EE795: Computer Vision and Intelligent Systems

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

Lecture 07
130212

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

Outline

- Review
  - Filtering for Detection
  - Pyramids
- Feature Detection and Matching
- Points and Patches

Filtering 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

- Use normalized cross-correlation for intensity matching between mask and image
  - $NCC = \frac{1}{NM}\sum_{x,y}\frac{(f(x,y)-\bar{f})(t(x,y)-\bar{t})}{\sigma_f \sigma_t}$
    - $\bar{f}, \sigma_f$ - mean, std of image,
    - $\bar{t}, \sigma_t$- mean, std of template
  - This is the inner product, and NCC is the cosine angle between the image vectors

Correlation Masking

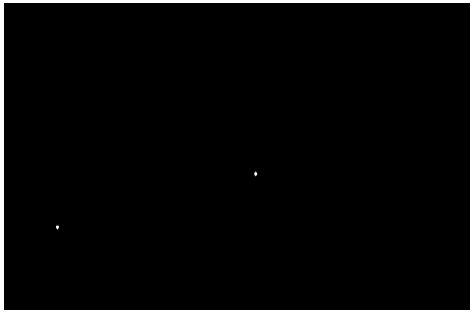
template

Detected template

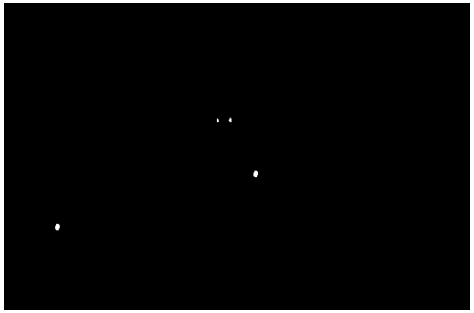
correlation

This is who I am.
Nobody said
you had to like it.

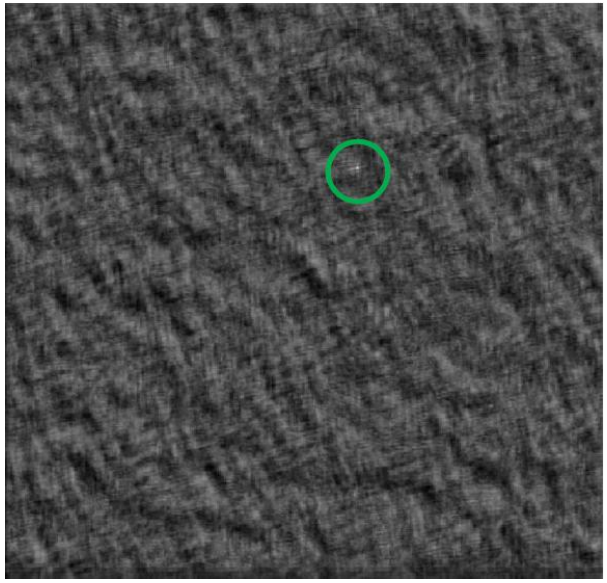
detected letter



0.9 max threshold



0.5 max threshold

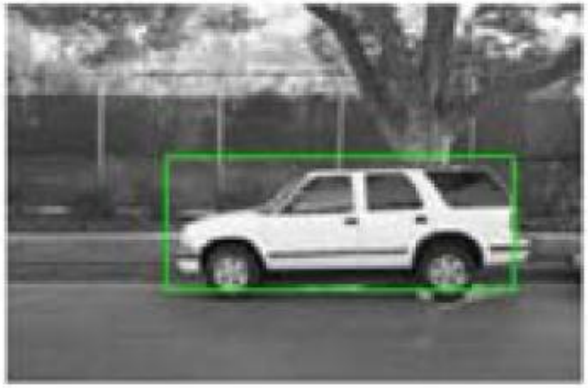


correlation map

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

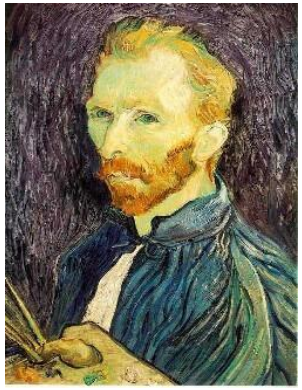
Adapted from K. Grauman

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?
  - Will have aliasing



1/2

1/4 (2x zoom)

1/8 (4x zoom)

