Topic 1 - Features

Jesse Hoey
School of Computing
University of Dundee

January 22, 2009

Scale Space

- at what scale are objects to be analysed?
- images contain information at many scales

Scale Space

- at what scale are objects to be analysed?
- images contain information at many scales
- best strategy: represent information at many scales

Gaussian Pyramid

- Pyramid of successively smaller images
- Smooth first,
- Interpolate and Downsample.
- Each Image gives image information at a particular scale

2D Convolution - Continuous Math

$$\begin{aligned} f(x,y) &= h(x,y) * g(x,y) \\ &= (f * g)(x,y) \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(x',y')g(x-x',y-y')dx'dy' \end{aligned}$$

Low Pass Filtering - Gaussian

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

Is Separable:

$$\begin{aligned} h(x,y) &= e^{-\frac{1}{2}\frac{x^2}{\sigma^2}+\frac{y^2}{\sigma^2}} \\ &= e^{-\frac{1}{2}\frac{x^2}{\sigma^2}} e^{-\frac{1}{2}\frac{y^2}{\sigma^2}} \end{aligned}$$

Low Pass Filtering - Gaussian

Advantages of Gaussian Kernel

- Separable
- Rotationally symmetric
- Smoothly falls to zero at edges
- Fourier Transform is a Gaussian

Low Pass Filtering - Gaussian

normal

5x5 mean filter

5x5 Gaussian filter

Transformation Equations

Affine transformation:

$$x' = a_0 x + a_1 y + a_2$$
$$y' = b_0 x + b_1 y + b_2$$

As an array equation

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a_0 & a_1 \\ b_0 & b_1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} a_2 \\ b_2 \end{bmatrix}$$

Or, using *homogeneous coordinates*

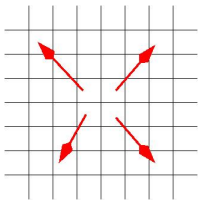
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_0 & a_1 & a_2 \\ b_0 & b_1 & b_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Scaling

1. Equation: Scaling Matrix

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

2.



Scaling Issues

	Small→Large	Large→Small
Technique:	Add Pixels (oversample or copy)	Delete Pixels (subsample)
Problems:		
Solutions:		

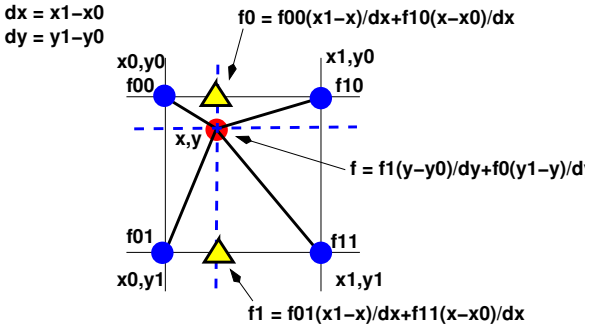
Scaling Issues

	Small→Large	Large→Small
Technique:	Add Pixels (oversample or copy)	Delete Pixels (subsample)
Problems:	Patchiness or “Blockiness”	Aliasing and missing data
Solutions:	smooth **after**	smooth **before**

Aliasing:
need to sample at **twice highest frequency**
if sample rate decreases **then** highest frequency *must decrease*, else aliasing

How to downsample

Bilinear Interpolation



What are interesting points at some scale?

Edges: compute Image Gradients

$$\vec{\nabla} f(x,y) = \frac{\partial f(x,y)}{\partial x}\vec{x} + \frac{\partial f(x,y)}{\partial y}\vec{y}$$

Simple gradient kernels (Prewitt)

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

Sobel Kernels (give more weight to on-axis pixels)

