More on template matching

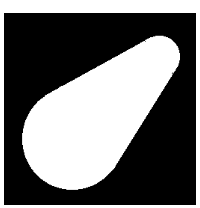
Shape based matching

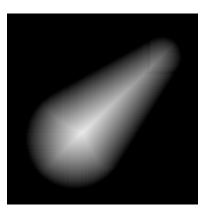
Chamfer matching

- Given:
  - binary image, B, of edge and local feature locations
  - binary "edge template", T, of shape we want to match
- Let D be an array in registration with B such that D(i,j) is the distance to the nearest "1" in B.
  - this array is called the **distance transform** of B
- Goal: Find placement of T in D that minimizes the sum, M, of the distance transform multiplied by the pixel values in T
  - if T is an exact match to B at location (i,j) then M(i,j) = 0
  - but if the edges in B are slightly displaced from their ideal locations in T, we still get a good match using the distance transform technique

Computing the distance transform

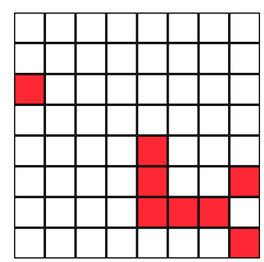
- Brute force, exact algorithm, is to scan B and find, for each "0", its closest "1" using the Euclidean distance.
  - expensive in time, and difficult to implement





Binary Image and its distance transform – suitable for medial axis representations

Distance transform example



0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7
2	2	2	3	0	1	2	3
3	3	3	1	0	1	2	0
4	4	2	1	0	0	0	1
5	3	2	1	1	1	1	0

f	f	f
f		b
b	b	b

Distance transform example

0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7
2	2	2	3	0	1	2	3
3	3	3	1	0	1	2	0
4	4	2	1	0	0	0	1
5	3	2	1	1	1	1	0

2	2	3	4	4	4	4	4
1	2	2	3	3	3	3	3
0	1	2	2	2	2	2	3
1	1	2	1	1	1	2	2
2	2	2	1	0	1	1	1
3	3	2	1	0	1	1	0
4	3	2	1	0	0	0	1
4	3	2	1	1	1	1	0

D(i,j) = min[D(i,j), D(i,j+1)+1, D(I+1, j-1)+1, D(i+1,j)+1, D(i+1,j+1)+1]

f	f	f
f		b
b	b	b

Chamfer matching

- Chamfer matching is convolution of a binary edge template with the distance transform
  - for any placement of the template over the image, it sums up the distance transform values for all pixels that are "1's" (edges) in the template
  - if, at some position in the image, all of the edges in the template coincide with edges in the image (which are the points at which the distance transform is zero), then we have a perfect match with a match score of 0.

Example

2	2	3	4	4	4	4	4
1	2	2	3	3	3	3	3
0	1	2	2	2	2	2	3
1	1	2	1	1	1	2	2
2	2	2	1	0	1	1	1
3	3	2	1	0	1	1	0
4	3	2	1	0	0	0	1
4	3	2	1	1	1	1	0

O		
	O	O
O		

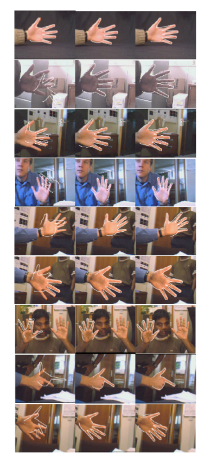
Template

2	2	3	4	4	4	4	4
1	2	2	3	3	3	3	3
0	1	2	2	2	2	2	3
1	1	2	1	1	1	2	2
2	2	2	1	0	1	1	1
3	3	2	1	0	1	1	0
4	3	2	1	0	0	0	1
4	3	2	1	1	1	1	0

Match score is $$\sum_{i=1}^{n}\sum_{j=1}^{n} T(i,j)D(i+k,j+l)$$

Chamfer Matching

From Shape Context and Chamfer Matching In Cluttered Scenes. A. Thayananthan, B. Stenger, P. Torr and R. Cippola



