Images and Features

CS635 Spring 2010

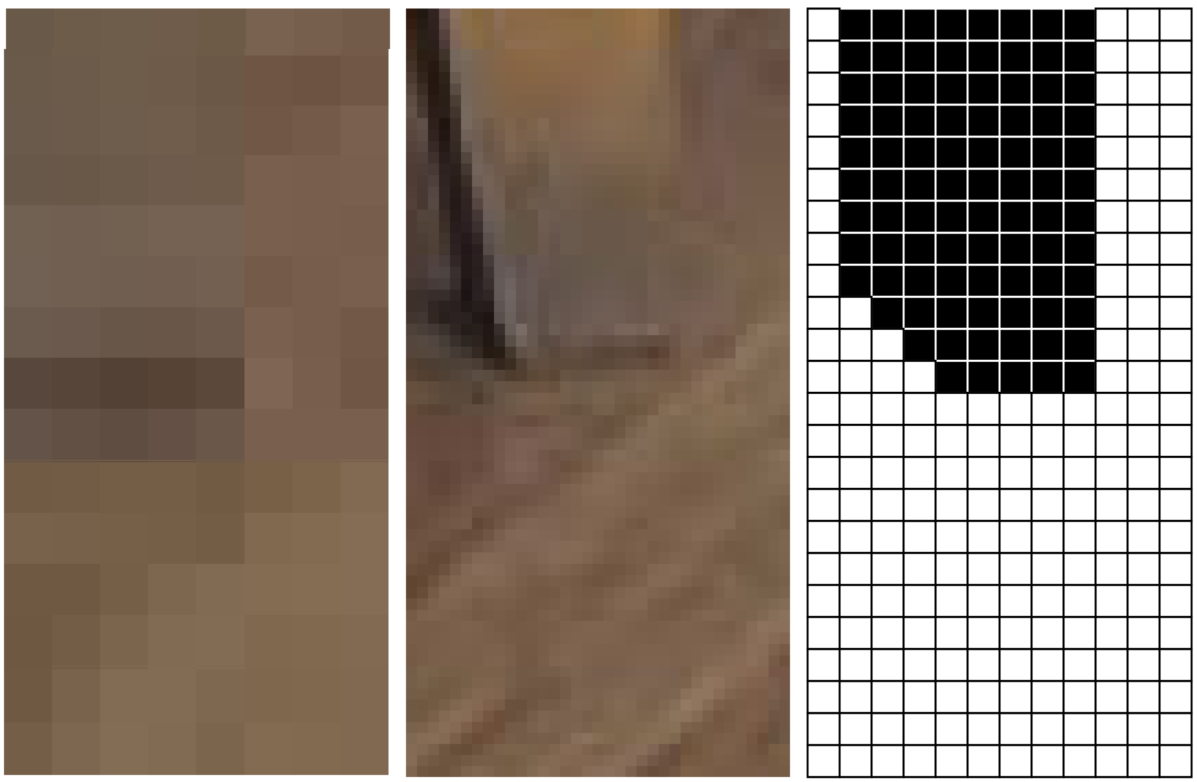
Daniel G. Aliaga
Department of Computer Science
Purdue University

What is an image?

- Images are collections of pixels
- Pixels are measurements of …?

What is an image?

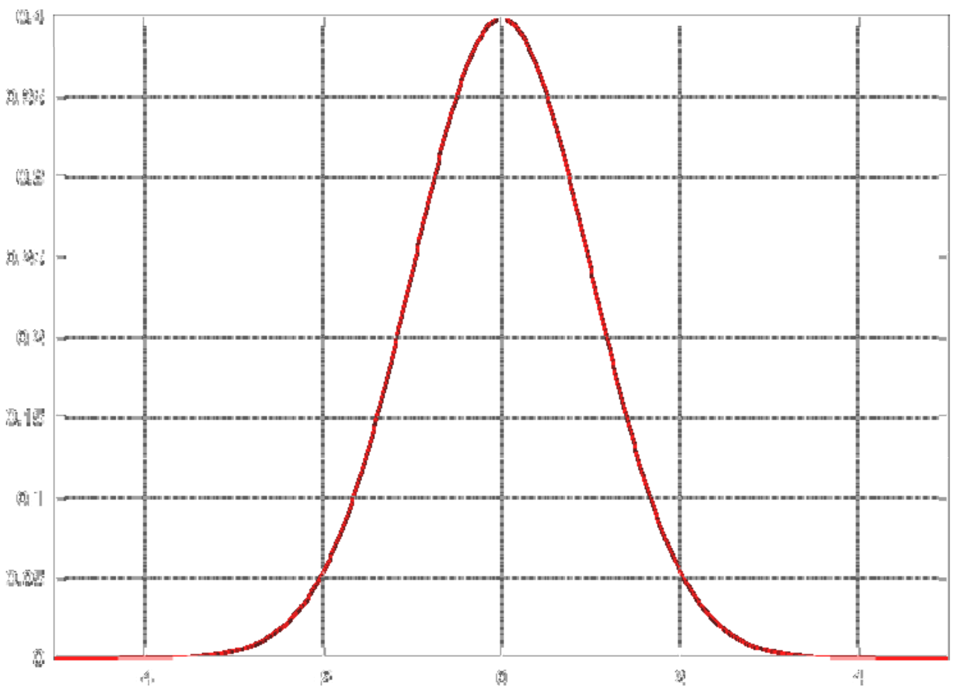
- Images can be “thresholded” to simplify them
  - If value ≥ 128 make white, else black



What is an image?

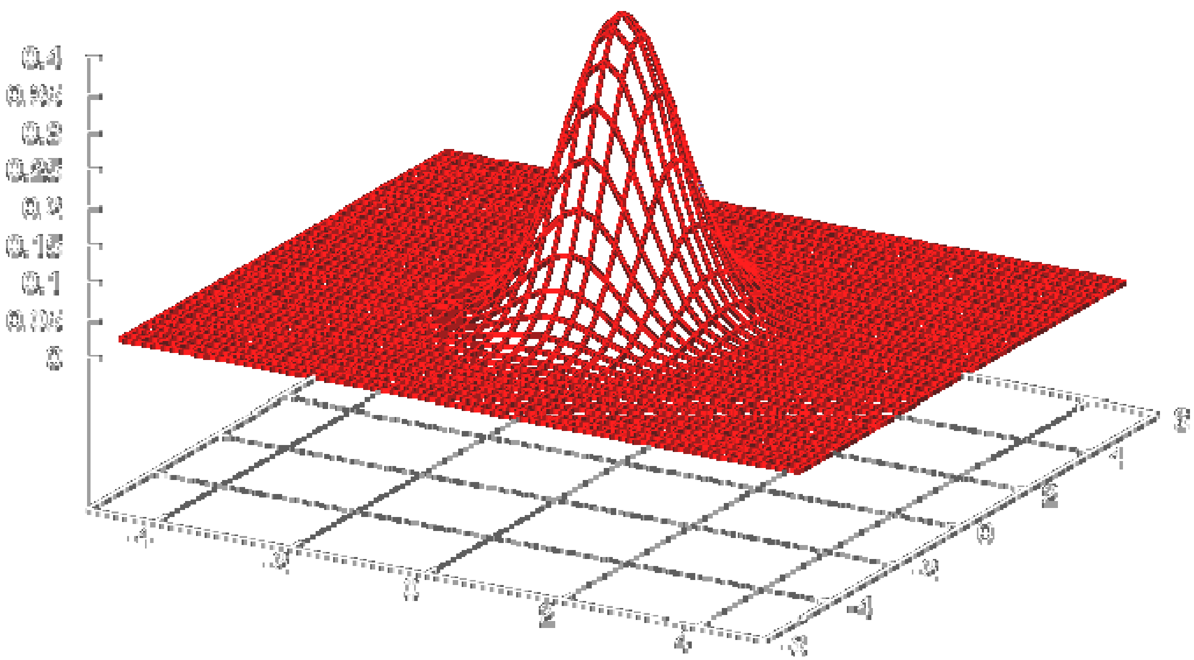
- Images have a resolution and can be filtered…

Gaussian Filters



$$G_1(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}}$$

Gaussian Filters



$$G_2(x, y) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

What is an image feature?

point

…ambiguous?

