EE795: Computer Vision and Intelligent Systems

Spring 2012
TTh 17:30-18:45 FDH 204

Lecture 06
130207

http://www.ee.unlv.edu/~b1morris/ecg795/

Outline

- Review
  - Morphology
  - Connected Components
- Filtering for Detection
- Pyramids
- Wavelets

Morphological Image Processing

- Filtering done on binary images
  - Images with two values [0,1], [0, 255], [black,white]
  - Typically, this image will be obtained by thresholding
    - $g(x,y) = \begin{cases} 1 & f(x,y) > T \\ 0 & f(x,y) \le T \end{cases}$
- Morphology is concerned with the structure and shape
- In morphology, a binary image is convolved with a structuring element *s* and results in a binary image

- See Chapter 9 of Gonzalez and Woods for a more complete treatment
- Matlab
  - http://www.mathworks.com/help/images/pixel-values-and-image-statistics.html

Mathematical Morphology

- Tool for extracting image components that are useful in the representation and description of region shape
  - Boundaries, skeletons, convex hull, etc.
- The language of mathematical morphology is set theory
  - A set represents an object in an image

- This is often useful in video processing because of the simplicity of processing and emphasis on "objects"
  - Handy tool for "clean up" of a thresholded image

Morphological Operations

- Threshold operation
  - $\theta(f,t) = \begin{cases} 1 & f \geq t \\ 0 & \text{else} \end{cases}$
- Structuring element
  - s – e.g. 3 x 3 box filter (1's indicate included pixels in the mask)
  - S – number of "on" pixels in s
- Count of 1s in a structuring element
  - $c = f \otimes s$
  - Correlation (filter) raster scan procedure
- Basic morphological operations can be extended to grayscale images

- Dilation
  - $\text{dilate}(f,s) = \theta(c,1)$
  - Grows (thickens) 1 locations
- Erosion
  - $\text{erode}(f,s) = \theta(c,S)$
  - Shrink (thins) 1 locations
- Opening
  - $\text{open}(f,s) = \text{dilate}(\text{erode}(f,s),s)$
  - Generally smooth the contour of an object, breaks narrow isthmuses, and eliminates thin protrusions
- Closing
  - $\text{close}(f,s) = \text{erode}(\text{dilate}(f,s),s)$
  - Generally smooth the contour of an object, fuses narrow breaks/separations, eliminates small holes, and fills gaps in a contour

Morphology Example

Note: Black is "1" location

(a) (b) (c) (d) (e) (f)

Figure 3.21 Binary image morphology: (a) original image; (b) dilation; (c) erosion; (d) majority; (e) opening; (f) closing. The structuring element for all examples is a 5 × 5 square. The effects of majority are a subtle rounding of sharp corners. Opening fails to eliminate the dot, since it is not wide enough.

- Dilation - grows (thickens) 1 locations
- Erosion - shrink (thins) 1 locations
- Opening - generally smooth the contour of an object, breaks narrow isthmuses, and eliminates thin protrusions
- Closing - generally smooth the contour of an object, fuses narrow breaks/separations, eliminates small holes, and fills gaps in a contour

Binary Image Logic Operations

- Extension of basic logic operators
  - NOT, AND, OR, XOR

- Often use for “masking”

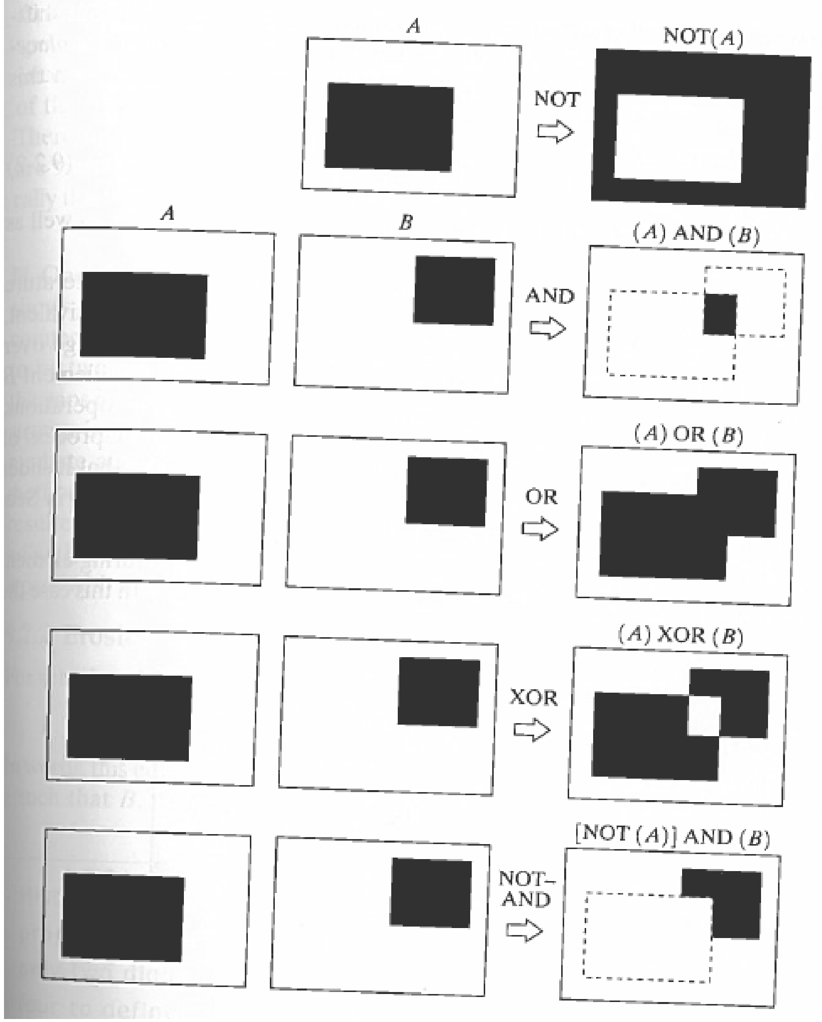


FIGURE 9.3 Some logic operations between binary images. Black represents binary 1s and white binary 0s in this example.

