IN 5520 25.9.18 Feature extraction -Information from gradients - Features from convolutions and filter banks

Anne Solberg (anne@ifi.uio.no)

Today

- Investigate the use of gradients for feature extraction
  - Histogram of Oriented Gaussians (HOG)
  - Speed Up Robust Features (SURF)
- Study the main principles and limitations of filter banks for feature extraction
  - Link to convolutional networks

Reading material

- HOG-features:
  - https://hal.inria.fr/file/index/docid/548512/filename/hog_cvpr2005.pdf

- SURF:
  Tutorial:SURF-tutorial

Histogram of Gradients feature desctiptor

- Concept introduced as part of the SIFT-transform (Scale Invariant Feature Transform) for matching keypoints in image pairs, but useful as feature descriptor also.
- Histogram of oriented gradient for human detection use a modified version for classification.

HOG overview

1. Normalize color/contrast in input image
2. Compute gradients
3. Divide the image into spatial bins
4. Divide each bin into orientations 0-180 degrees
5. Weighted vote the gradients into neighboring spatial and orientation bins
6. Collect gradient orientation histograms for each spatial cell
7. Normalize gradients
8. Use histogram counts as features

   - Or reduce to e.g. mean orientation/mean amplitude

HOG step 1: color normalization

- Normalize each channel to [-1,1]

HOG step 2: compute gradients

- Gaussian filtering as preprocessing can be considered, but might blur the result.
- Use a simple gradient operator, eg. Sobel or Prewitt
- Compute gradient direction and magnitude for each pixel
  - Do you remember how?

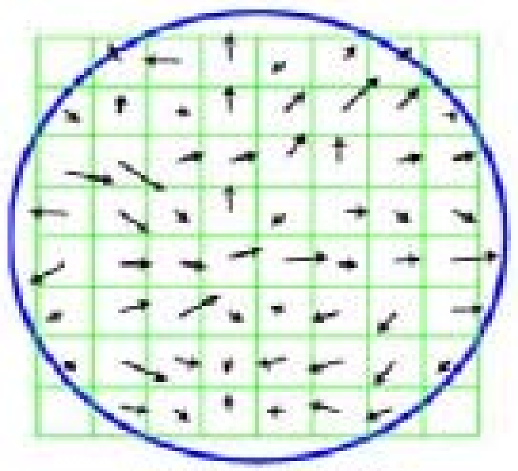


Image gradients

HOG step 3: divide image into spatial cells

- Use cells of a given size (e.g. 2x2, 4x4, 8x8, 16x16..) and partly overlapping windows (stride = window size/2)
- The optimal size depends on the size of the objects in the image.
- HOG is not scale invariant!
- Consider studying orientation histograms at several cell sizes to choose a the best size.

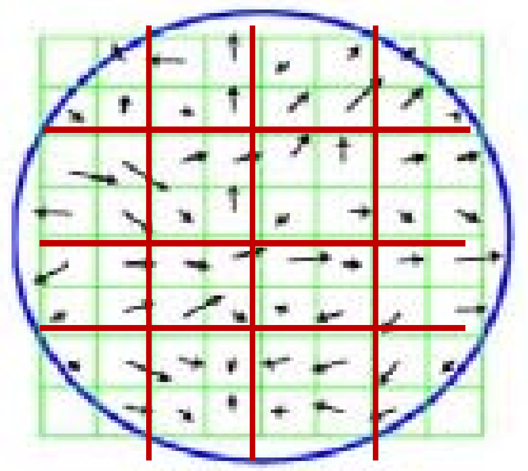
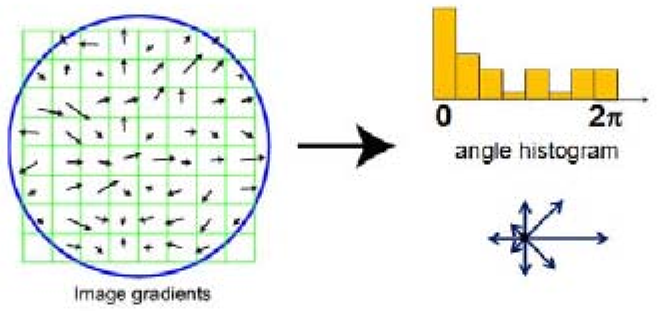