What is an image feature?

edge

What is an image feature?

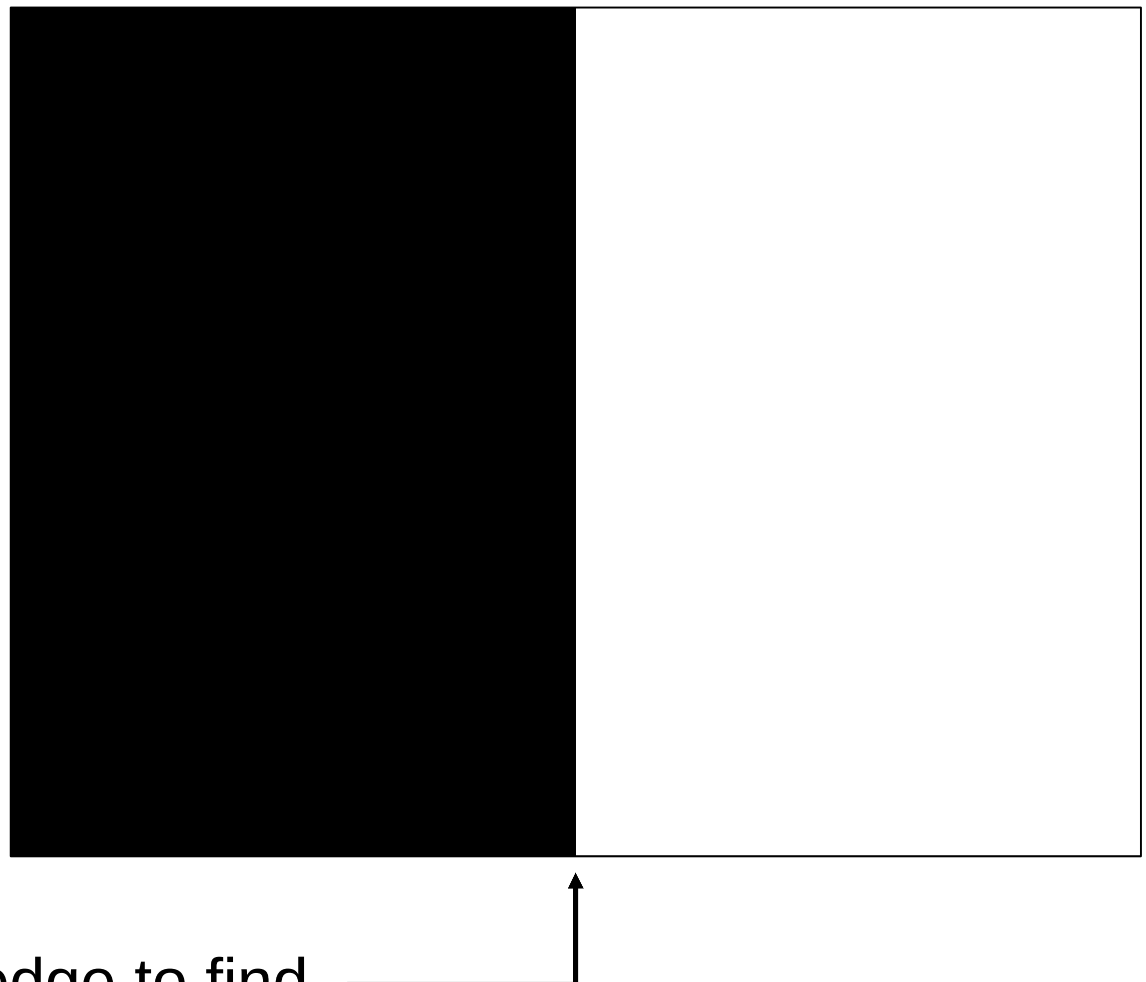
What is an image feature?

texture

Feature Detection

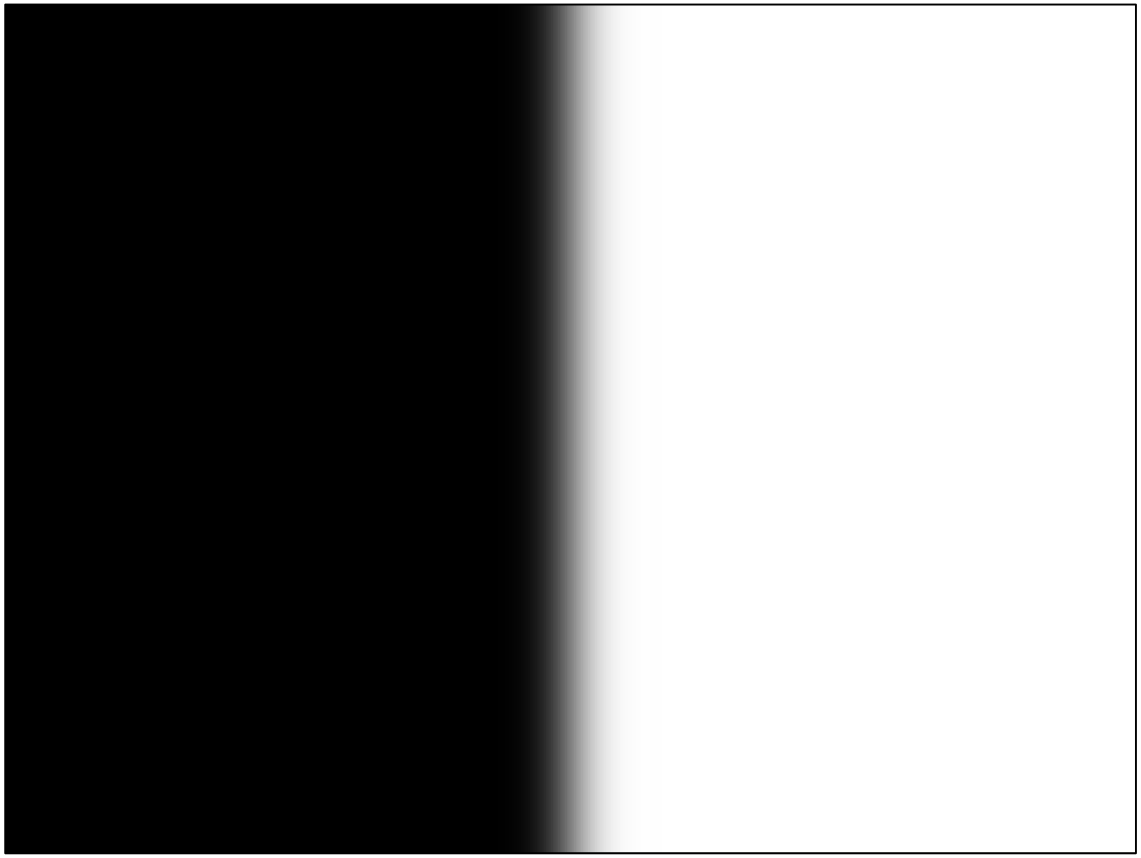
- Can we precisely define/detect features?