Structuring Elements

- The structuring element (SE) can be any shape
  - This is a mask of “on” pixels within a rectangular container
  - Typically, the SE is symmetric

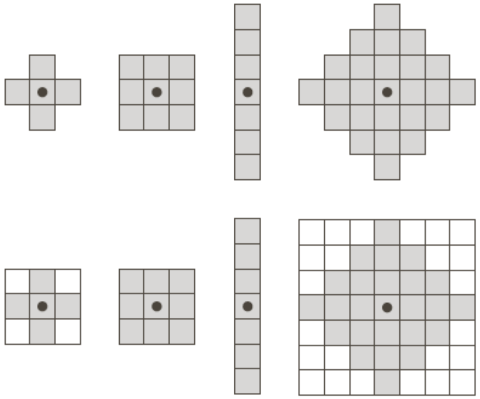


FIGURE 9.2 First row: Examples of structuring elements. Second row: Structuring elements converted to rectangular arrays. The dots denote the centers of the SEs.

Erosion

- Retain only pixels where the entire SE is overlapped
  - $A \ominus B = \{z | (B)_z \subseteq A\}$
    - A – image
    - B – SE
    - z – displacements (x,y locations)

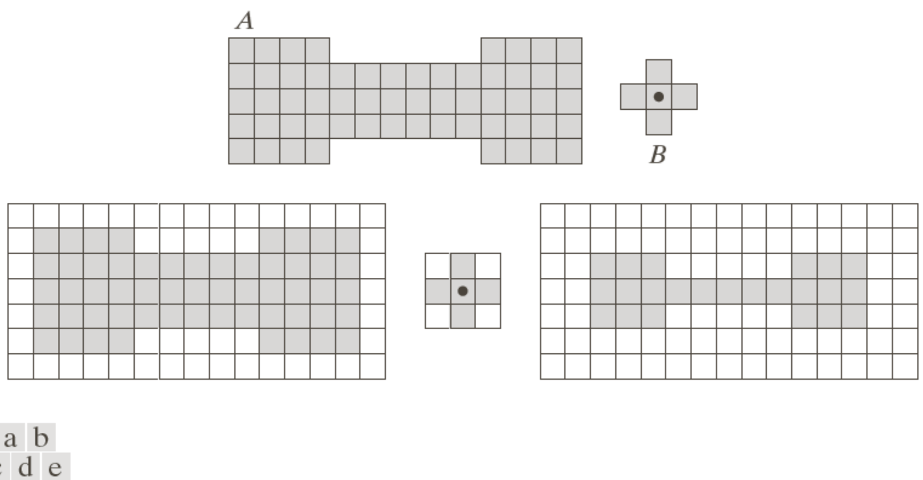


FIGURE 9.3 (a) A set (each shaded square is a member of the set). (b) A structuring element. (c) The set padded with background elements to form a rectangular array and provide a background border. (d) Structuring element as a rectangular array. (e) Set processed by the structuring element.

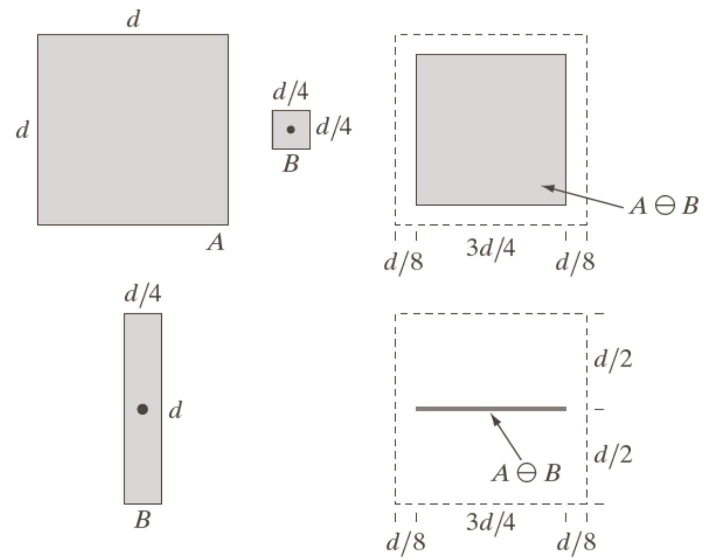


FIGURE 9.4 (a) Set A. (b) Square structuring element, B. (c) Erosion of A by B, shown shaded. (d) Elongated structuring element. (e) Erosion of A by B using this element. The dotted border in (c) and (e) is the boundary of set A, shown only for reference.