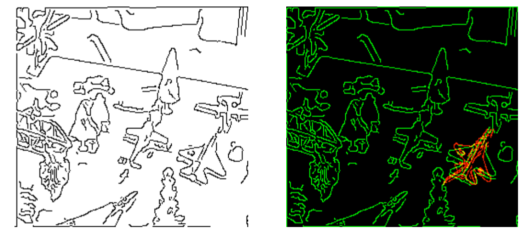
Hausdorff distance matching

- Let M be an nxn binary template and N an nxn binary image we want to compare to that template
- H(M,N)) = max(h(M, N), h(N, M)) where

$$h(A,B) = \max_{a \in A} \min_{b \in B} \|a - b\|$$

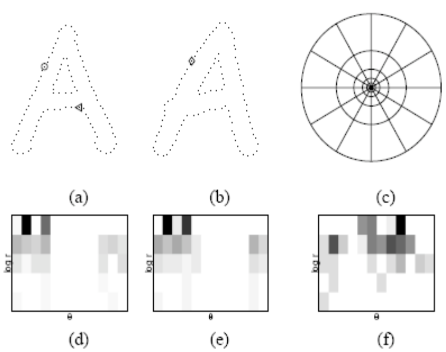
  - || || is a distance function like the Euclidean distance function
- h(A,B) is called the directed Hausdorff distance.
  - ranks each point in A based on closeness to a point in B
  - most mis-matched point is measure of match
  - if h(A,B) = e, then all points in A must be within distance e of B.
  - generally, h(A,B) <> h(B,A)
  - easy to compute Hausdorff distances from distance transform

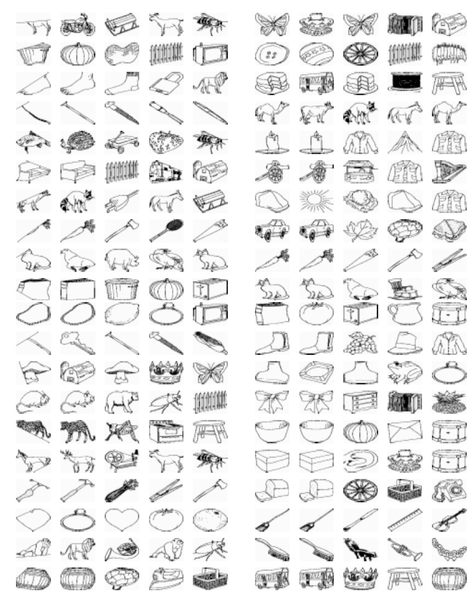
Haussdorf Distance Matching



Shape Context

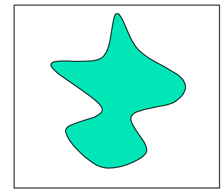
- From Shape Matching and Object Recognition using Shape Context by Belongie, Malik, Puzicha. IEEE PAMI (24), 2002





Alternative shape properties

- Consider blobs - characterized by connected componets of contours



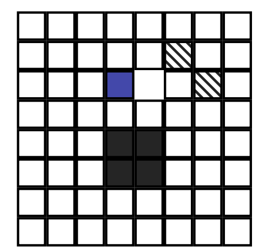
- Our goal is to recognize each connected component as one of a set of known objects
  - letters of the alphabet
  - good potatoes versus bad potatoes
- We need to associate measurements, or properties, with each connected component that we can compare against expected properties of different object types.

Properties

- Area
- Perimeter
- Compactness: P^2/A
  - smallest for a circle: $4\pi^2r^2/\pi r^2 = 4\pi$
  - higher for elongated objects
- Properties of holes
  - number of holes
  - their sizes, compactness, etc.

How do we compute the perimeter of a connected component?