What is an Edge?



An easy edge to find

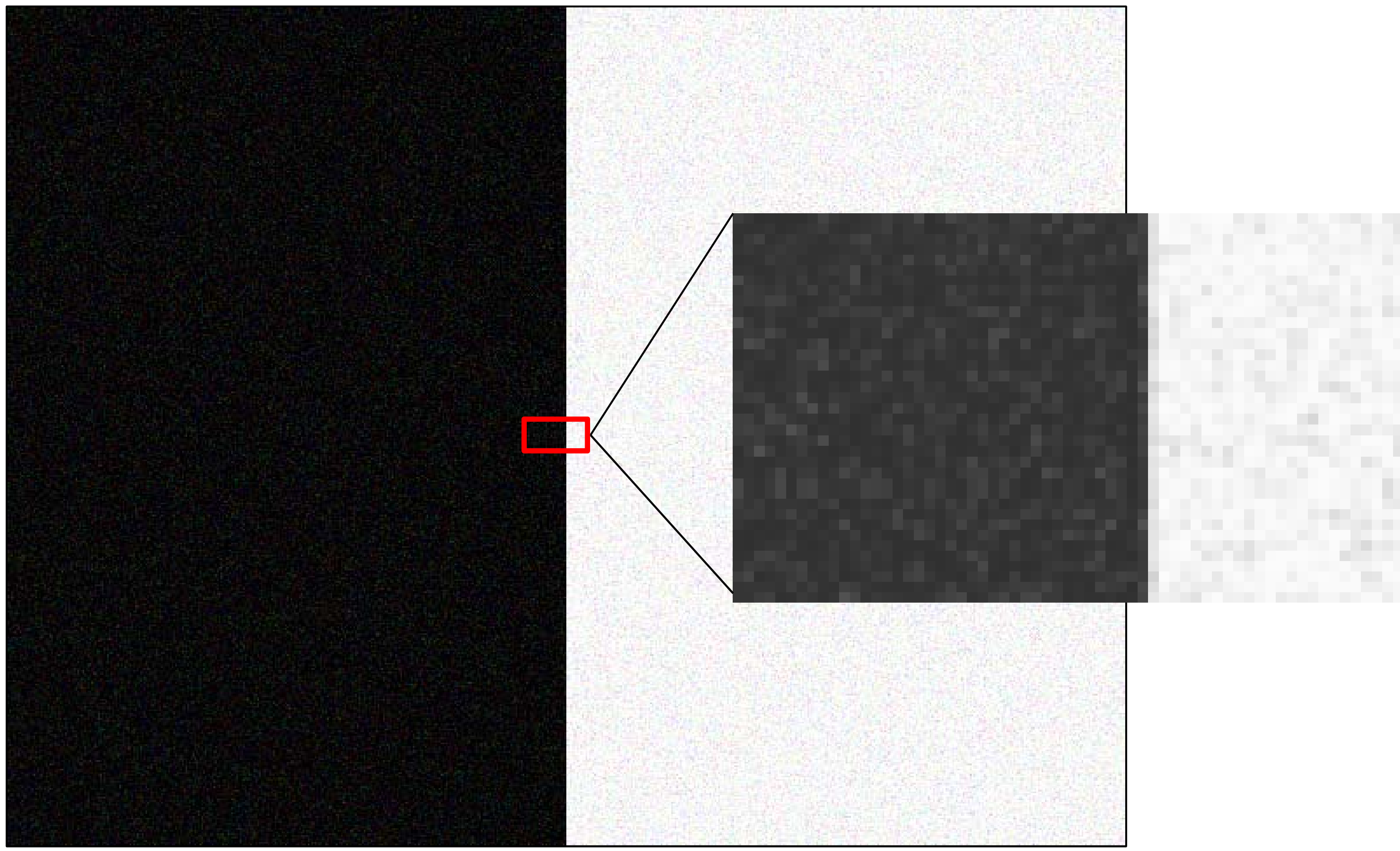
What is an Edge?



Where is the edge?

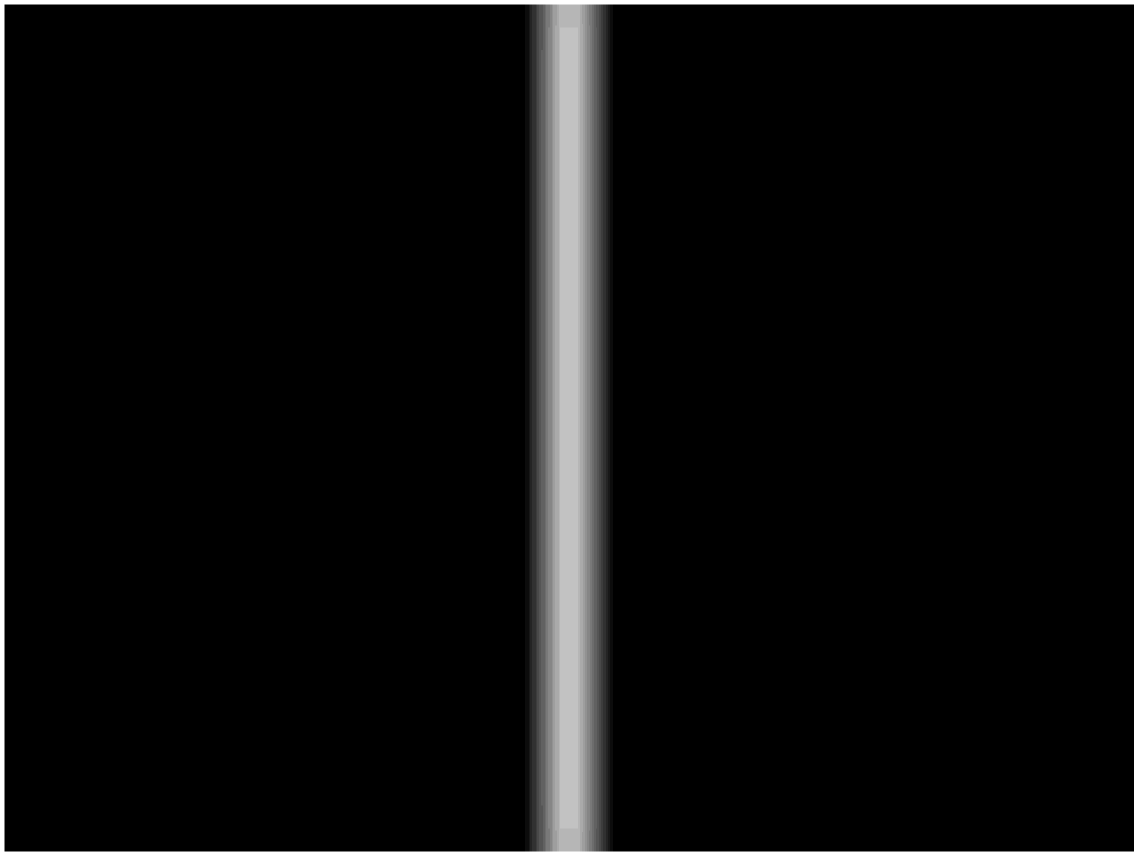
Single pixel wide or multiple pixels?