Dilation

- Output image is "on" anywhere the SE touches an "on" pixel
  - $A \oplus B = \{z | (B)_z \cap A \neq \emptyset\}$
    - A – image
    - B – SE
    - z – displacements (x,y locations)

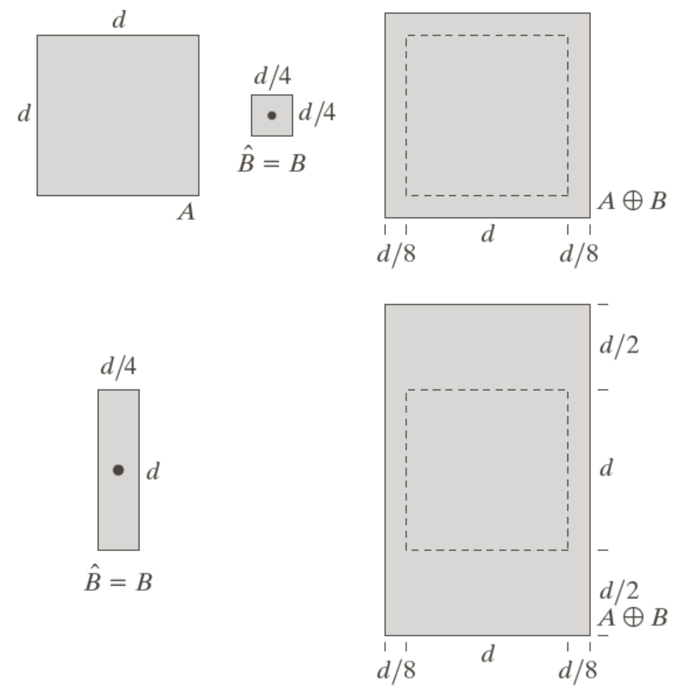


FIGURE 9.6 (a) Set A. (b) Square structuring element (the dot denotes the origin). (c) Dilation of A by B, shown shaded. (d) Elongated structuring element. (e) Dilation of A using this element. The dotted border in (c) and (e) is the boundary of set A, shown only for reference

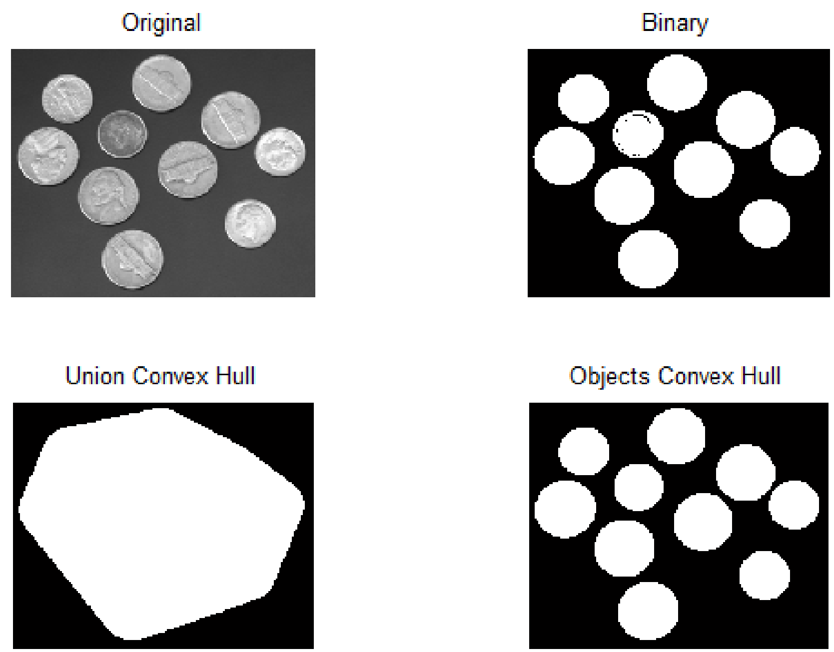
Other Morphological Operations

- Boundary extraction
  - $\beta(A) = A - (A \ominus B)$
  - Subtract erosion from original
  - Notice this is an edge extraction

a b

FIGURE 9.14 (a) A simple binary image, with 1s represented in white. (b) Result of using Eq. (9.5-1) with the structuring element in Fig. 9.13(b).

- Convex hull (H)
  - Smallest convex set that contains another set S
  - This is often done for a collection of 2D or 3D point
  - `bwconvhull.m`



Connected Components

- Semi-global image operation to provide consistent labels to similar regions
  - Based on adjacency concept
- Most efficient algorithms compute in two passes

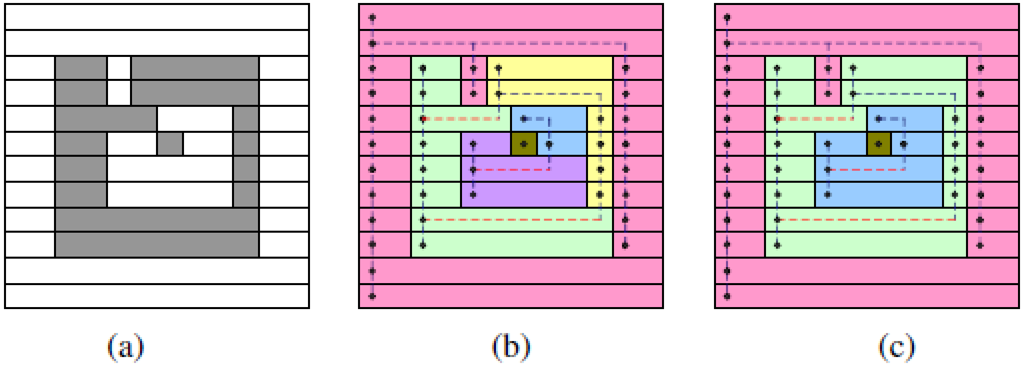


Figure 3.23 Connected component computation: (a) original grayscale image; (b) horizontal runs (nodes) connected by vertical (graph) edges (dashed blue)—runs are pseudocolored with unique colors inherited from parent nodes; (c) re-coloring after merging adjacent segments.

