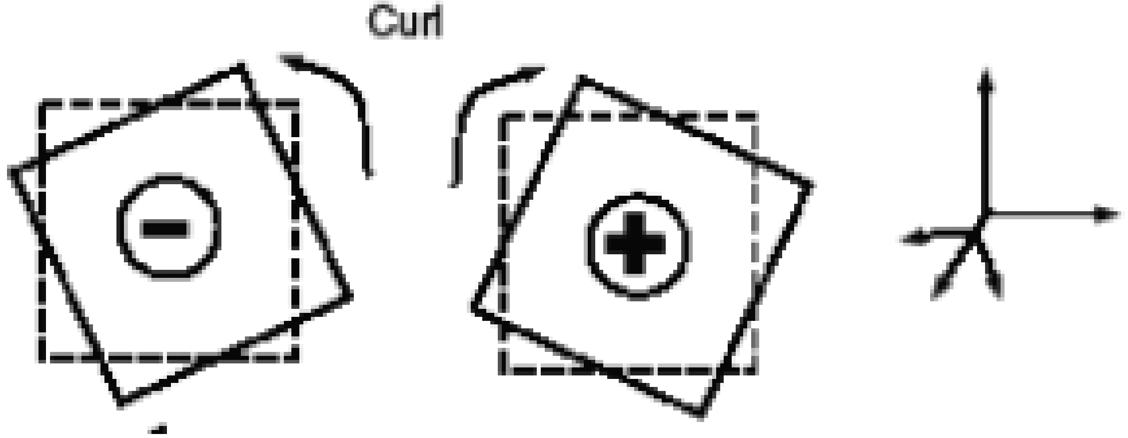
Typical flows generated by the model (u(x), v(x)T = (θ1 +θ2x+θ3y + θ7x2+θ8xy, θ4 +θ5x+θ6y +θ7xy + θ8y2).



pitch models rotation about a horizontal axis in a perspective camera;

for example θ = (0,0,0,0,0,0,1).

Typical flows generated by the model $(u(x), v(x))^T = (\theta_1 + \theta_2 x + \theta_3 y + \theta_7 x^2 + \theta_8 xy, \theta_4 + \theta_5 x + \theta_6 y + \theta_7 xy + \theta_8 y^2)$.



Curl can result from in plane rotation;

for example, $\theta = (0,0,-1,0,1,0,0,0)$.

Motion Segmentation with Layers

Layered motion model

- estimate a flow model for 2 frames that is a mixture of k parametric flow models
- The motion at each pixel in the first frame will come from this mixture,
- will take the pixel to some pixel in the second frame, same brightness value
- Pixels whose flow came from the first model would be in segment one
- set of rigid objects at different depths and a moving camera
- separate motion fields are as layers and the model as a layered motion model

a probabilistic model

$$V_{uv,j} = \left\{ \begin{array}{c} 1, \text{ if the } x, y\text{th pixel belongs to the } j\text{th motion field} \\ 0, \text{ otherwise} \end{array} \right\}.$$

The complete data log-likelihood becomes

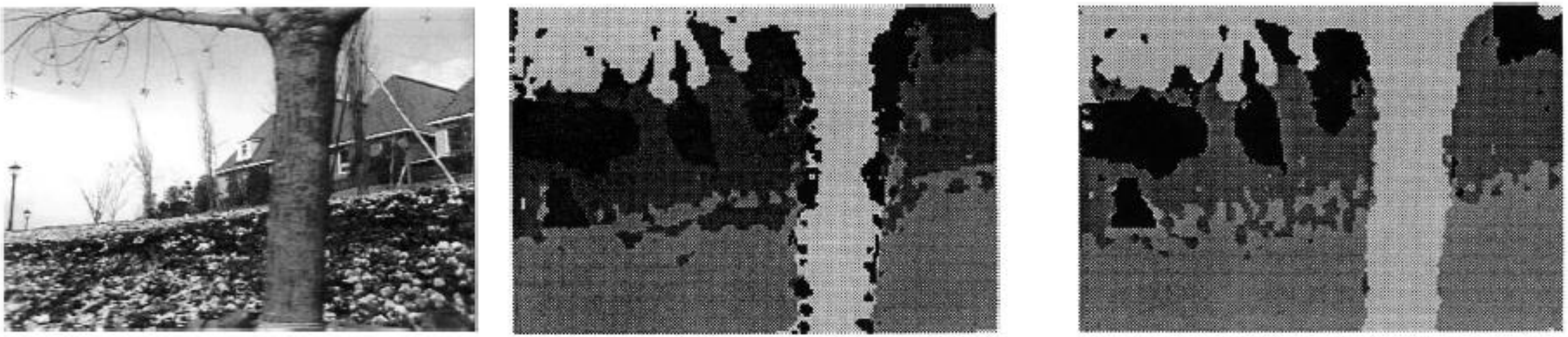
$$L(V, \Theta) = -\sum V_{xy,j} \frac{(I_1(x,y) - I_2(x + v_1(x,y;\theta_j), y + v_2(x,y;\theta_j)))^2}{2\sigma^2} + C,$$

where $\Theta = (\theta_1, \ldots, \theta_k)$. Setting up the EM algorithm from here on is straightforward. As before, the crucial issue is determining