Adapted from S. Seitz

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

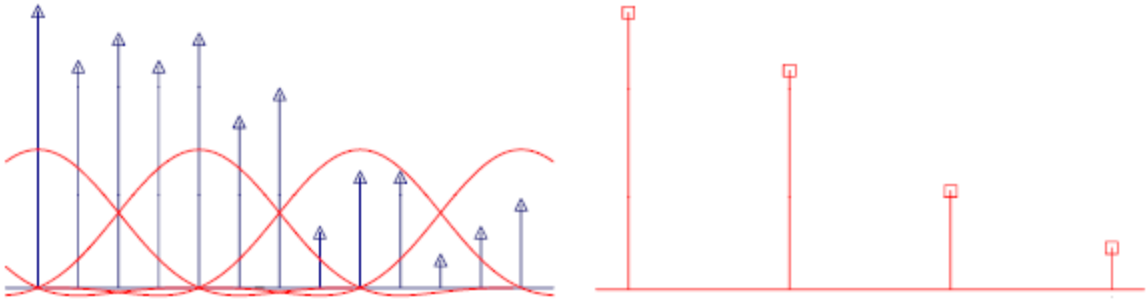


Figure 3.30 Signal decimation: (a) the original samples are (b) convolved with a low-pass filter before being downsampled.

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× 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.

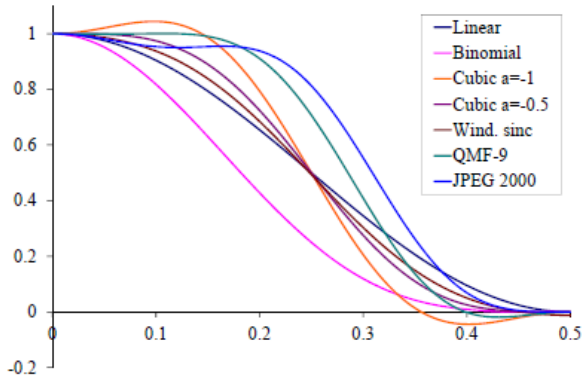
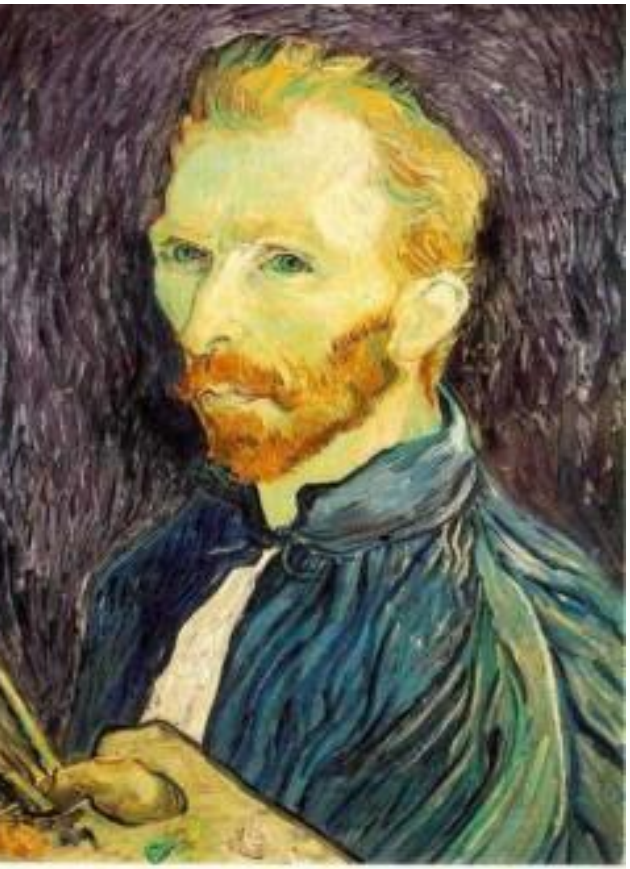


Figure 3.31 Frequency response for some 2× decimation filters. The cubic $a = -1$ filter has the sharpest fall-off but also a bit of ringing; the wavelet analysis filters (QMF-9 and JPEG 2000), while useful for compression, have more aliasing.