- More computational formulations (iterative) exist from morphology
  - $X_k = (X_{k-1} \oplus B) \cap A$

Set

Connected component

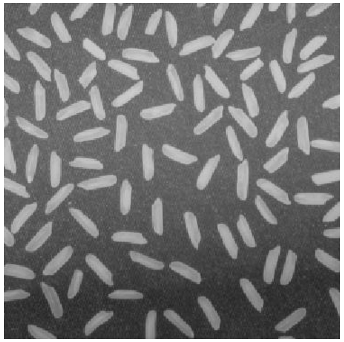
Structuring element

More Connected Components

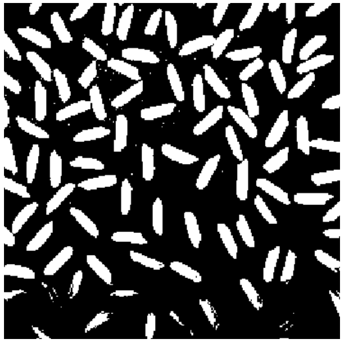
- Typically, only the "white" pixels will be considered objects
  - Dark pixels are background and do not get counted
- After labeling connected components, statistics from each region can be computed
  - Statistics describe the region – e.g. area, centroid, perimeter, etc.

- Matlab functions
  - `bwconncomp.m`, `labelmatrix.m` (`bwlabel.m`)- label image
  - `label2rgb.m` – color components for viewing
  - `regionprops.m` – calculate region statistics

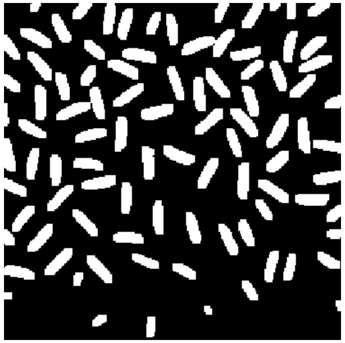
Connected Component Example



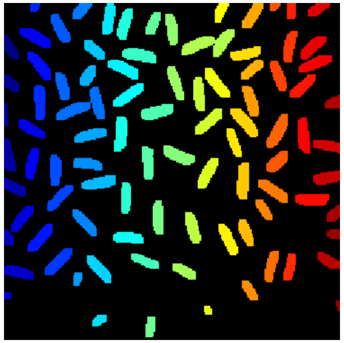
Grayscale image



Threshold image



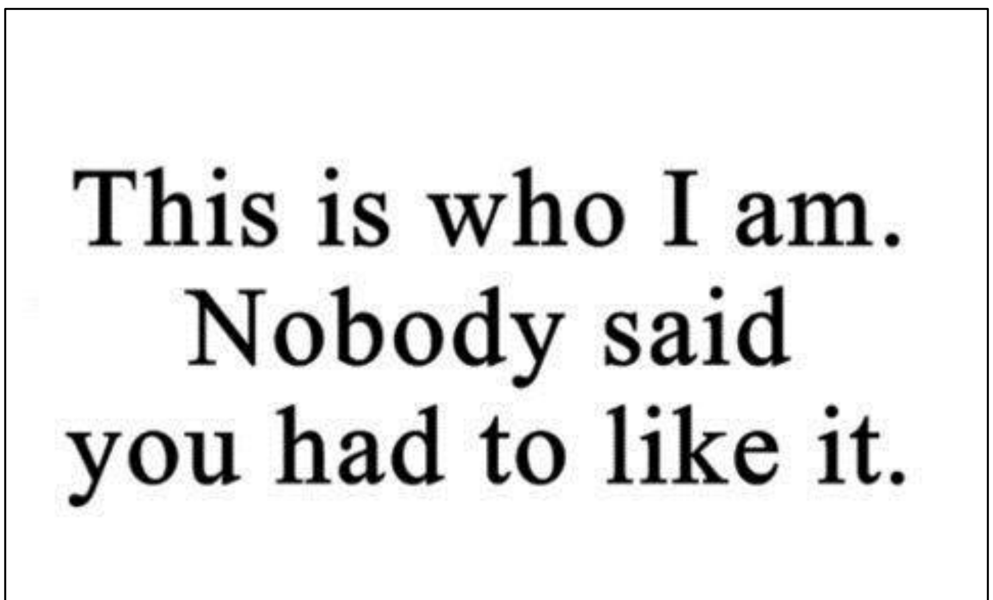
Opened Image



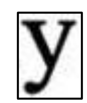
Labeled image – 91 grains of rice

Binary Filtering as Detection

- Filtering (correlation) can be used a simple object detector
  - Mask provides a search template
  - “Matched filter” – kernels look like the effects they are intended to find