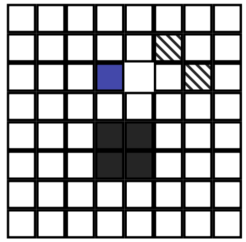
1. Count the number of pixels in the component adjacent to 0's
   - perimeter of black square would be 1
   - but perimeter of gray square, which has 4x the area, would be 4
   - but perimeter should go up as sqrt of area
2. Count the number of 0's adjacent to the component
   - works for the black and gray squares, but fails for the red dumbbell

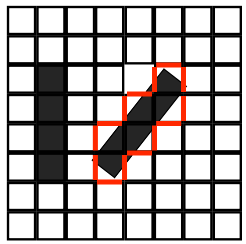


How do we compute the perimeter of a connected component?

3) Count the number of sides of pixels in the component adjacent to 0's
- these are the **cracks** between the pixels
- clockwise traversal of these cracks is called a crack code
- perimeter of black is 4, gray is 8 and red is 8

- What effect does rotation have on the value of a perimeter of the digitization of a simple shape?
  - rotation can lead to large changes in the perimeter and the area!

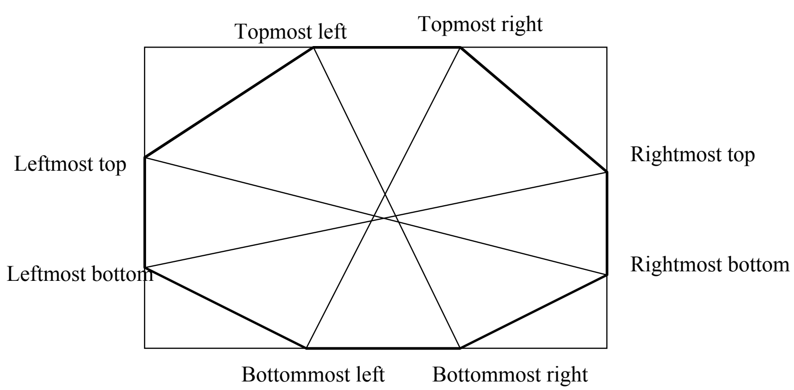




Perimeter computation (cont.)

- We can give different weights to boundary pixels
  - 1 – vertical and horizontal pairs
  - $2^{1/2}$ – diagonal pairs
- The boundary can be approximated by a polygon line (or splines) and its length could be used
- It matters most for small (low resolution objects)

Bounding Box and Extremal Points



Other features

- Convex hull:
  - Create a monotone polygon from the boundary (leftmost and rightmost points in each row)
  - Connect the extremal points by removing all concavities (can be done by examining triples of boundary points)
- Minimal bounding box from the convex hull
- Deficits of convexity

A better (and universal) set of features

- An "ideal" set of features should be independent of
  - the position of the connected component
  - the orientation of the connected component
  - the size of the connected component
    - ignoring the fact that as we "zoom in" on a shape we tend to see more detail
- These problems are solved by features called moments

Central moments

- Let S be a connected component in a binary image
  - generally, S can be any subset of pixels, but for our application the subsets of interest are the connected components
- The (j,k)'th moment of S is defined to be

$$M_{jk}(S) = \sum_{(x,y)\in S} x^j y^k$$

Central moments

- M_{00} = the area of the connected component

$$M_{00}(S) = \sum_{(x,y)\in S} x^0 y^0 = \sum_{(x,y)\in S} 1 = |S|$$

- The center of gravity of S can be expressed as

$$\bar{x} = \frac{M_{10}(S)}{M_{00}(S)} = \frac{\sum x}{|S|}$$

$$\bar{y} = \frac{M_{01}(S)}{M_{00}(S)} = \frac{\sum y}{|S|}$$

Central moments

- Using the center of gravity, we can define the central (j,k)'th moment of S as

$$\mu_{jk} = \sum (x-\bar{x})^j (y-\bar{y})^k$$

- If the component S is translated, this means that we have added some numbers (a,b) to the coordinates of each pixel in S
  - for example, if a = 0 and b = -1, then we have shifted the component up one pixel