Subsampling with Gaussian pre-filtering

Adapted from S. Seitz

- Improved results by first smoothing



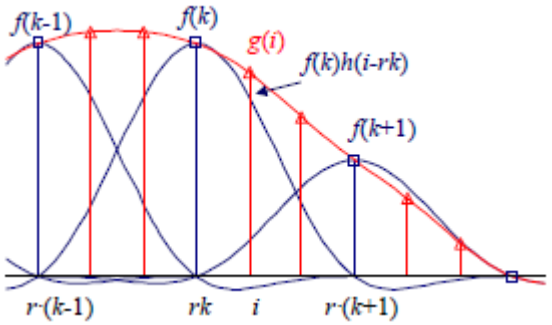
Gaussian 1/2

G 1/4

G 1/8

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



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

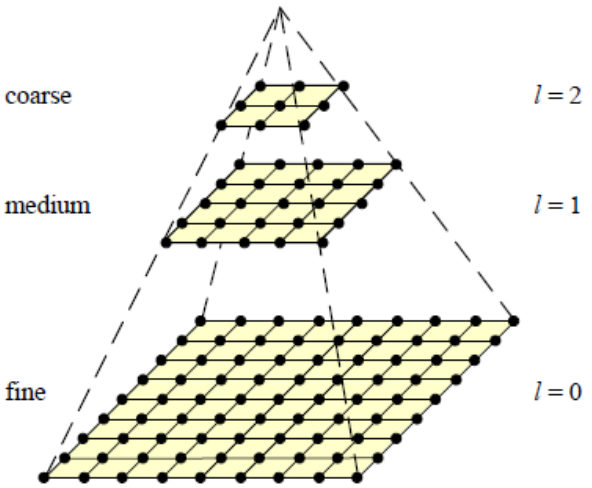
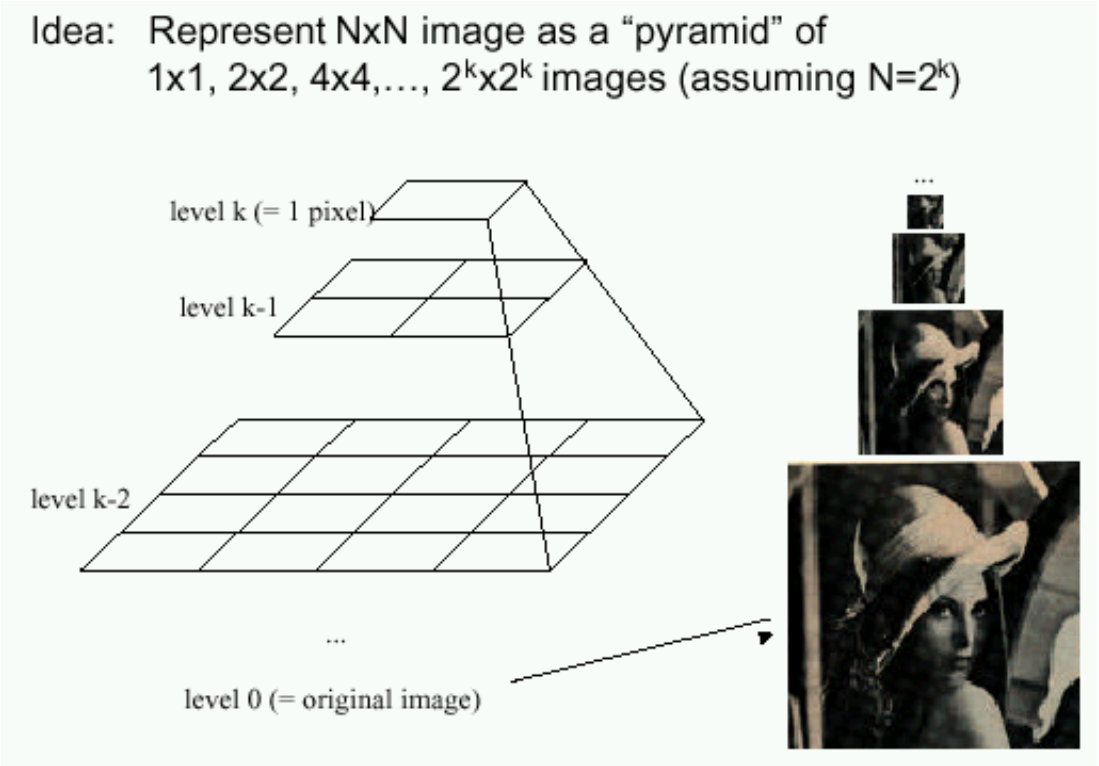


Figure 3.32 A traditional image pyramid: each level has half the resolution (width and height), and hence a quarter of the pixels, of its parent level.