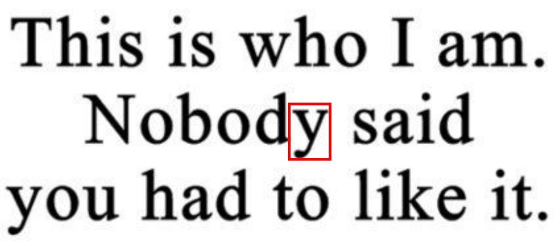
image



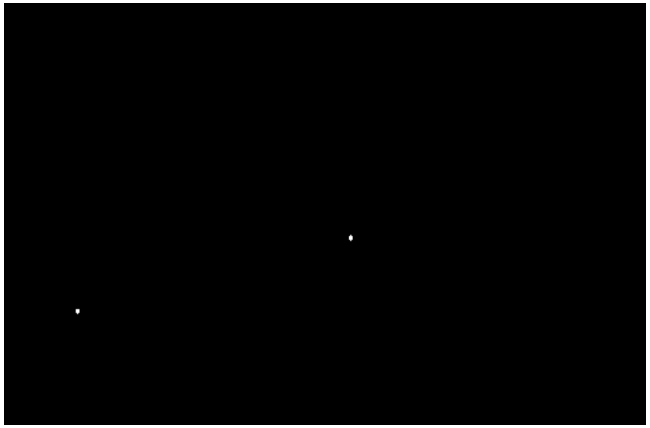
template

Correlation Masking

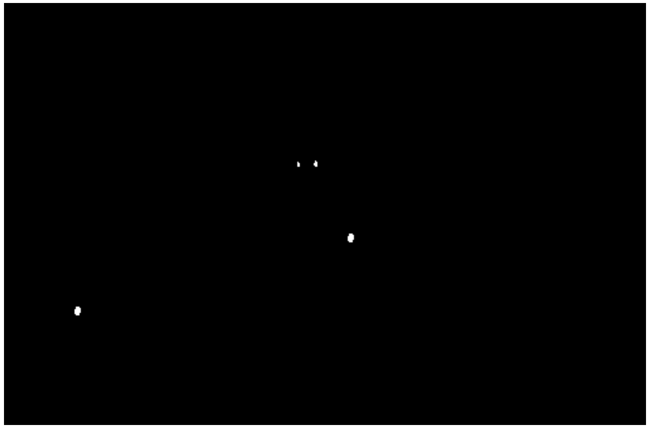
correlation



detected letter



0.9 max threshold

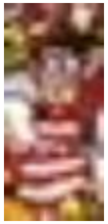


0.5 max threshold

Normalized Cross-Correlation

- Extension to intensity values
  - Handle variation in template and image brightness

scene

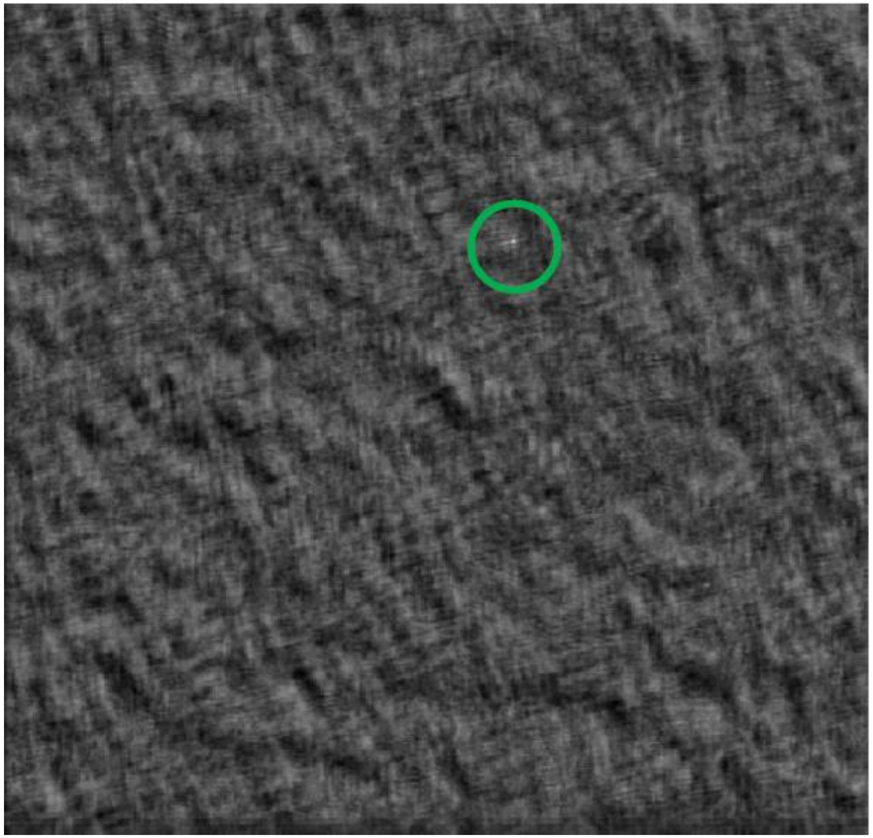


template

Adapted from
http://kurser.iha.dk/ee-ict-master/ticovi/

Where's Waldo

Detected template

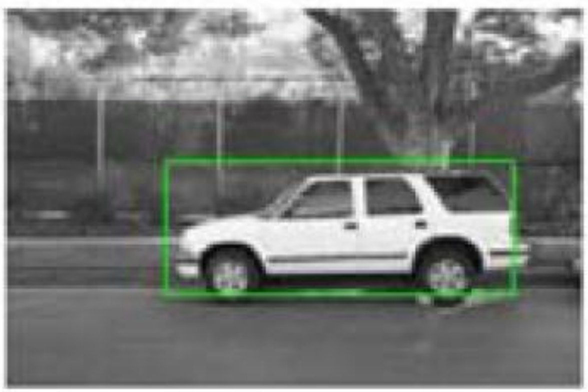


correlation map

Adapted from
http://kurser.iha.dk/ee-ict-master/ticovi/

Detection of Similar Objects

- Previous examples are detecting exactly what we want to find
  - Give the perfect template
- What happens with similar objects



Detected template

Template

- Works fine when scale, orientation, and general orientation are matched
- What to do with different sized objects, new scenes

Adapted from K. Grauman

Pyramids

- Image processing so far has input/output images of the same size
  - Will want to change the size of an image
    - Interpolation to make small image larger
    - Decimation to operate on a smaller image