What is an Edge?



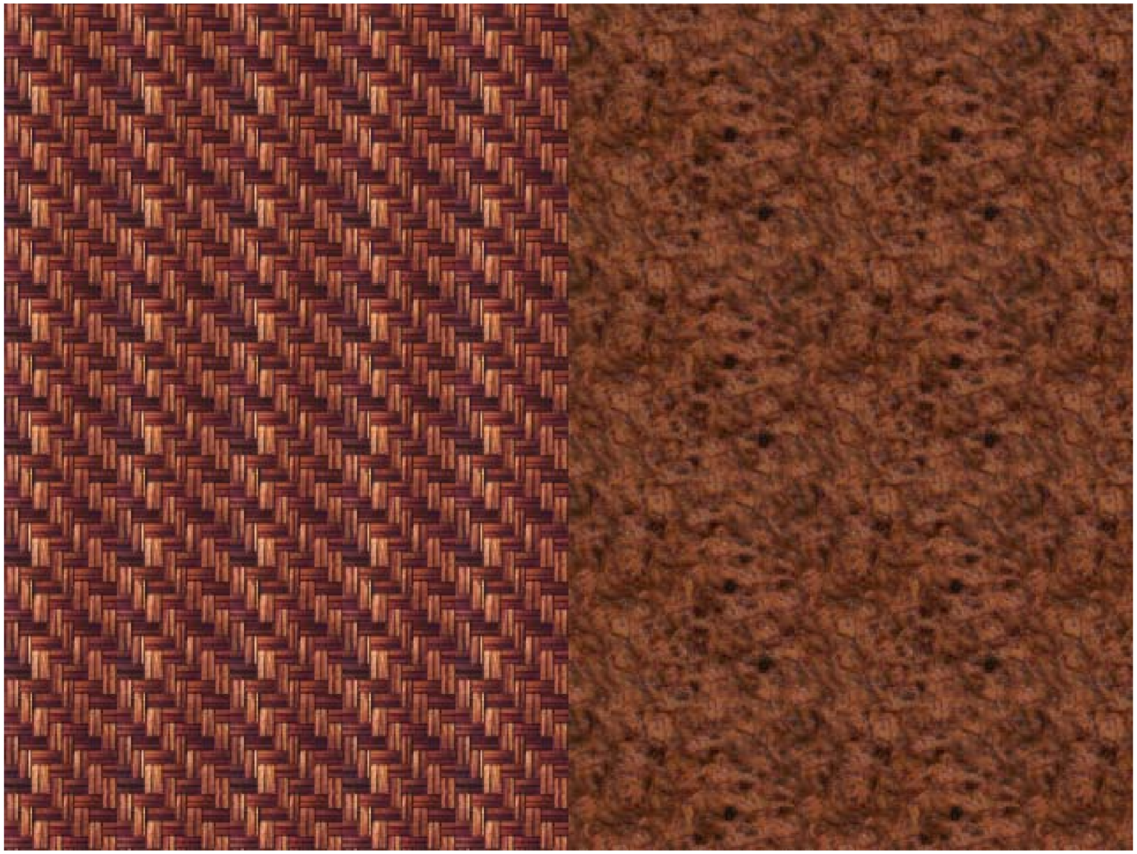
Noise: have to distinguish noise from an actual edge…

What is an Edge?



Is this one edge or two?

What is an Edge?



Is a texture discontinuity an edge?

Canny Edge Detector

Example edge detector: is the result perfect?

Feature Tracking

- Follow feature “X” from image i to image j
- Requires “matching” features
- Some options:
  - Window matching
  - Corner matching
  - Scale-Invariant Feature Transform matching (SIFT)

Window Matching

- Given an image i of a face…
- Find the person in an image j…

X

Where is X?

Window Matching

- Its not easy…

X

Which is X?

Window Matching

- Basic Implementation:
  - Use sum-of-squares differences (SSD) with pixel windows

f

? =

g

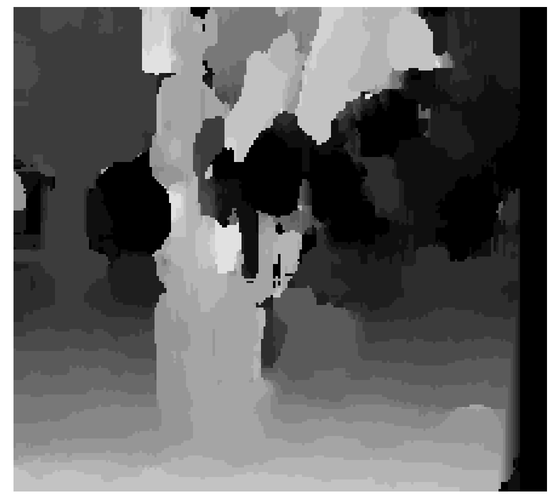
$$SSD = \sum_{[i,j] \in R} (f(i,j) - g(i,j))^2$$

Most popular

Window Matching Problems

- Must choose a good window size

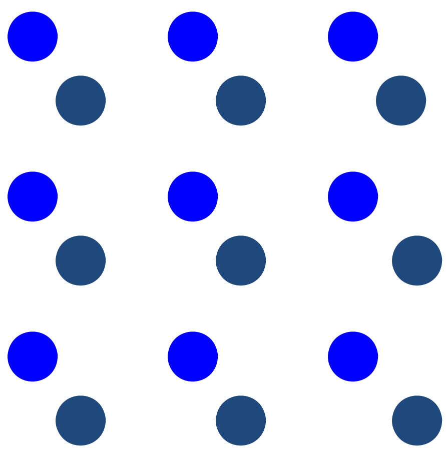
W = 3



W = 20

Window Matching Problems

- When motion is a few pixels or less, motion of an integer number of pixels can be insufficient



Bilinear Interpolation

- A solution: resample the image
- Assume image is locally bilinear

$$I(x, y) = ax + by + cxy + d = 0$$

- Given the value of the image at four points, *I(x,y)*, *I(x+1,y)*, *I(x,y+1)*, *I(x+1,y+1)* we can solve for *a,b,c,d* linearly
- Then, for any *u* between *x* and *x+1*, for any *v* between *y* and *y+1*, we use this equation to find *I(u,v)*