Central moments

- Central moments are not affected by translations of S. Let S'={(x', y'):x'=x+a, y'=y+b, (x,y) in S}
  - The center of gravity of S' is the c.o.g. of S shifted by (a,b)

$$\bar{x}(S') = \frac{\sum x'}{|S'|} = \frac{\sum (x+a)}{|S|} = \frac{\sum x}{|S|} + \frac{\sum a}{|S|} = \bar{x} + a$$

  - The central moments of S' are the same as those of S

$$\mu_{jk}(S') = \sum (x' - \bar{x}(S'))^j (y' - \bar{y}(S'))^k$$
$$= \sum (x + a - [\bar{x}(S) + a])^j (y + b - [\bar{y}(S) + b])^k$$
$$= \sum (x - \bar{x})^j (y - \bar{y})^k = \mu_{jk}(S)$$

Central moments

- The standard deviations of the x and y coordinates of S can also be obtained from central moments:

$$\sigma_x = \sqrt{\frac{\mu_{20}}{|S|}}$$
$$\sigma_y = \sqrt{\frac{\mu_{02}}{|S|}}$$

- We can then create a set of normalized coordinates of S that we can use to generate moments unchanged by translation and scale changes

$$\tilde{x} = \frac{x - \bar{x}}{\sigma_x} \qquad \tilde{y} = \frac{y - \bar{y}}{\sigma_y}$$

Normalized central moments

- The means of these new variables are 0, and their standard deviations are 1. If we define the normalized moments; m_{jk} as follows

$$m_{jk} = \frac{\sum \tilde{x}^j \tilde{y}^k}{M_{00}}$$

- then these moments are not changed by any scaling or translation of S
- Let S* = {(x*,y*): x* = ax + b, y* = ay + c, (x,y) in S}
  - if b and c are 0, then we have scaled S by a
  - if a is 0, then we have translated S by (b,c)

Normalized central moments

$$m_{jk}(S^*) = \frac{\sum (\frac{x^* - \overline{x(S^*)}}{\sigma_x(S^*)})^j (\frac{y^* - \overline{y(S^*)}}{\sigma_y(S^*)})^k}{|S|}$$
$$= \frac{\sum (\frac{a^j (x - \bar{x}(S))^j}{a^j \sigma_x{}^j(S)})(\frac{a^k (y - \bar{y}(S))^k}{a^k \sigma_y{}^k(S)})}{|S|}$$
$$= m_{jk}(S)$$

- Details of the proof are simple.

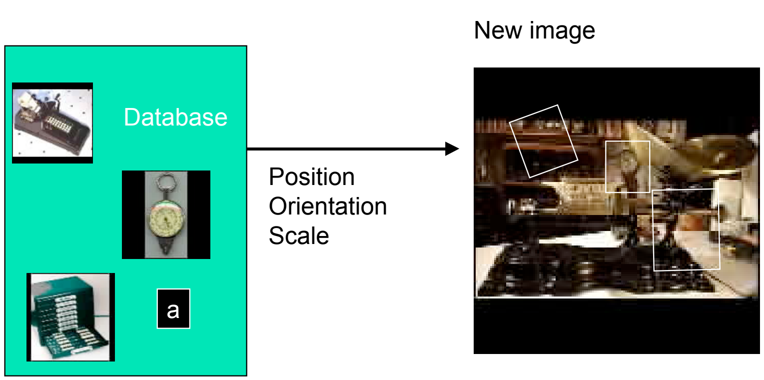
Shortcomings

- Object detection
  - thresholding will not extract intact objects in complex images
    - shading variations on object surfaces
    - texture
  - advanced segmentation methods
    - edge detection - locate boundaries between objects and background, between objects and objects
    - region analysis - find homogeneous regions; small combinations might correspond to objects.