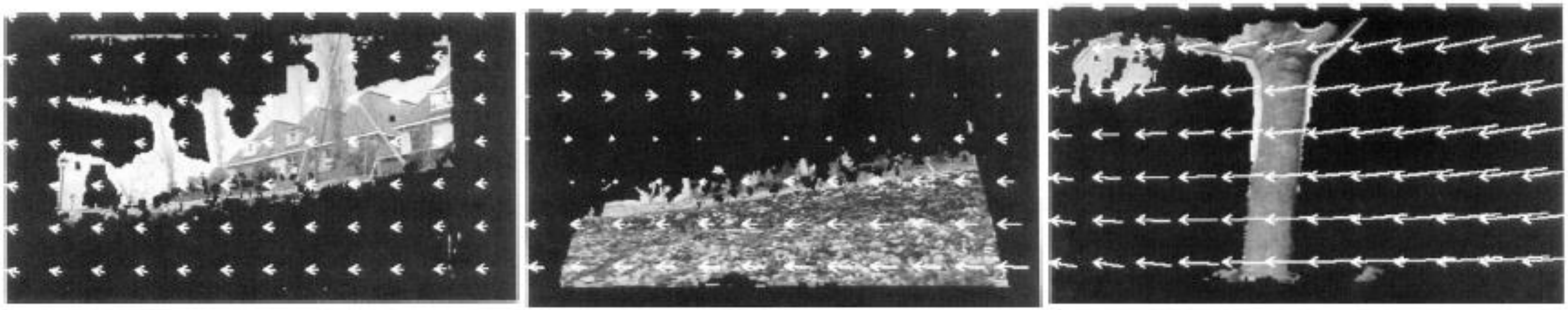
$$P\{V_{xy,j} = 1 | I_1, I_2, \Theta\}.$$

These probabilities are often represented as *support maps*—maps assigning a gray-level representing the maximum probability layer to each pixel (Figure 10.16).

a map indicating to which layer pixels in a frame of the flower garden sequence belong, obtained by clustering local estimates of image motion



a map indicating to which layer pixels in a frame of the flower garden sequence belong, obtained by clustering local estimates of image motion



Grouping and model fitting

Contents

$$\nabla \cdot \mathbf{E} = \rho/\varepsilon_0$$

$$\nabla \cdot \mathbf{B} = 0$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$$

$$\nabla \times \mathbf{B} = \mu_0 \varepsilon_0 \frac{\partial \mathbf{E}}{\partial t} + \mu_0 \boldsymbol{j}_c$$

where

$$\nabla = \hat{\mathbf{i}}\frac{\partial}{\partial x} + \hat{\mathbf{j}}\frac{\partial}{\partial y} + \hat{\mathbf{k}}\frac{\partial}{\partial z}$$

Trade off between bias and variance

- The data points are a sample that comes from some underlying process
- Representing a lot of data points with a single line is a biased representation,
- because it cannot represent all the complexity of the model that produced the dataset.
- Some information about the underlying process is inevitably lost.

Trade off between bias and variance

- represent the data points with a zigzag set of lines that joined them up
- representation would have no bias, but
- would be different for each new sample of data points from the same source
- our estimate of the model we fit changes wildly from sample to sample
- it is overwhelmed by variance.

Selection bias

- predict future samples from the mode
- A proper choice of the parameters predicts future samples from the model—a test set—
- the parameters chosen ensure that the model is an optimal fit to the training set,
- rather than to the entire set of possible data. The effect is known as selection bias.

AIC: An Information Criterion

- choosing the model with minimum value of $-2L(x;\Theta*) + 2p$.
- lacks a term in the number of data points.
- AIC tends to overfit—
- fits the training set well but doesn't perform as well on test sets.

Bayesian Methods and Schwartz's BIC

Bayesian Methods and Schwartz's BIC

For simplicity, let us write $\mathcal{D}$ for the data, $\mathcal{M}$ for the model, and θ for the parameters. Bayes' rule then yields:

$$
\begin{aligned}
P(\mathcal{M}|\mathcal{D}) &= \frac{P(\mathcal{D}|\mathcal{M})}{P}(\mathcal{M})P(\mathcal{D}) \\
&= \frac{\int P(\mathcal{D}|\mathcal{M}_i,\theta)P(\theta)d\theta P(\mathcal{M})}{P(\mathcal{D})}.
\end{aligned}
$$

Now we could choose the model for which the posterior is large. Computing this posterior can be difficult, but, by a series of approximations, we can obtain a criterion

$$-L(\mathcal{D};\theta^*) + \frac{p}{2}\log N$$

(where N is the number of data items). Again, we choose the model that minimizes

Minimum description length criterion

are similar ideas rooted in information theory, due to Kolmogorov, and expounded in Cover and Thomas (1991). Surprisingly, the BIC emerges from this analysis, yielding

$$-L(\mathcal{D}; \theta^*) + \frac{p}{2} \log N.$$

Again, we choose the model that minimizes this score.

Model Selection Using Cross-Validation

- split the training set intoone to fit the model and the other the test the fit.
- This approach is known as cross-validation.
- use cross-validation to determine the number of components in a model
- choosing the model that performs best on the test data.

Leave-one-out cross-validation

- fit a model to each set of N−1 of the training set
- compute the error on the remaining data point.
- sum these errors to obtain an estimate of the model error
- The model that minimizes this estimate is then chosen.

Conclusion

Fitting data points to a model

- Variety of methods
- Different scores
- Choice of model

Q&A

Contact

- **Prof. D. Antony Louis Piriyakumar**
  Dean (Research and development)
  Cambridge Institute of Technology
- K.R. Puram,
  560036 Bengaluru, India
- Mobile: +91 98459 25132
- E-mail:
  dean_rd@Cambridge.edu.in