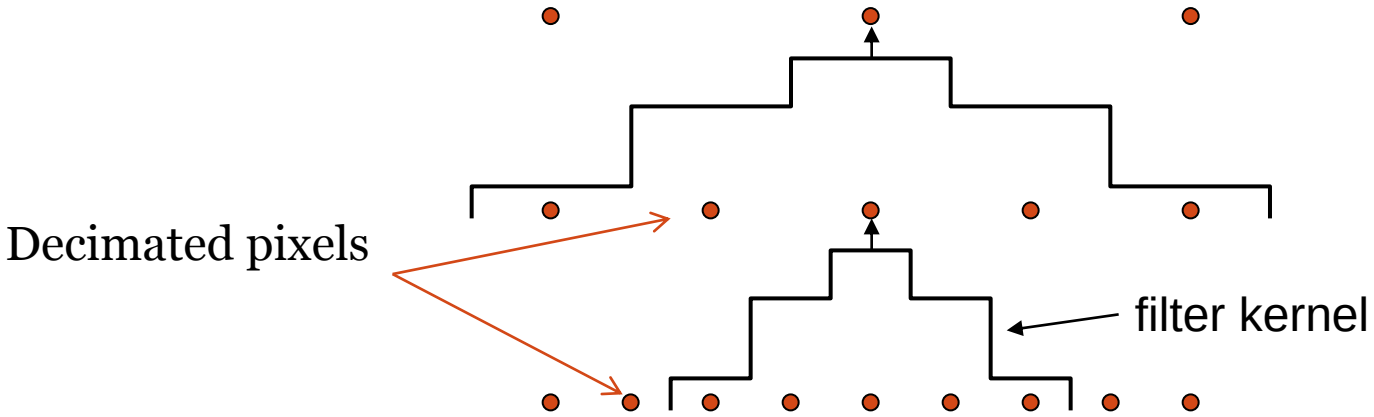
Laplacian (Gauassian) Pyramid

- [Burt and Adelson, 1983]
- Blur and subsample the original image by a factor of 2 at each level
  - Avoid aliasing with decimation
- Keep the Laplacian image for reconstruction of higher resolution image
  - Laplacian (derivative of Gaussian) required for perfect reconstruction



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

Gaussian Pyramid Signal Processing Diagram

- [Down] High resolution image to coarse resolution
  - Smooth then decimate
- [Down] Low resolution image to fine resolution
  - Interpolate by upsample and smooth

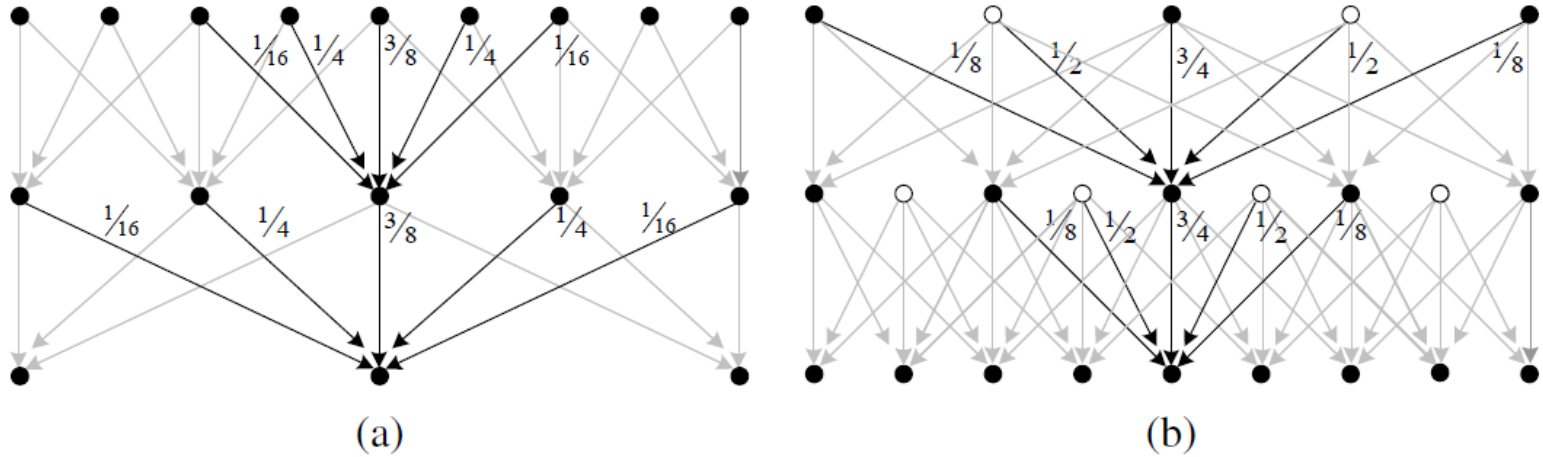


Figure 3.33 The Gaussian pyramid shown as a signal processing diagram: The (a) analysis and (b) re-synthesis stages are shown as using similar computations. The white circles indicate zero values inserted by the $\uparrow 2$ upsampling operation. Notice how the reconstruction filter coefficients are twice the analysis coefficients. The computation is shown as flowing down the page, regardless of whether we are going from coarse to fine or *vice versa*.

Feature Detection and Matching

- Essential component of modern computer vision
  - E.g. alignment for image stitching, correspondences for 3D model construction, object detection, stereo, etc.

- Need to establish some features that can be detected and matched

Determining Features to Match

- What can help establish correspondences between images?