Window Matching Problems

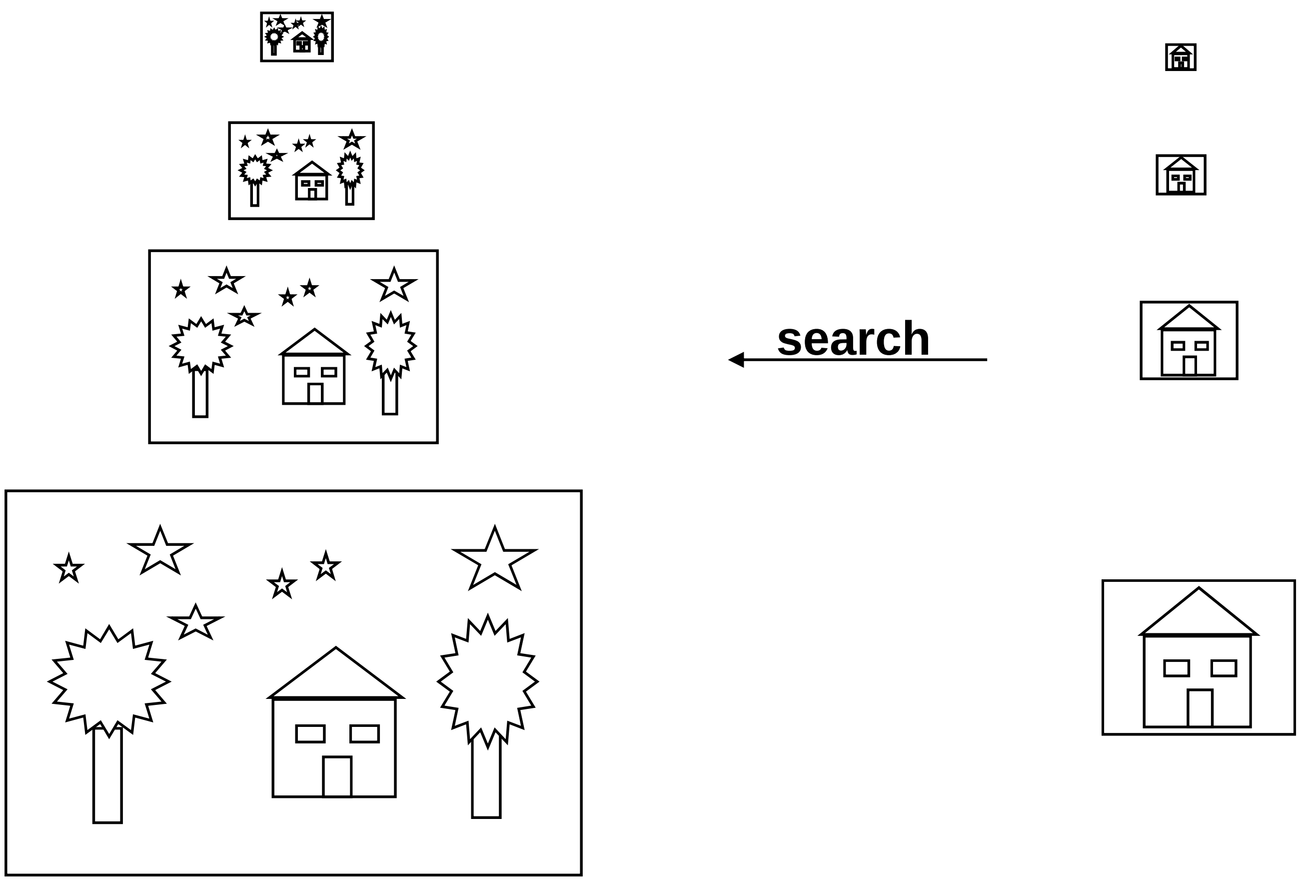
- Where do we look for “X” in image j?

$$\arg\min_{u,v} \sum \left(W(x,y) - I(x+u, y+v)\right)^2$$

- Could range over whole image…
  - Inefficient, false positives
- Or only over a small displacement…
  - What is small? Can “miss” X?

Window Matching Problems

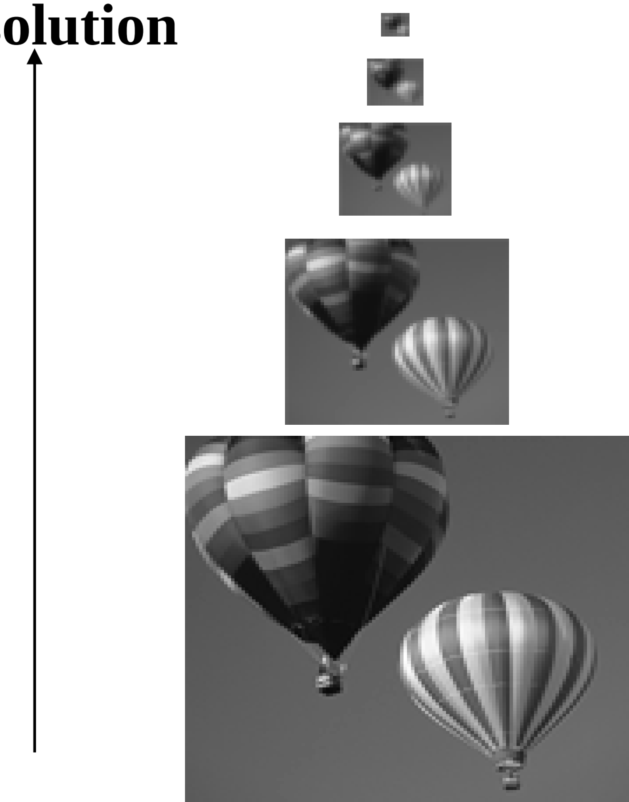
- At what scale do we look for "X" in image j?



Window Matching Problems

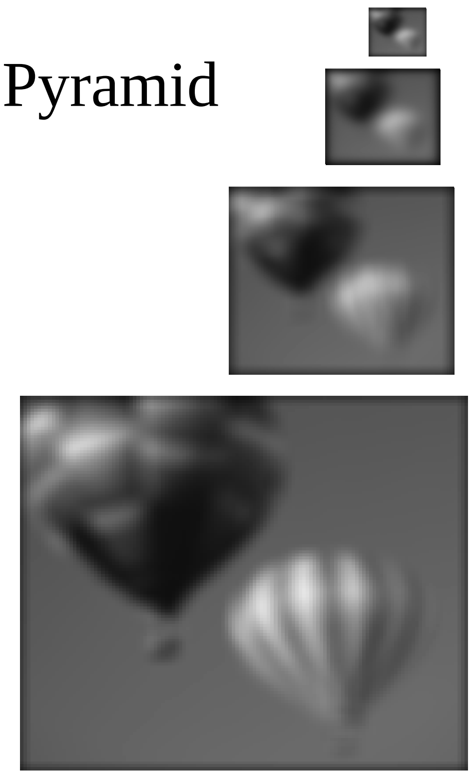
- At what resolution do we look for “X” in image j?