Shortcomings

- Occlusion
  - What if one object is partially hidden by another?
    - properties of the partially obscured, or occluded, object will not match the properties of the class model
  - Correlation - directly compare image of the "ideal" objects against real images
    - in correct overlap position, matching score will be high
  - Represent objects as collection of local features such as corners of a rectangular shape
    - locate the local features in the image
    - find combinations of local features that are configured consistently with objects

Template matching



Reducing the cost of correlation matching

- A number of factors lead to large costs in correlation matching:
  - the image N is much larger than the template M, so we have to perform correlation matching of M against every nxn window of N
  - we might have **many** templates, M_i, that we have to compare against a given image N
    - face recognition - have a face template for **every** known face; this might easily be tens of thousands
    - character recognition - template for each character
  - we might not know the orientation of the template in the image
    - template might be rotated in the image N - example: someone tilts their head for a photograph
    - would then have to perform correlation of rotated versions of M against N

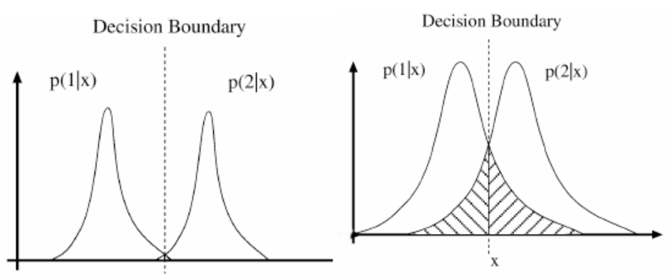
Reducing the cost of correlation matching

- A number of factors lead to large costs in correlation matching:
  - we might not know the scale, or size, of the template in the unknown image
    - the distance of the camera from the object might only be known approximately
  - would then have to perform correlation of scaled versions of M against N
  - Alternative to templates - scale/viewpoint invariant patches (e.g. SIFT features)

Recognition by finding patterns

- We have seen very simple template matching (under filters)
- Some objects behave like quite simple templates
  - Frontal faces
- Strategy:
  - Find image windows
  - Correct lighting
  - Pass them to a statistical test (a classifier) that accepts faces and rejects non-faces

Bayesian Decision Making



Histogram based classifiers

- Use a histogram to represent the class-conditional densities
  - (i.e. p(x|1), p(x|2), etc)
- Advantage: estimates become quite good with enough data!
- Disadvantage: Histogram becomes big with high dimension
  - but maybe we can assume feature independence?

Finding skin

- Skin has a very small range of (intensity independent) colours, and little texture
  - Compute an intensity-independent colour measure, check if colour is in this range, check if there is little texture (median filter)
  - See this as a classifier - we can set up the tests by hand, or learn them.
  - get class conditional densities (histograms), priors from data (counting)
- Classifier is
  - if $p(\text{skin}|\boldsymbol{x}) > \theta$, classify as skin
  - if $p(\text{skin}|\boldsymbol{x}) < \theta$, classify as not skin
  - if $p(\text{skin}|\boldsymbol{x}) = \theta$, choose classes uniformly and at random

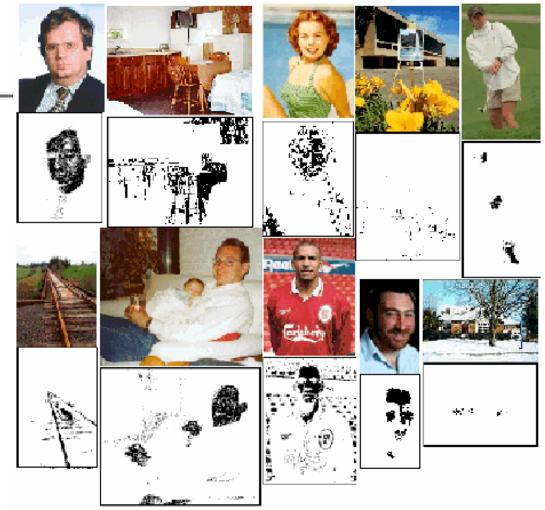


Figure from "Statistical color models with application to skin detection," M.J. Jones and J. Rehg, Proc. Computer Vision and Pattern Recognition, 1999 copyright 1999, IEEE