Different Types of Features

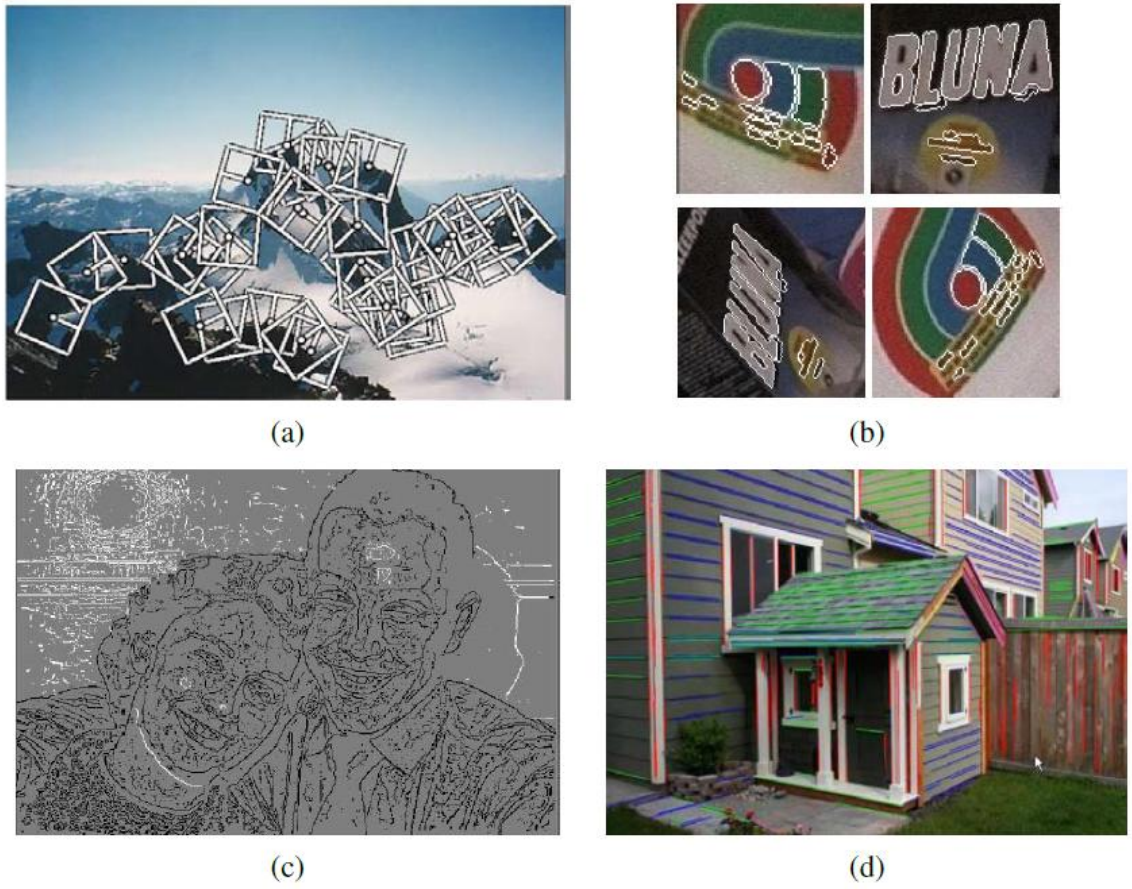


Figure 4.1 A variety of feature detectors and descriptors can be used to analyze, describe and match images: (a) point-like interest operators (Brown, Szeliski, and Winder 2005) © 2005 IEEE; (b) region-like interest operators (Matas, Chum, Urban *et al.* 2004) © 2004 Elsevier; (c) edges (Elder and Goldberg 2001) © 2001 IEEE; (d) straight lines (Sinha, Steedly, Szeliski *et al.* 2008) © 2008 ACM.

Different Types of Features

- Points and patches
- Edges
- Lines

- Which features are best?
  - Depends on the application
  - Want features that are robust
    - Descriptive and consistent (can readily detect)

Points and Patches

- Maybe most generally useful feature for matching
  - E.g. Camera pose estimation, dense stereo, image stitching, video stabilization, tracking
  - Object detection/recognition

- Key advantages:
  - Matching is possible even in the presence of clutter (occlusion)
  - and large scale and orientation changes

Point Correspondence Techniques

- Detection and tracking
  - Initialize by detecting features in a single image
  - Track features through localized search
  - Best for images from similar viewpoint or video
- Detection and matching
  - Detect features in all images
  - Match features across images based on local appearance
  - Best for large motion or appearance change

Keypoint Pipeline

- Feature detection (extraction)
  - Search for image locations that are likely to be matched in other images
- Feature description
  - Regions around a keypoint are represented as a compact and stable descriptor
- Feature matching
  - Descriptors are compared between images efficiently
- Feature tracking
  - Search for descriptors in small neighborhood
  - Alternative to matching stage best suited for video