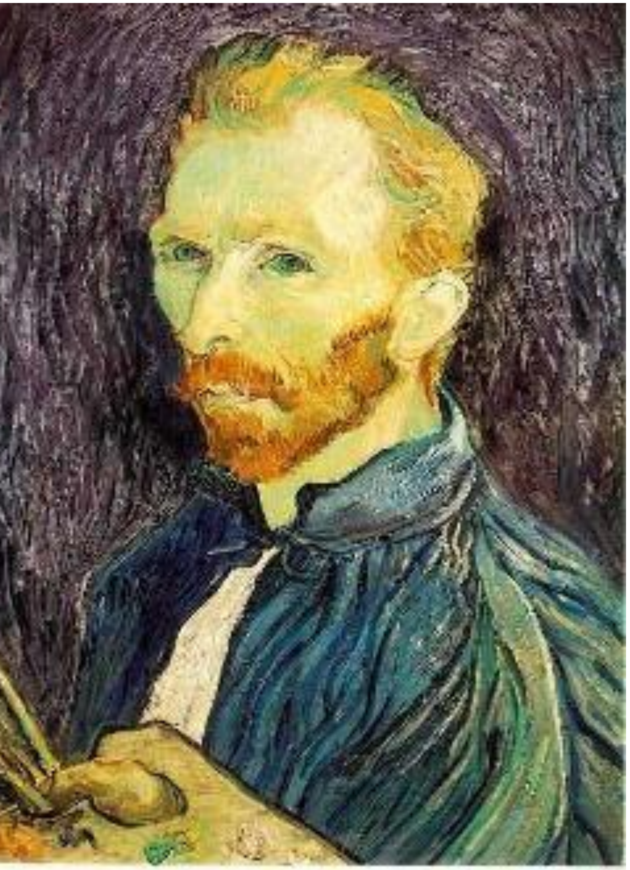
- Sometimes we may not know the appropriate resolution
  - What size cars are we looking for in the previous example?
  - Create a “pyramid” of images at different resolutions to do processing
    - Accelerates search process
    - Multi-scale representation and processing

Decimation

- Also known as downsampling
  - Decreases the size of an image by removing pixels
- Easiest form of decimation is subsampling by a factor of 2
  - Throw away pixels
  - What can we say about the quality of this?

Subsampling *without* pre-filtering

Adapted from S. Seitz



1/2

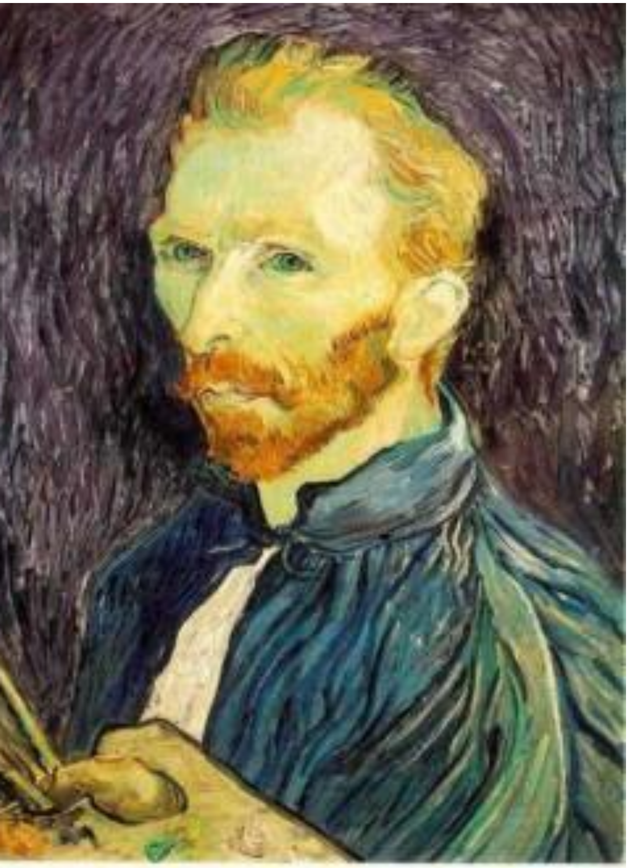
1/4 (2x zoom)

1/8 (4x zoom)

Subsampling with Gaussian pre-filtering

Adapted from S. Seitz

- Improved results by first smoothing



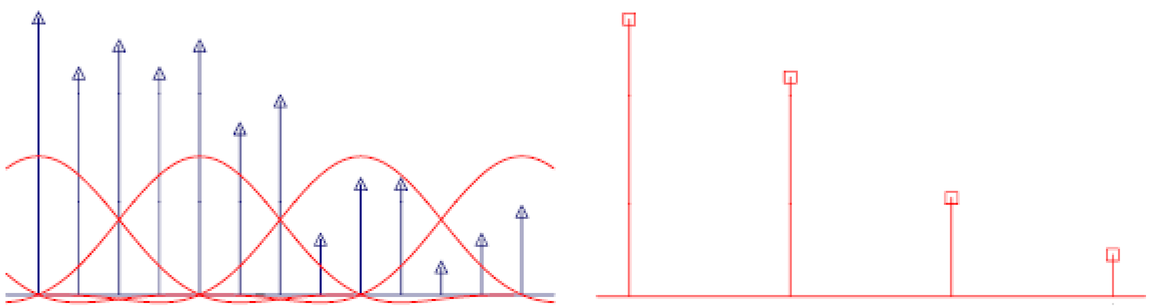
Gaussian 1/2

G 1/4

G 1/8

Smooth to Avoid Aliasing

- Low-pass smoothing filter avoids aliasing
- In practice, the convolution can be evaluated at a reduced rate
  - $g(i,j) = \sum_{k,l} f(k,l)h(ri-k, rj-l)$
    - r – the downsampling rate



(a) (b)

Figure 3.30 Signal decimation: (a) the original samples are (b) convolved with a low-pass filter before being downsampled.

Downsample Filters

- A number of filters can be used and have various performance
  - Amount of high-frequency removal
- Bilinear filter is the simplest
- Bicubic fits a 3rd degree polynomial to the neighborhood

$\|n\|$	Linear	Binomial	Cubic $a=-1$	Cubic $a=-0.5$	Windowed sinc	QMF-9	JPEG 2000
0	0.50	0.3750	0.5000	0.50000	0.4939	0.5638	0.6029
1	0.25	0.2500	0.3125	0.28125	0.2684	0.2932	0.2669
2		0.0625	0.0000	0.00000	0.0000	-0.0519	-0.0782
3			-0.0625	-0.03125	-0.0153	-0.0431	-0.0169
4					0.0000	0.0198	0.0267

Table 3.4 Filter coefficients for $2\times$ decimation. These filters are of odd length, are symmetric, and are normalized to have unit DC gain (sum up to 1). See Figure 3.31 for their associated frequency responses.