Low resolution



High resolution

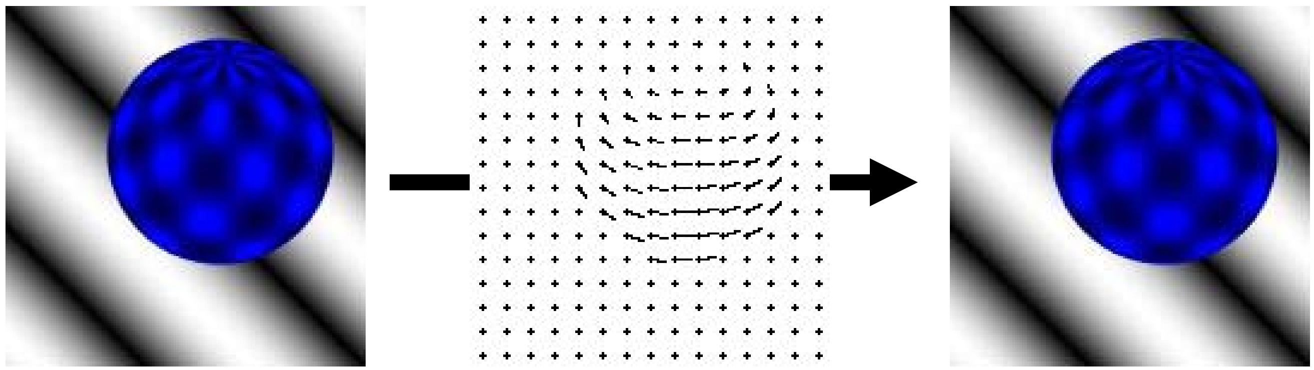
Gaussian Pyramid



Corner Matching

- Let’s first look at optical flow…
- Then, lets define a corner…

Optical Flow of an Image



Optical flow

(Some of the optical flow/corner slides help from Steve Seitz @ UW CS)

Optical Flow of an Image

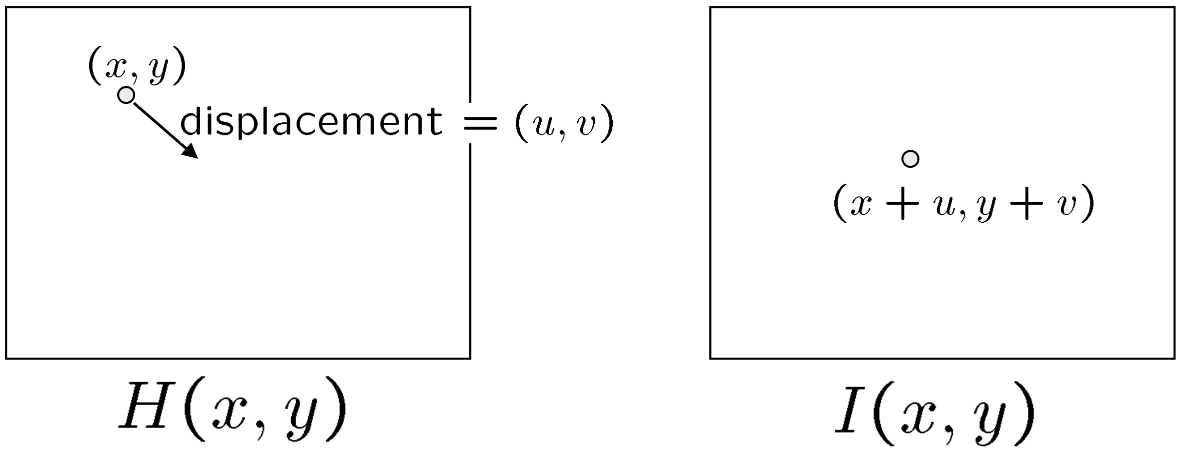
- Can pixels flow in any direction?

Optical Flow of an Image

- Can pixels flow at any speed?

Optical Flow of an Image

- Small motion: (u and v are less than 1 pixel)



Let’s take the Taylor series expansion of *I*:

$$I(x+u, y+v) = I(x, y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \text{higher order terms}$$

$$\approx I(x, y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v$$

Optical Flow Equation

- Combining these two equations

$$\begin{aligned} 0 &= I(x+u, y+v) - H(x,y) \\ &\approx I(x,y) + I_x u + I_y v - H(x,y) \\ &\approx (I(x,y) - H(x,y)) + I_x u + I_y v \\ &\approx I_t + I_x u + I_y v \\ &\approx I_t + \nabla I \cdot [u \;\; v] \end{aligned}$$

- In the limit as u and v go to zero, this becomes exact: $0 = I_t + \nabla I \cdot [\frac{\partial x}{\partial t} \;\; \frac{\partial y}{\partial t}]$

shorthand: $I_x = \frac{\partial I}{\partial x}$

Optical flow equation

$$0 = I_t + \nabla I \cdot [u \;\; v]$$

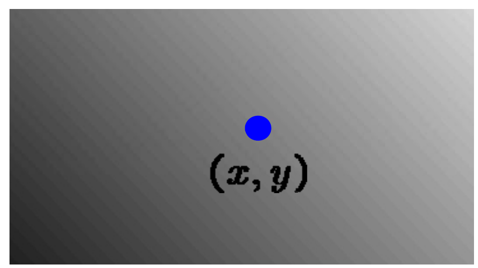
- How many unknowns and equations per pixel?
  - 2 unknowns, 1 equation
- Intuitively, what does this equation mean?
  - The component of the flow in the gradient direction is *determined*
  - The component of the flow parallel to an edge is *unknown*

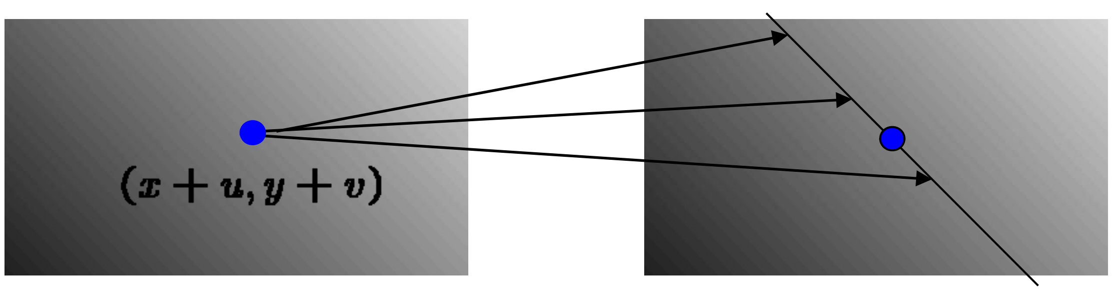
First Order Approximation

When we assume that:

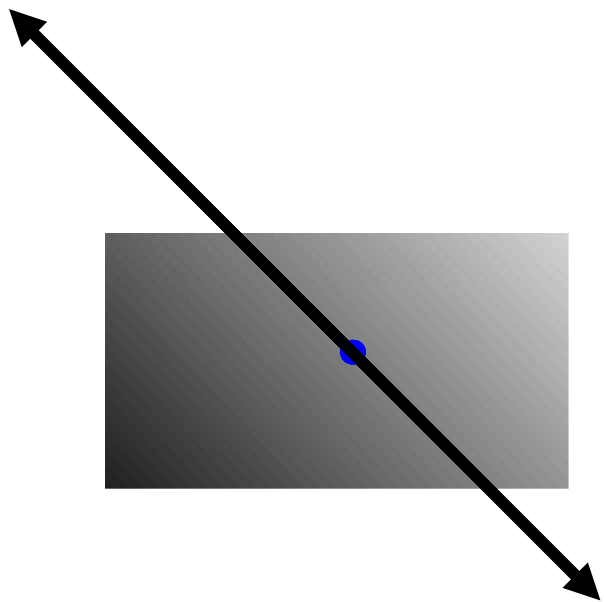
$$I(x+u, y+v) \approx I(x, y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v$$

We assume an image locally is:

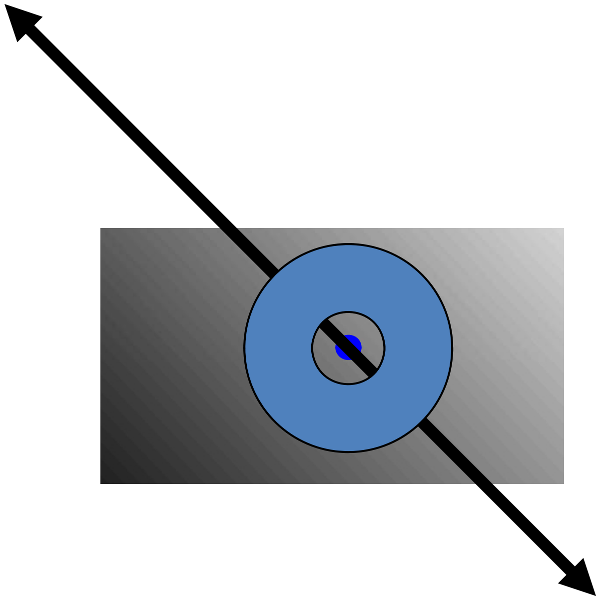




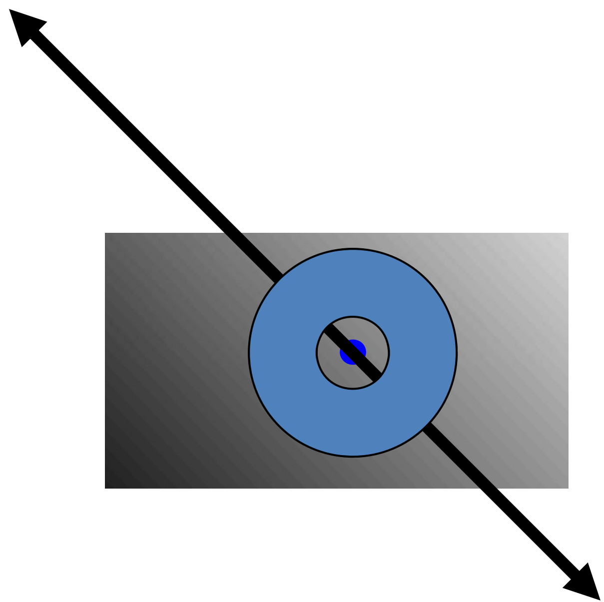
Aperture problem: where are we?



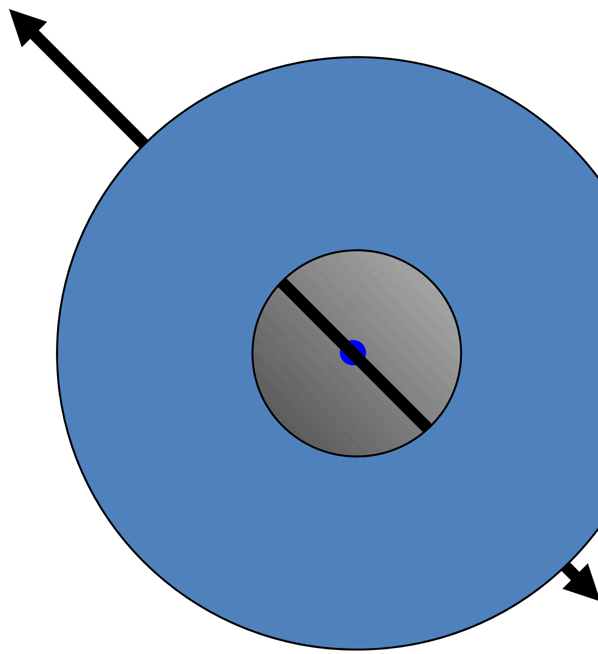
Aperture problem: where are we?



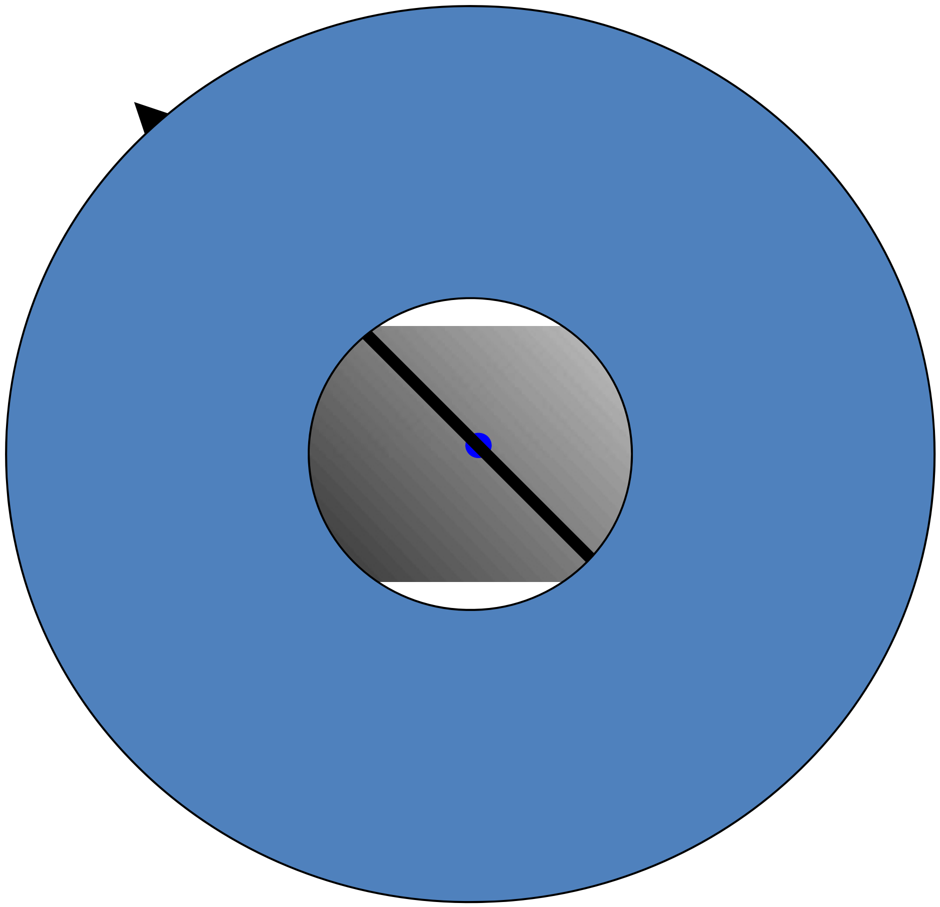
Aperture problem: where are we?



Aperture problem: where are we?



Aperture problem: where are we?