Feature Detectors

- Must determine image locations that can be reliably located in another image

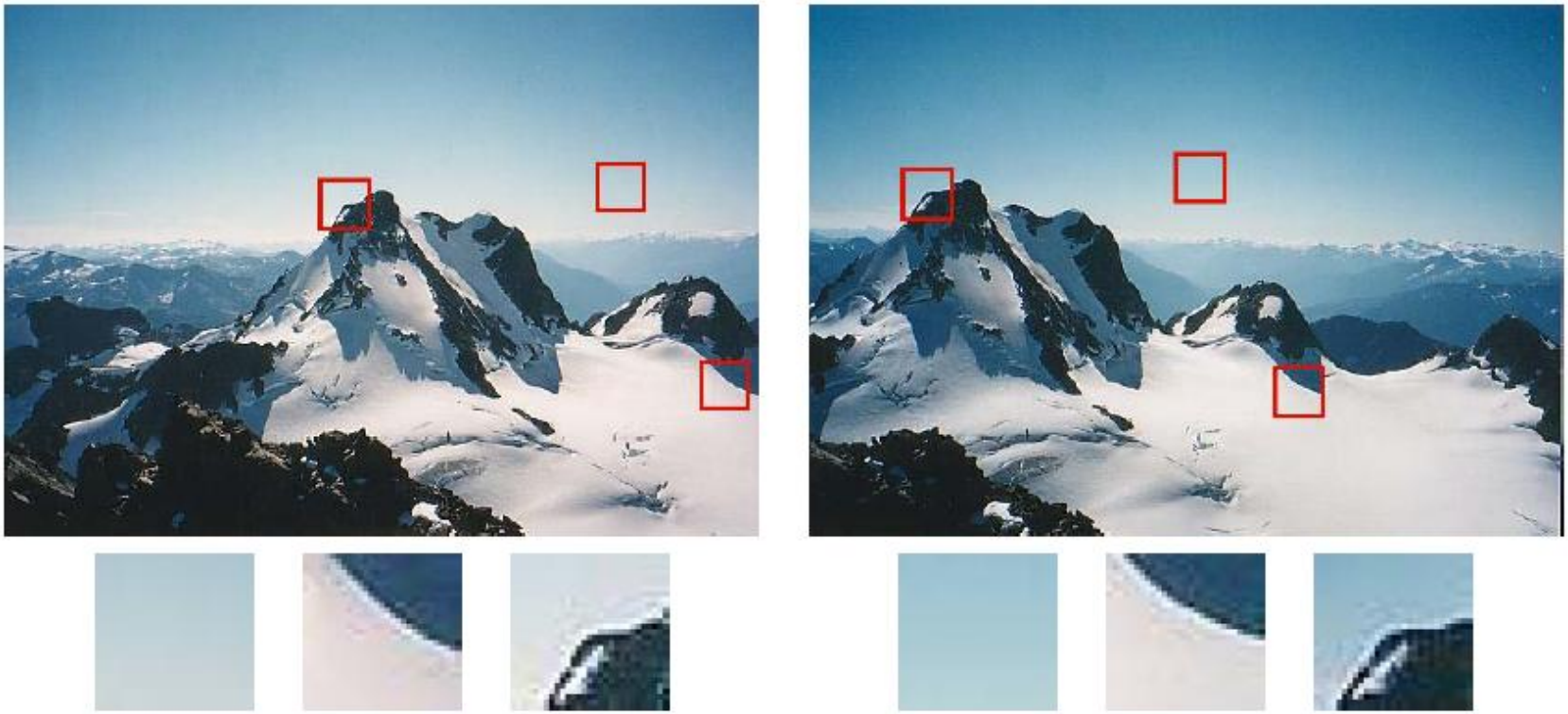


Figure 4.3 Image pairs with extracted patches below. Notice how some patches can be localized or matched with higher accuracy than others.

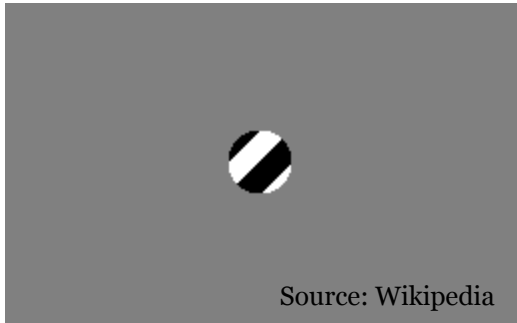
Comparison of Image Patches

- Textureless patches
  - Nearly impossible to localize and match
    - Sky region “matches” to all other sky areas
- Edge patches
  - Large contrast change (gradient)
  - Suffer from aperture problem
    - Only possible to align patches along the direction normal the edge direction
- Corner patches
  - Contrast change in at least two different orientations
  - Easiest to localize

Aperture Problem I

- Only consider a small window of an image
  - Local view does not give global structure
  - Causes ambiguity

- Best visualized with motion (optical flow later)
  - Imagine seeing the world through a straw hole
  - Aperture Problem - MIT – Demo
  - Also known as the barber pole effect