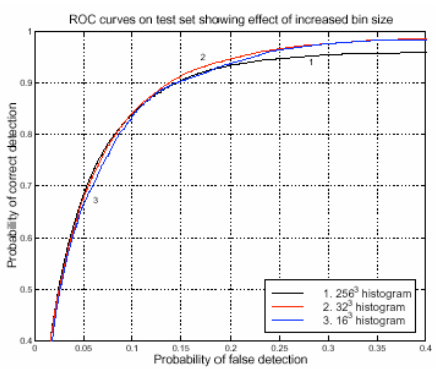


Figure from "Statistical color models with application to skin detection," M.J. Jones and J. Rehg, Proc. Computer Vision and Pattern Recognition, 1999 copyright 1999, IEEE

Reducing the cost of template matching

- Reducing the number of image windows that need to be compared against the database
  - find "objects" in the image
- Reducing the number of database objects that need to be matched against any window
  - index templates by features such as moments that are not changed by rotations and scale changes
  - measure moments of candidate windows
  - only match "similar" templates
- Reducing the number of operations needed to match a given template to the image

Reducing the cost of correlation matching

- Two basic techniques for reducing the number of operations associated with correlation
  - reduce the number of pixels in M and N
    - multi-resolution image representations
    - principal component or "feature selection" reductions
  - match a subset of M against a subset of N
    - random subsets
    - Boundary subsets

Multi-resolution correlation

- Multi-resolution template matching
  - reduce resolution of both template and image by creating an **image pyramid**
  - match small template against small image
  - identify locations of strong matches
  - expand the image and template, and match higher resolution template selectively to higher resolution image
  - iterate on higher and higher resolution images
- Issue:
  - how to choose detection thresholds at each level
    - too low will lead to too much cost
    - too high will miss match

Image pyramids

- Base of the pyramid, level 0, is the full resolution image - say 2^n x 2^n
- Level i of the pyramid is obtained from level i-1 as follows
  - partition level i-1 into non-overlapping 2^k x 2^k blocks
    - typically, k = 1 or 2
  - compute an average grey level in each of these blocks
    - unweighted average
    - Gaussian weighted average more typical
  - assign that average grey level to the corresponding level i pixel
- **For you to think about: How many pixels are there in an image pyramid having an nxn base and a reduction by neighborhoods of size 2^k x 2^k?**

Example

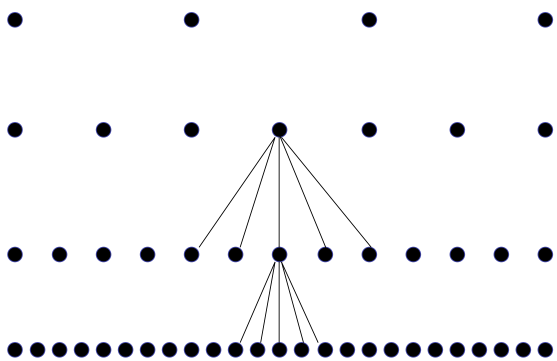
8	7	8	12				
13	12	7	6				
30	32	34	30				
32	26	28	33				

10	8		
30	31		

20	

- Representation of images at multiple levels of resolution
- Importance – at different resolutions different features look differently
- Used for localization properties, motion computation and matching, biological motivation

Pyramid construction



Pyramid construction/reconstruction

- Take an original image – convolve with a blurring
- Filter and subsample to get an image at lower resolution

Reduction – how is the signal at level l+1 related to level l

$$f^{l+1}[x,y] = \sum_{k=-\frac{w}{2}}^{\frac{w}{2}} \sum_{l=-\frac{w}{2}}^{\frac{w}{2}} g[k,l] f^{l}[2x-k, 2y-l]$$