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Image Gradients

image Sobel x Sobel y

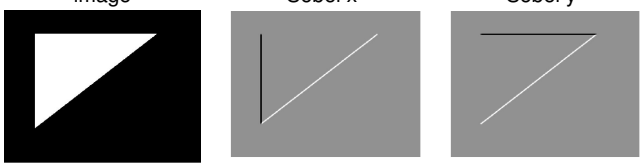
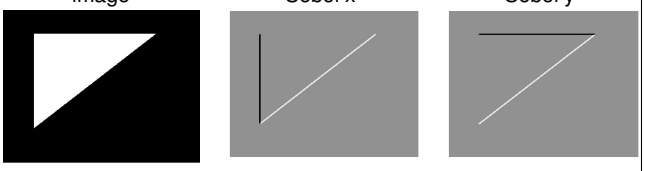
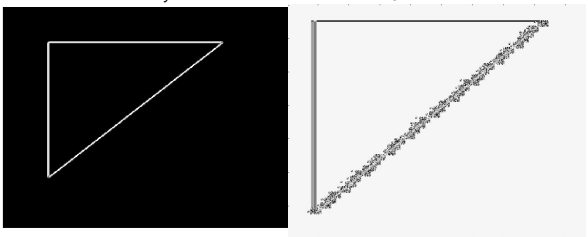


Image Gradients

image Sobel x Sobel y



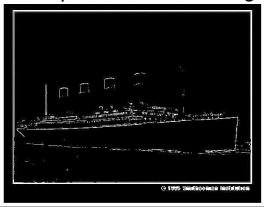
Magnitude: $g_x^2 + g_y^2$ phase $atan(g_y/g_x)$



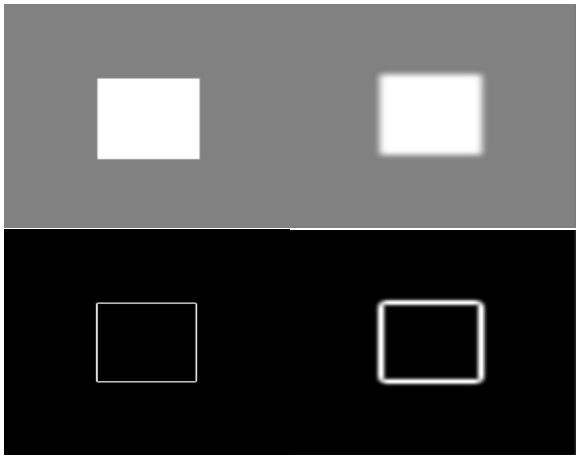
From Gradients to Edges

image and gradient magnitude:

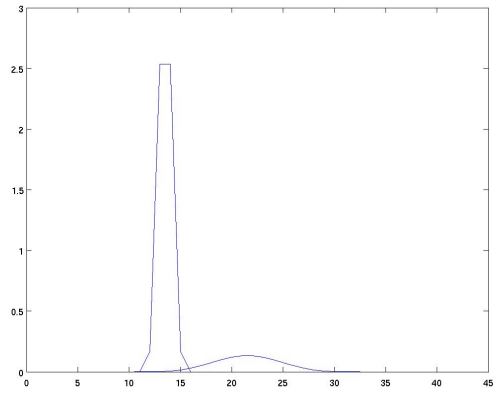
last step: threshold the magnitude.



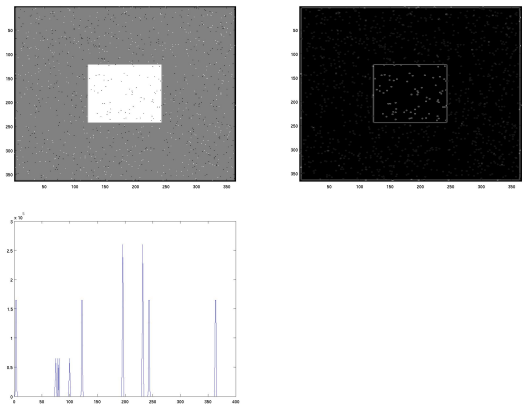
From Gradients to Edges



From Gradients to Edges



From Gradients to Edges



From Gradients to Edges

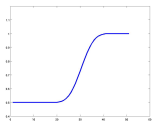
Problems with simple thresholding

- Sometimes edges are diffuse
- Noise can produce large gradient magnitudes
- Entire ridge (not maximum point) is detected

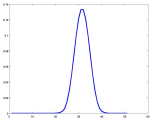
Need: Second order derivative
Second derivative allows us to find *maximum* or *minimum* values of gradient