Source: Wikipedia

Aperture Problem II

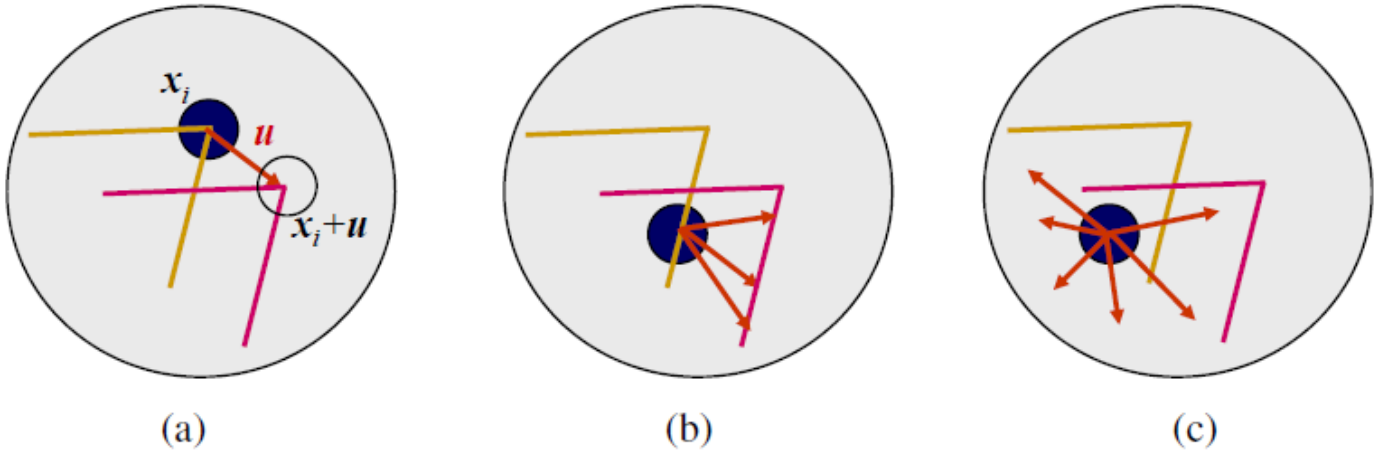


Figure 4.4 Aperture problems for different image patches: (a) stable ("corner-like") flow; (b) classic aperture problem (barber-pole illusion); (c) textureless region. The two images I_0 (yellow) and I_1 (red) are overlaid. The red vector u indicates the displacement between the patch centers and the $w(x_i)$ weighting function (patch window) is shown as a dark circle.

- Corners have strong matches
- Edges can have many potential matches
  - Constrained upon a line
- Textureless regions provide no useful information

WSSD Matching Criterion

- Weighted summed squared difference
  - $E_{WSSD}(\boldsymbol{u}) = \sum_i w(\boldsymbol{x}_i)\,[I_1(\boldsymbol{x}_i - \boldsymbol{u}) - I_0(\boldsymbol{x}_i)]^2$
    - I_1, I_0 - two image patches to compare
    - $\boldsymbol{u} = (u, v)$ – displacement vector
    - $w(\boldsymbol{x})$ - spatial weighting function
- Normally we do not know the image locations to perform the match
  - Calculate the autocorrelation in small displacements of a single image
    - Gives a measure of stability of patch
  - $E_{AC}(\Delta\boldsymbol{u}) = \sum_i w(\boldsymbol{x}_i)\,[I_0(\boldsymbol{x}_i - \Delta\boldsymbol{u}) - I_0(\boldsymbol{x}_i)]^2$

Image Patch Autocorrelation

$$E_{AC}(\Delta \boldsymbol{u}) = \sum_i w(\boldsymbol{x}_i)\,[I_0(\boldsymbol{x}_i - \Delta \boldsymbol{u}) - I_0(\boldsymbol{x}_i)]^2$$
$$= \sum_i w(\boldsymbol{x}_i)\,[\nabla I_0(\boldsymbol{x}_i) \cdot \Delta \boldsymbol{u}]^2$$
$$= \Delta \boldsymbol{u}^T A \Delta \boldsymbol{u}$$

- $\nabla I_0(\boldsymbol{x}_i)$ - image gradient
  - We have seen how to compute this
- A – autocorrelation matrix

$$A = w * \begin{bmatrix} I_x^2 & I_x I_y \\ I_y I_x & I_y^2 \end{bmatrix}$$

  - Compute gradient images and convolve with weight function
  - Also known as second moment matrix