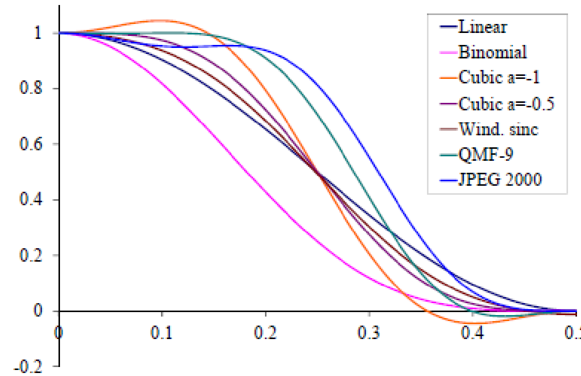
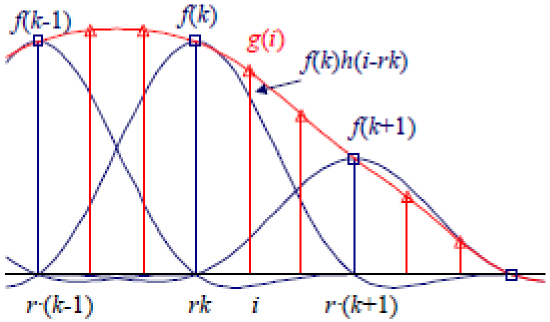


Figure 3.31 Frequency response for some $2\times$ decimation filters. The cubic $a = -1$ filter has the sharpest fall-off but also a bit of ringing; the wavelet analysis filters (QMF-9 and JPEG 2000), while useful for compression, have more aliasing.

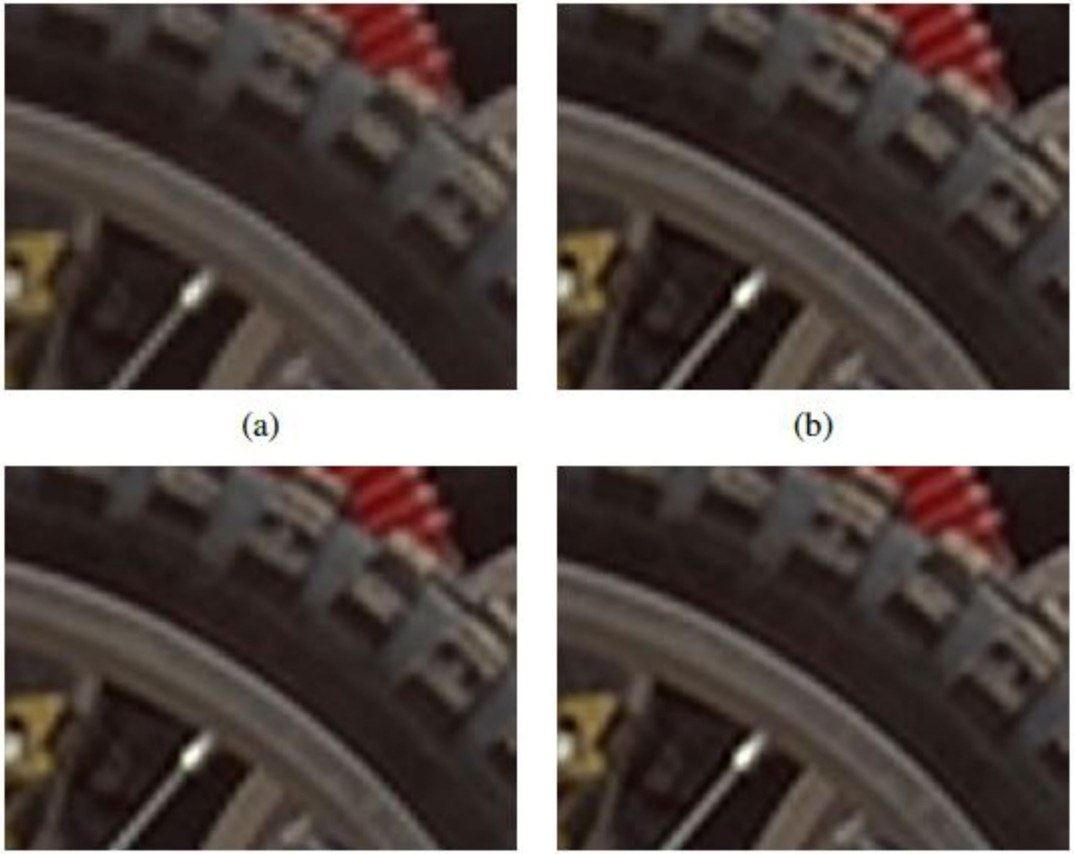
Interpolation

- Also known as upsampling
  - Increases the size of an image by inserting new pixels between existing ones
- This can be done with a modified convolution operation
  - $g(i,j) = \sum_{k,l} f(k,l)h(i - rk, j - rl)$
    - r – the upsampling rate
- Each new pixel is a weighted sum of samples



Interpolation Kernels

Notice some ringing
with bilinear



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

Figure 3.28 Two-dimensional image interpolation: (a) bilinear; (b) bicubic ($a = -1$); (c) bicubic ($a = -0.5$); (d) windowed sinc (nine taps).