From Gradients to Edges

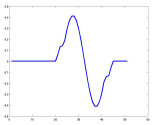
original .



first deriv



second deriv



Laplacian

$$\nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial^2 x} + \frac{\partial^2 f(x,y)}{\partial^2 y}$$

Laplacian

$$\nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial^2 x} + \frac{\partial^2 f(x,y)}{\partial^2 y}$$

but recall that gradient and convolution are Linear operators, so

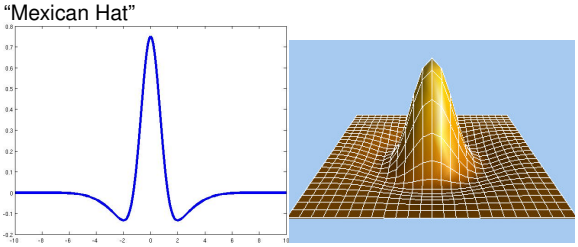
$$\nabla^2(\mathcal{N} * f(x,y)) = (\nabla^2 \mathcal{N}) * f(x,y)$$

taking Laplacian of a smoothed image = smoothing an image with a Laplacian of Gaussian

Laplacian of Gaussian

$$\nabla^2 \mathcal{N}(x;\sigma) = \left(\frac{x^2-\sigma^2}{\sigma^4}\right) e^{-x^2/2\sigma^2}$$

"Mexican Hat"



Scale Space

scale-space representation $L(x;\sigma)$ of $f(x)$ is

$$L(x;\sigma) = f(x) * \mathcal{N}(x;\sigma) = \int_y f(x-y)\mathcal{N}(y;\sigma)dy$$

where $\mathcal{N}(y;\sigma)$ is Gaussian with variance σ: the *scale parameter*.
First derivative of $L(x;\sigma)$ at scale σ is then

$$\frac{\partial}{\partial x} L(x;\sigma) = \left(\frac{\partial}{\partial x}\mathcal{N}(x;\sigma)\right) * f(x)$$

Second derivtive of $L(x;\sigma)$ at scale σ is then

$$\frac{\partial^2}{\partial x^2} L(x;\sigma) = \left(\frac{\partial^2}{\partial x^2}\mathcal{N}(x;\sigma)\right) * f(x)$$

Zero crossings of first or second derivative: local minima at each scale

Scale Space: Examples

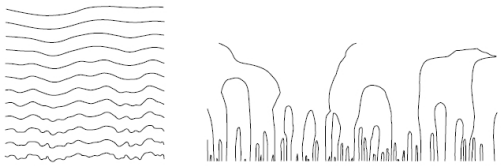


Figure 2: (a) The main idea of a scale-space representation is to generate a one-parameter family of derived signals in which the fine-scale information is successively suppressed. This figure shows a signal which has been successively smoothed by convolution with Gaussian kernels of increasing width. (b) Since new zero-crossings cannot be created by the diffusion equation in the one-dimensional case, the trajectories of zero-crossings in scale-space (here, zero-crossings of the second derivative) form paths across scales that are never closed from below.

Scale Space: Examples

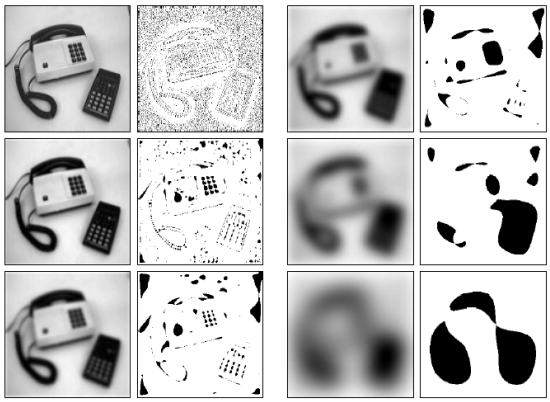


Figure 3: Different levels in the scale-space representation of a two-dimensional image at scale levels $t = 0, 2, 8, 32, 128$ and 512 together with grey-level blobs indicating local minima at each scale.