- Example autocorrelation

Image Autocorrelation II

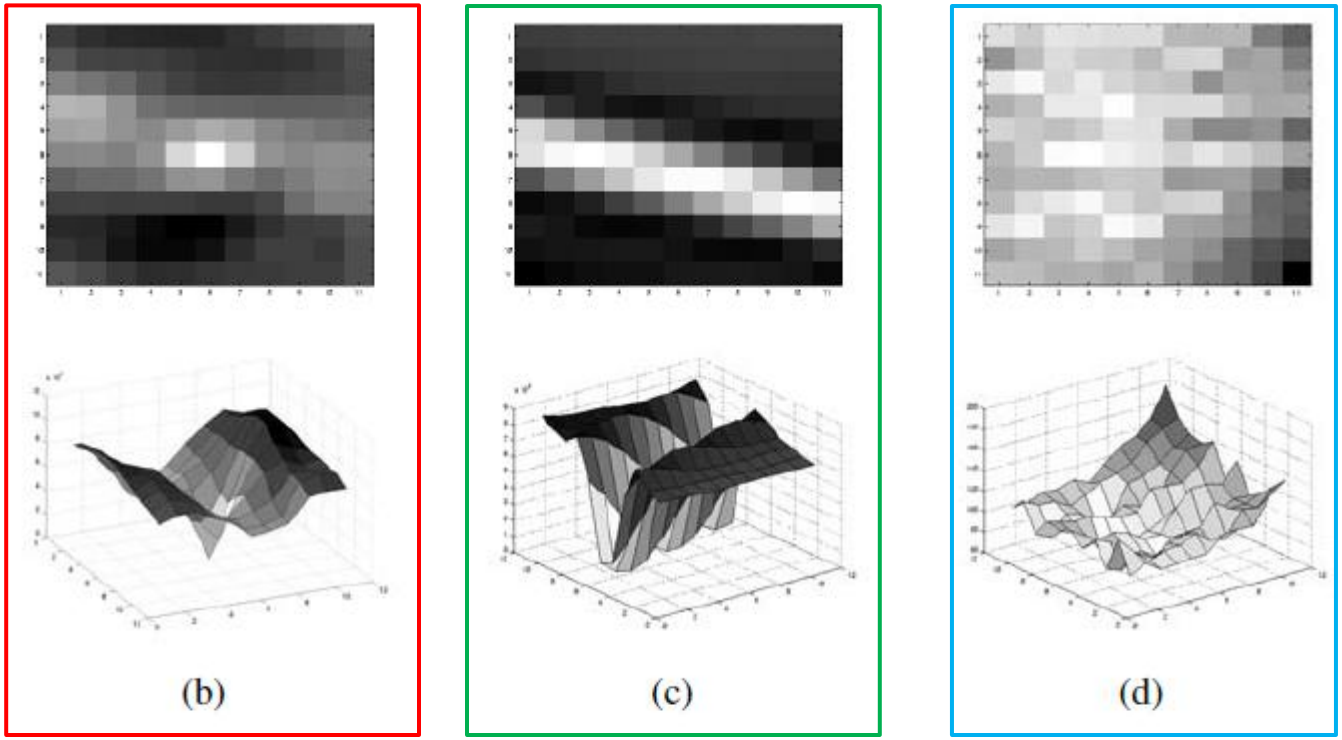


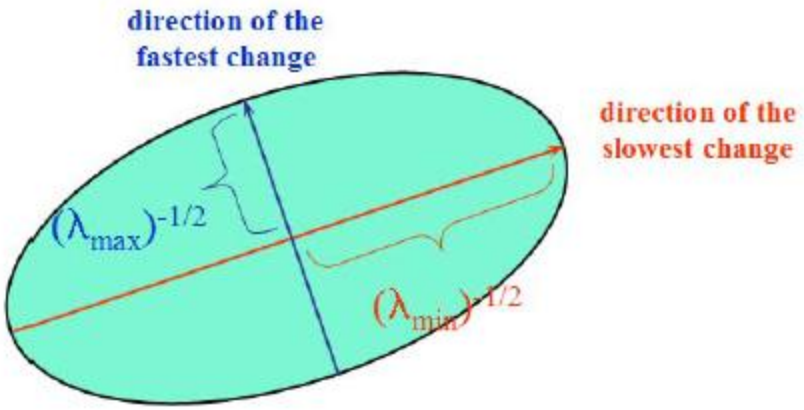
Figure 4.5 Three auto-correlation surfaces $E_{\mathrm{AC}}(\Delta u)$ shown as both grayscale images and surface plots: (a) The original image is marked with three red crosses to denote where the auto-correlation surfaces were computed; (b) this patch is from the flower bed (good unique minimum); (c) this patch is from the roof edge (one-dimensional aperture problem); and (d) this patch is from the cloud (no good peak). Each grid point in figures b–d is one value of Δu.

Image Autocorrelation III

- The matrix A provides a measure of uncertainty in location of the patch
- Do eigenvalue decomposition
  - Get eigenvalues and eigenvector directions

- Good features have both eigenvalues large
  - Indicates gradients in orthogonal directions (e.g. a corner)

- Uncertainty ellipse



- Many different methods to quantify uncertainty
  - Easiest: look for maxima in the smaller eigenvalue

Basic Feature Detection Algorithm

1. Compute the horizontal and vertical derivatives of the image I_x and I_y by convolving the original image with derivatives of Gaussians (Section 3.2.3).
2. Compute the three images corresponding to the outer products of these gradients. (The matrix $\boldsymbol{A}$ is symmetric, so only three entries are needed.)
3. Convolve each of these images with a larger Gaussian.
4. Compute a scalar interest measure using one of the formulas discussed above.
5. Find local maxima above a certain threshold and report them as detected feature point locations.

Algorithm 4.1 Outline of a basic feature detection algorithm.