Solving the aperture problem

- How to get more equations for a pixel?
- A solution: impose additional constraints
  - Assume the flow field is smooth locally
  - Pretend the pixel's neighbors have the same (u,v)
  - e.g., use a 5x5 window to get 25 equations per pixel!

$$0 = I_t(\mathbf{p_i}) + \nabla I(\mathbf{p_i}) \cdot [u \;\; v]$$

$$\underbrace{\begin{bmatrix} I_x(\mathbf{p_1}) & I_y(\mathbf{p_1}) \\ I_x(\mathbf{p_2}) & I_y(\mathbf{p_2}) \\ \vdots & \vdots \\ I_x(\mathbf{p_{25}}) & I_y(\mathbf{p_{25}}) \end{bmatrix}}_{\substack{A \\ 25\times 2}} \underbrace{\begin{bmatrix} u \\ v \end{bmatrix}}_{\substack{d \\ 2\times 1}} = - \underbrace{\begin{bmatrix} I_t(\mathbf{p_1}) \\ I_t(\mathbf{p_2}) \\ \vdots \\ I_t(\mathbf{p_{25}}) \end{bmatrix}}_{\substack{b \\ 25\times 1}}$$

Lucas-Kanade flow

- We have more equations than unknowns
- Solve as least squares problem:

$$\underset{25\text{x}2}{A}\ \underset{2\text{x}1}{d} = \underset{25\text{x}1}{b} \longrightarrow \text{minimize } \|Ad - b\|^2$$

- What is the rank? 2
- Thus, $\underset{2\text{x}2}{(A^T A)}\ \underset{2\text{x}1}{d} = \underset{2\text{x}1}{A^T b}$

$$\underbrace{\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix}}_{A^T A} \begin{bmatrix} u \\ v \end{bmatrix} = - \underbrace{\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}}_{A^T b}$$

(e.g., summations over all pixels in a K x K window)

Conditions for solvability

$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

$$A^T A \qquad\qquad\qquad\qquad A^T b$$

- When is this solvable?
  - **$A^T A$** should be invertible
  - **$A^T A$** should not be too small due to noise
  - eigenvalues λ_1 and λ_2 of **$A^T A$** should not be too small
  - **$A^T A$** should be well-conditioned
    - λ_1 / λ_2 should not be too large (λ_1 = larger eigenvalue)

Back to Corner Matching

- Lets look more closely at the optical flow:

Gradient with respect to x, times gradient with respect to y

Sum over a small region, the hypothetical corner

$$C = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}$$

Matrix is symmetric

Back to Corner Matching

- Consider this case

$$C = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

- What does this mean about the gradients?
  – All gradients in pixel neighborhood are (k,0), (0,c) or (0,0) (=they “cancel”)

Back to Corner Matching

- Consider this case

$$C = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

- What is region like if:
  - $\lambda_1 = 0?$
  - $\lambda_2 = 0?$
  - $\lambda_1 = 0$ and $\lambda_2 = 0?$
  - $\lambda_1 > 0$ and $\lambda_2 > 0?$

Back to Corner Matching

- From SVD it follows that since C is symmetric:

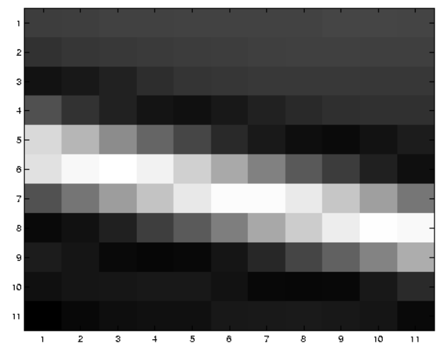
$$C = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$$

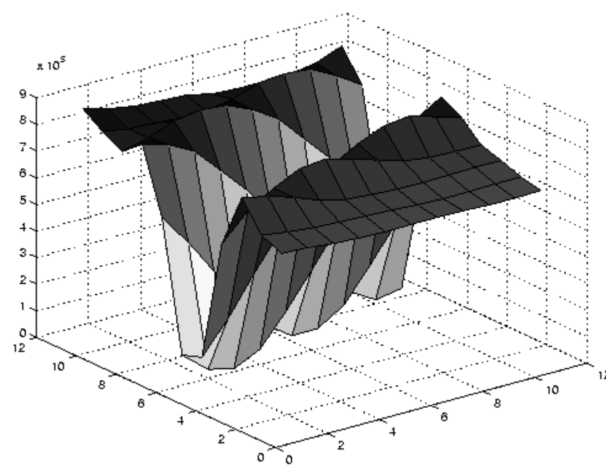
- Where *R* is a rotation matrix
- What does this mean?
  - – All instances are similar to the previous slide!

Corner Features

- Corners are when λ_1 and/or λ_2 are big
- Corners are regions with two different directions of gradient (at least).
- Aperture problem disappears at corners.
- Not perfect, but works…

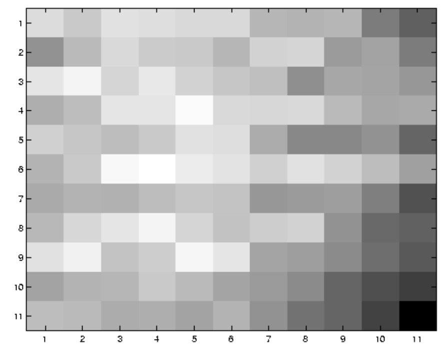
Edge example

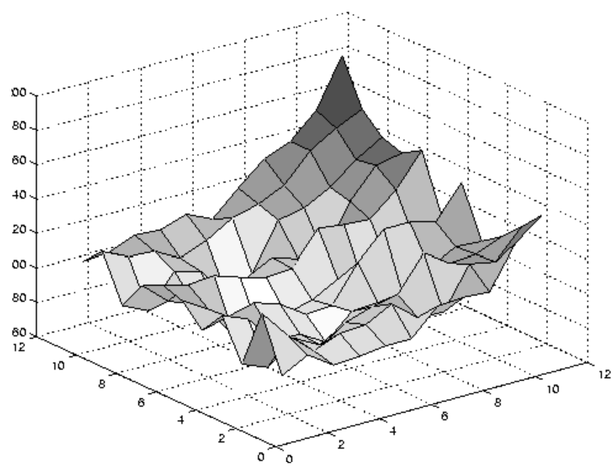




- large gradients, all the same
- large λ_1, small λ_2

Low texture example

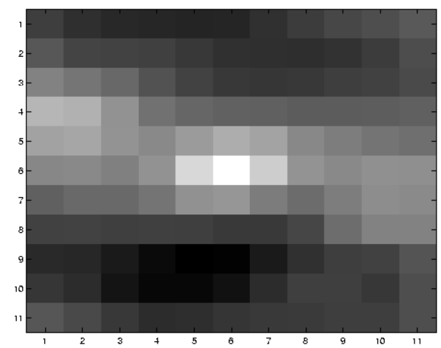


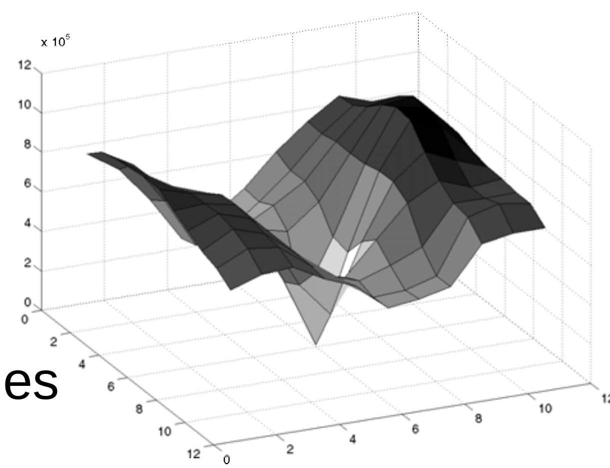


- gradients have small magnitude
- small λ_1, small λ_2

(Seitz)

High textured example





- gradients are different, large magnitudes
- large λ_1, large λ_2

Errors in Lucas-Kanade

- What are the potential causes of errors in this procedure?
  - Suppose $A^T A$ is easily invertible
  - Suppose there is not much noise in the image
- Our assumptions are violated when
  - Brightness constancy is **not** satisfied
  - The motion is **not** small
  - A point does **not** move like its neighbors
    - window size is too large
    - (what is the ideal window size?)