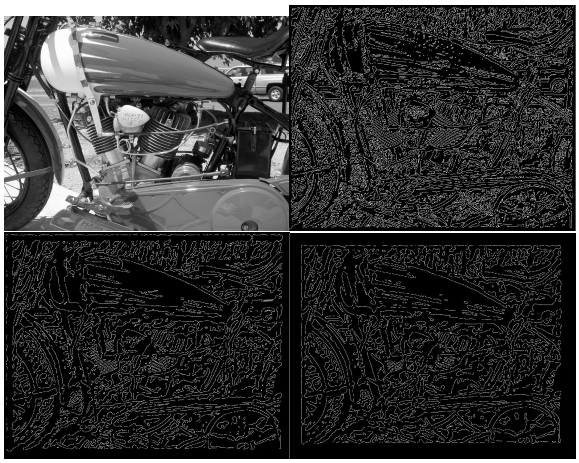
Laplacian of Gaussian

Approximation to Laplacian of Gaussian is Difference of Gaussians

$$e^{-r^2/2\sigma_1^2} - e^{-r^2/2\sigma_2^2}$$

where $\sigma_1 < \sigma_2$
σ_1, σ_2 control the scale
Therefore, can combine Scale-space construction (Gaussian Pyramid) with second-order derivative analysis by subtracting levels of Gaussian Pyramid

Laplacian of Gaussian



Scale Space: biological vision

Receptive field profiles: superpositions of Gaussian derivatives

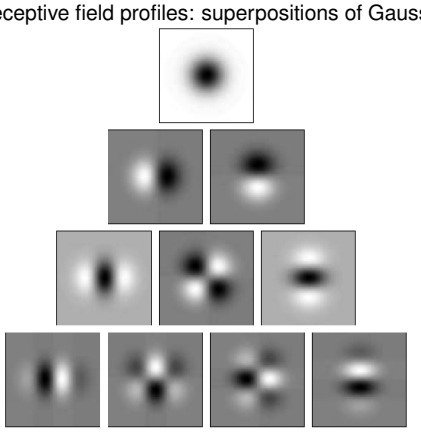


Figure 4: Gaussian derivative kernels up to order four the two-dimensional case.

Review

1. Interesting stuff at multiple scales in an image
2. Construct **Gaussian Pyramid**
   - 2.1 successive levels are smaller images, represent larger scales
   - 2.2 smooth with Gaussian kernel (low-pass filter) before downsampling to avoid **aliasing**
   - 2.3 downsample using bilinear interpolation
3. Compute Laplacian of Gaussian at each level (or subtract levels if properly done)
   - 3.1 second derivative finds center of edges
   - 3.2 Gaussian filter smooths away noise
4. Find zero-crossings: interesting points
5. But ... edges are difficult to localize

From Edges to Corners

- Edge cannot be localised exactly
- Corner can be easily distinguished from its neighbors: good feature?
- Moravec: Simple Corner detection
- Harris: based on Moravec, but more complicated

Moravec Corner Detector

- Look for points with low *self-similarity*
- Test each image patch to see if it looks the same as neighboring patches
- Use sum of squared differences (SSD) between patches

$$\sum_{\text{pixels}} (x_i - y_i)^2$$

- Uniform intensity: patches look similar, SSD is low
- Edge pixel: patches look similar in direction parallel to edge, different in direction orthogonal to edge
- Corner pixel: patches look different in all directions
- Problem?

Moravec Corner Detector

- Look for points with low *self-similarity*
- Test each image patch to see if it looks the same as neighboring patches
- Use sum of squared differences (SSD) between patches

$$\sum_{\text{pixels}} (x_i - y_i)^2$$

- Uniform intensity: patches look similar, SSD is low
- Edge pixel: patches look similar in direction parallel to edge, different in direction orthogonal to edge
- Corner pixel: patches look different in all directions
- Problem? Non-isotropic: edge not in direction of neighbours .. interest point

Harris Corner Detector

- deals with isotropy of Moravec
- Basic idea: orient Moravec detector to the gradient direction