Multi-Resolution Pyramids

- Accelerates coarse-to-fine search algorithms
  - Faster search at lower resolution and higher resolution for better localization
- Detect objects at different scales
  - Use same template with different sized images = having different sized templates
- Perform multi-resolution blending operations

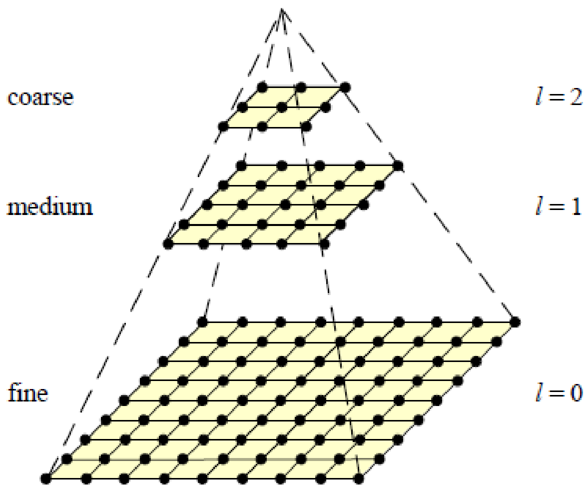
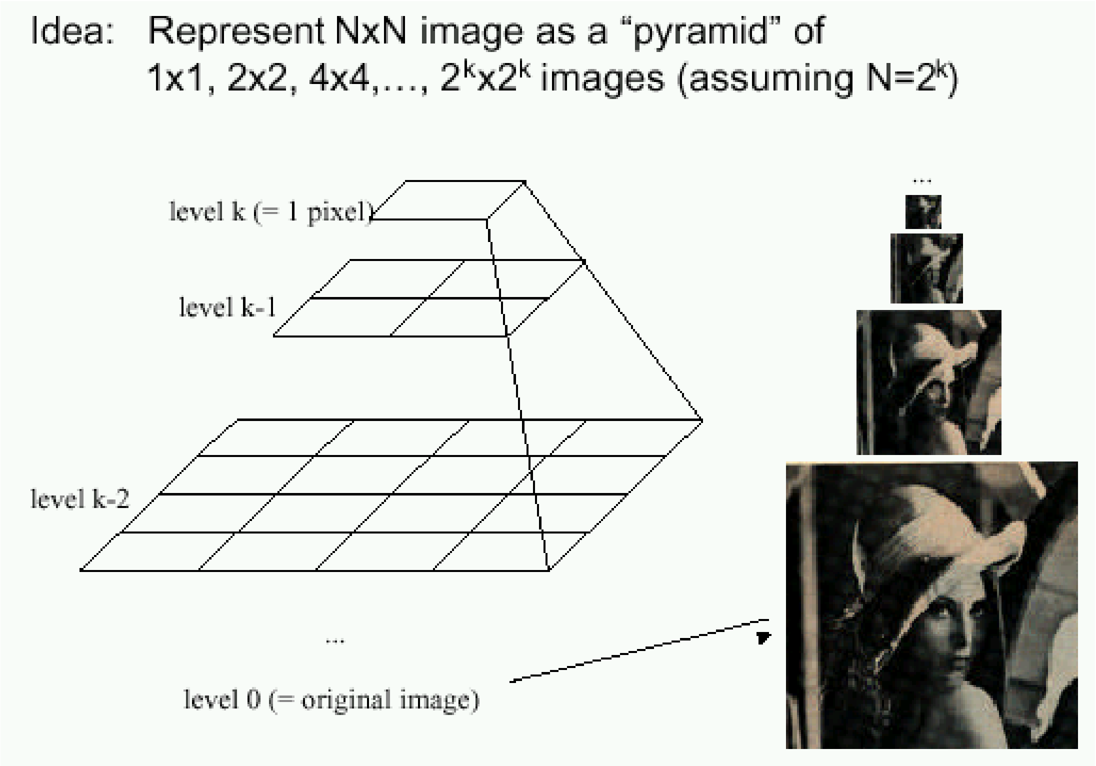


Figure 3.32 A traditional image pyramid: each level has half the resolution (width and height), and hence a quarter of the pixels, of its parent level.

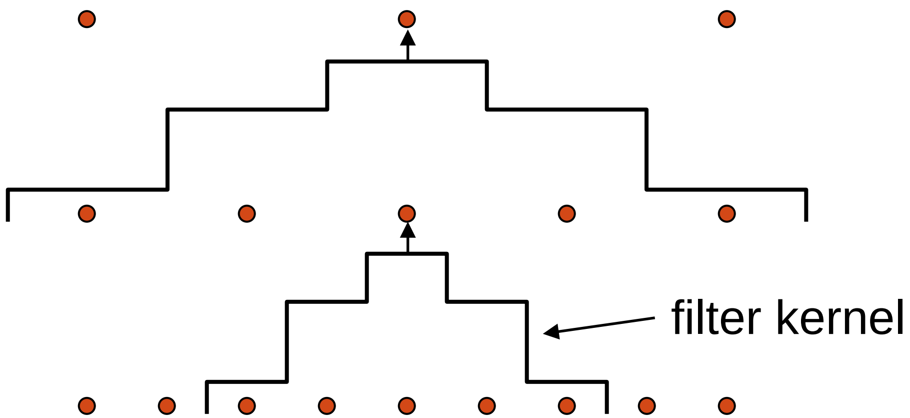
Laplacian (Gauassian) Pyramid

- [Burt and Adelson, 1983]
- Blur and subsample the original image by a factor of 2 at each level



Gaussian Pyramid

- Construction



- Repeat:
  - Filter
  - Subsample
- Until minimum resolution is reached
- The whole pyramid is only 4/3 the size of the original image
  - Each higher level is ¼ the size of lower level

Adapted from S. Seitz