Pyramid construction/reconstruction

Expansion – how to reconstruct the signal at level l given related to level l-1 – notation

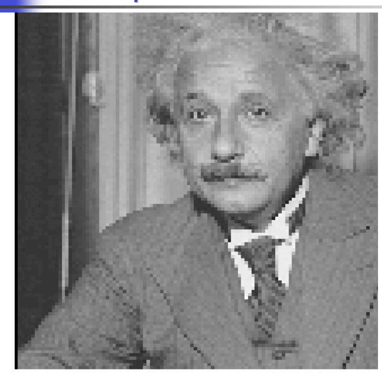
$$f^{l,k}[x,y]$$

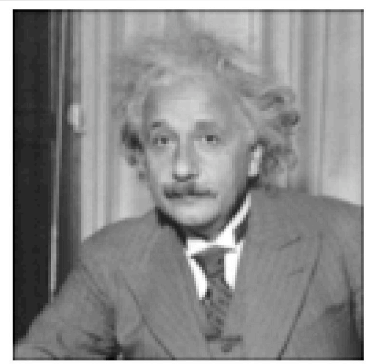
Signal at level l, expanded k - times

$$f^{l,k+1}[x,y] = \sum_{k=-\frac{w}{2}}^{\frac{w}{2}} \sum_{l=-\frac{w}{2}}^{\frac{w}{2}} c[k,l] f^{l}[(x-k)/2, (y-l)/2]$$

Idea - take the smaller signal fill every second entry
With zero and convolve with the blurring filter

"Drop" vs "Smooth and Drop"



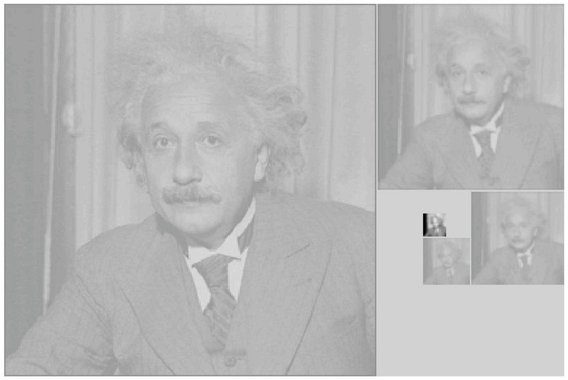


Drop every second pixel

Smooth and Drop every second pixel

Aliasing problems

Gaussian Pyramid

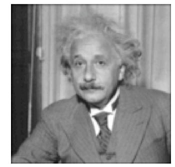


Consecutive smoothing and sub-sampling

Laplacian Pyramid

Idea

when we convolve and down-sample some information will get lost i.e. if we reverse the process we cannot get the original image back. Example:



Blurred image of Lower resolution

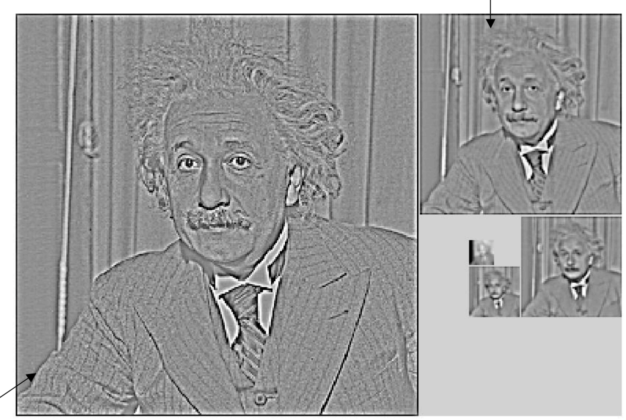
(original image - upsampled blurred image) they are not the same – fine details are lost

Laplacian pyramid

- Store the fine differences which get lost
- Each level of the Laplacian pyramid difference between two consecutive levels of gaussian pyramids