Image gradients

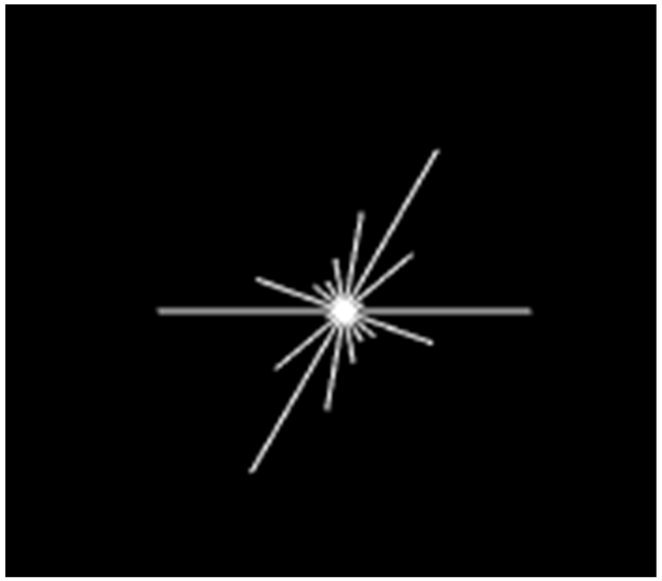
HOG step 4: divide each cell into orientation grids

- Each pixel inside a cell has a gradient direction and magnitude
- Divide the cell into either rectangular or circular bins (log-polar).
- Divide the orientation into 0-180 degrees ignoring the sign (usually 9 bins)
  - Use bilinear interpolation to distribute a gradient value into neighboring directional bins
  - Weight the histogram increment with the gradient magnitude
- Weight the bin with a Gaussian window function to have largest influence of pixels in the center of the bin (combining this weight with the gradient magnitude)



HOG step 5-6: bin histogram

- Compute an orientation histogram for each cell by accumulating the gradient information
- Can be visualized for a cell:

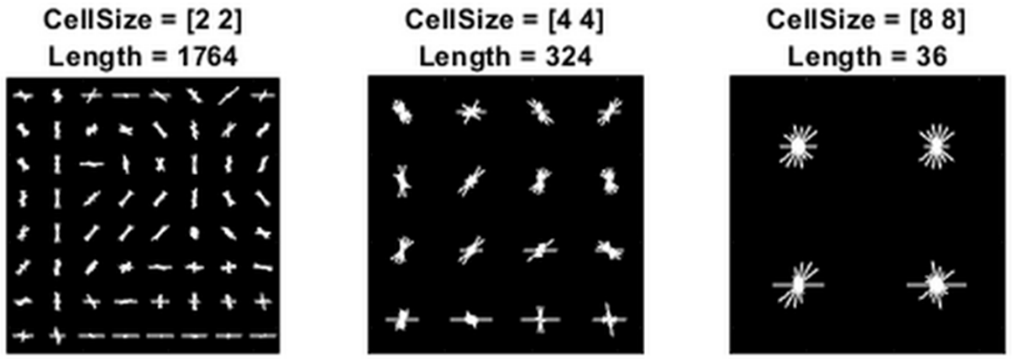


- As features we use the 9 values in each bin, so we have 9*nof. Bins features per grid cell