Auto-correlation function

$$c(x,y) = \sum_i [I(x_i, y_i) - I(x_i + \Delta x, y_i + \Delta y)]^2$$

Taylor expand the shifted image

$$I(x_i + \Delta x, y_i + \Delta y) \approx I(x_i, y_i) + \begin{bmatrix} I_x(x_i, y_i) & I_y(x_i, y_i) \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

substitute into expression for $c(x,y)$

$$c(x,y) \approx \begin{bmatrix} \Delta x \Delta y \end{bmatrix} \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_y I_x & \sum I_y^2 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

Harris Corner Detector

$$c(x,y) \approx \begin{bmatrix} \Delta x \Delta y \end{bmatrix} \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_y I_x & \sum I_y^2 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

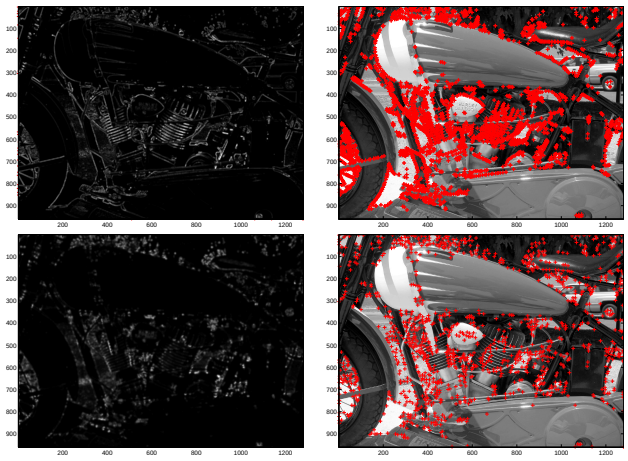
Look at "eigenvalues" of

$$C(x,y) = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_y I_x & \sum I_y^2 \end{bmatrix}$$

eigenvalues characterise the change in I with Δx and Δy, both in terms of direction and quantity

1. if both are small: constant intensity patch (change in no direction)
2. if one is large: edge (change in one direction only)
3. if both are large: corner (change in both directions)

Moravec vs. Harris Corner Detectors



Scale Invariant Feature

- Associate a *characteristic scale* for each feature: the scale at which this feature "lives"
- The scale at which the operator response is maximal
- scaled Laplacian of Gaussian best for scale detection:

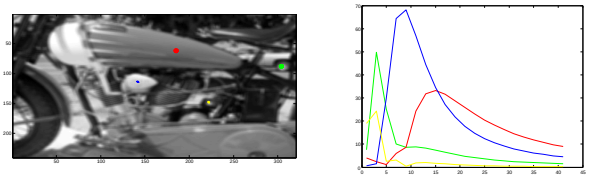
$$\sigma^2 \nabla \mathcal{N}$$

But can show that

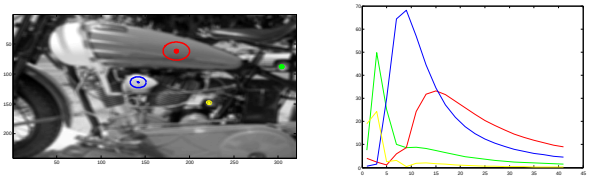
$$\mathcal{N}(x,y,k\sigma) - \mathcal{N}(x,y,\sigma) \approx (k-1)\sigma^2 \nabla \mathcal{N}$$

- Therefore, filter with a difference of Gaussians (DoG), and look for maximal response

Characteristic Scale



Characteristic Scale



Scale-Adapted Harris Corner Detector

- Look for maximal response of Harris detector at each scale
- For each maximal response point, find the characteristic scale

Affine Invariant Feature

- Pick a *characteristic orientation (rotation)* for each feature
- The orientation of the gradient
- Pick a *characteristic affine scale ratio* for each feature
- Based on second order gradient

In general, pick a characteristic *affine transformation* for each feature:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_0 & a_1 & a_2 \\ b_0 & b_1 & b_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Object Recognition

Given: an image of an object and a new image
We want to know: is the object in the new image?
How to do it:

1. Find features on the object.
   - 1.1 Each feature has a characteristic *affine transformation*
   - 1.2 Compute local *affine invariant* descriptor at each feature
2. Find features in new image (in same way)
3. **Match** new image features with object features by comparing *affine invariant* descriptors
4. These matches should succeed even if object has undergone an affine transformation