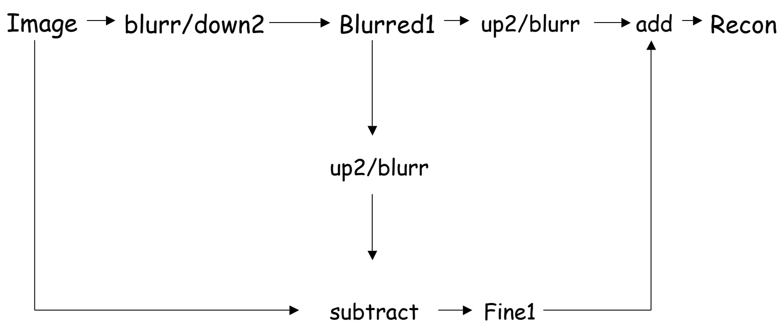
Laplacian Pyramid

blurred1 image – upsampled blurred2 image



original image – original image smoothed by gaussian

Schematic for construction/reconstruction



Laplacian pyramids

- Laplacian pyramids – each level holds the additional information which is needed for better resolution
- We can obtain same image by convolving the image with difference of Gaussians filter or appropriate width
- Reflects the similarity between difference of Gaussians DoG and Laplacian operator introduced in the edge detection stage

Finding faces

- Faces "look like" templates (at least when they're frontal).
- General strategy:
  - search image windows at a range of scales
  - Correct for illumination
  - Present corrected window to classifier
- Issues
  - How corrected?
  - What features?
  - What classifier?
  - what about lateral views?

Naive Bayes

- (Important: naive not necessarily perjorative)
- Find faces by vector quantizing image patches, then computing a histogram of patch types within a face
- Histogram doesn't work when there are too many features
  - features are the patch types
  - assume they're independent and cross fingers
  - reduction in degrees of freedom
  - very effective for face finders
    - why? probably because the examples that would present real problems aren't frequent.

Many face finders on the face detection home page
http://home.t-online.de/home/Robert.Frischholz/face.htm

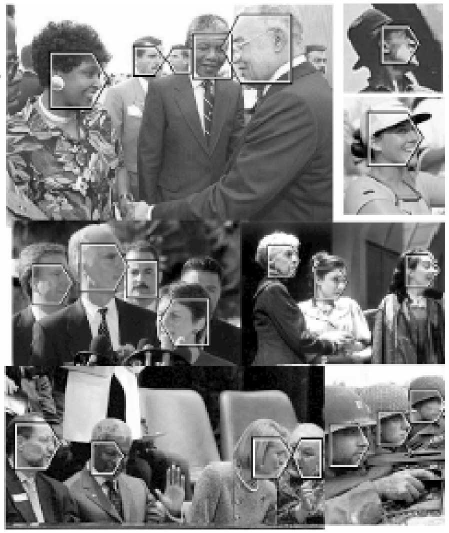


Figure from A Statistical Method for 3D Object Detection Applied to Faces and Cars, H. Schneiderman and T. Kanade, Proc. Computer Vision and Pattern Recognition, 2000, copyright 2000, IEEE

Summary of 2-D object matching/recognition

- Binary vision systems
  - segmentation by thresholding and connected component analysis
  - object modeling using statistical techniques
    - means and variances of global object features such as area, perimeter, etc.
  - recognition using statistical recognition techniques
    - k-nearest neighbors
    - Bayesian recognition
- Drawbacks
  - touching objects
  - occluded objects
  - weak segmentation techniques
- Grey level vision systems
  - (optional) segmentation by edge detection
  - object modeling by templates
    - gray level region templates
    - edge templates (binary)
  - recognition using correlation
    - brute force image correlation
      - speedup methods
    - Hough transform methods
    - Chamfer matching
- Drawbacks
  - computational complexity
    - to support rotations and scaling of templates