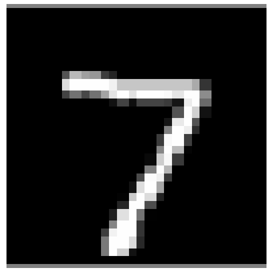
HOG step 7: normalize features

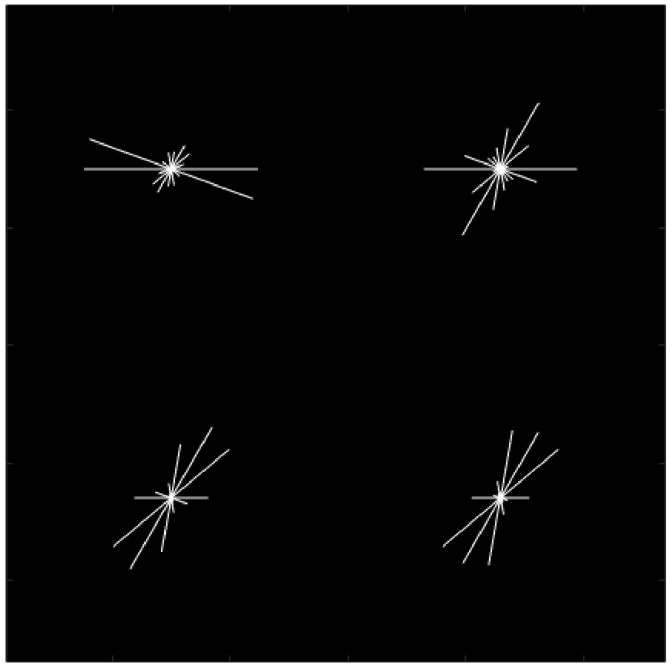
- The gradient values are normalized insize a block.
- Consider value v, normalize as
- $v = v/\sqrt{\|v\|_2^2 + \varepsilon^2}$
  Clip large values down to 0.2

HOG example


CellSize = [2 2]
Length = 1764
CellSize = [4 4]
Length = 324
CellSize = [8 8]
Length = 36

HOG example





2x2 cells of bin size 14x14 for image of size 28x28

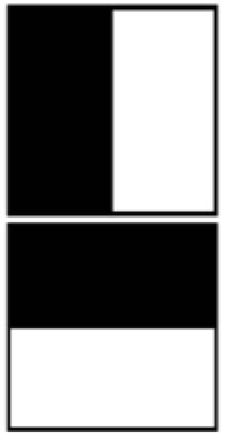
Feature vector has 9 numbers for each cell, in this example length 36

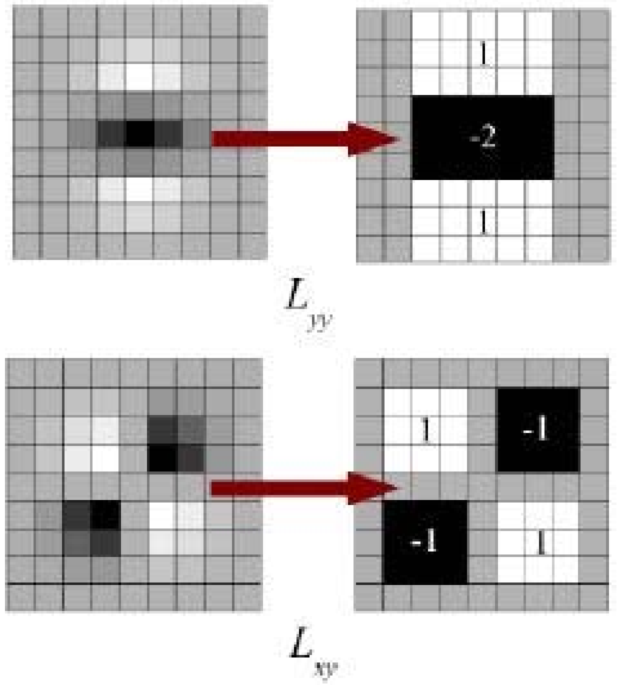
9 values corresponds to normalized histogram of gradients distributed over 9 directions

Speeded Up Robust Features (SURF)

- Speeded Up Robust Features
- Originally used to detect and characterize keypoint for matching image pairs, but also useful as feature descriptors for dense locations and classification.
- Can be made invariant to orientation by estimating dominant orientation and centering the window about this orientation, but for descriptors U-SURF (upright SURF) this is not done.
- SURF is based on integer images for fast computation, and uses Haar-wavelets

Integer approximations of filters

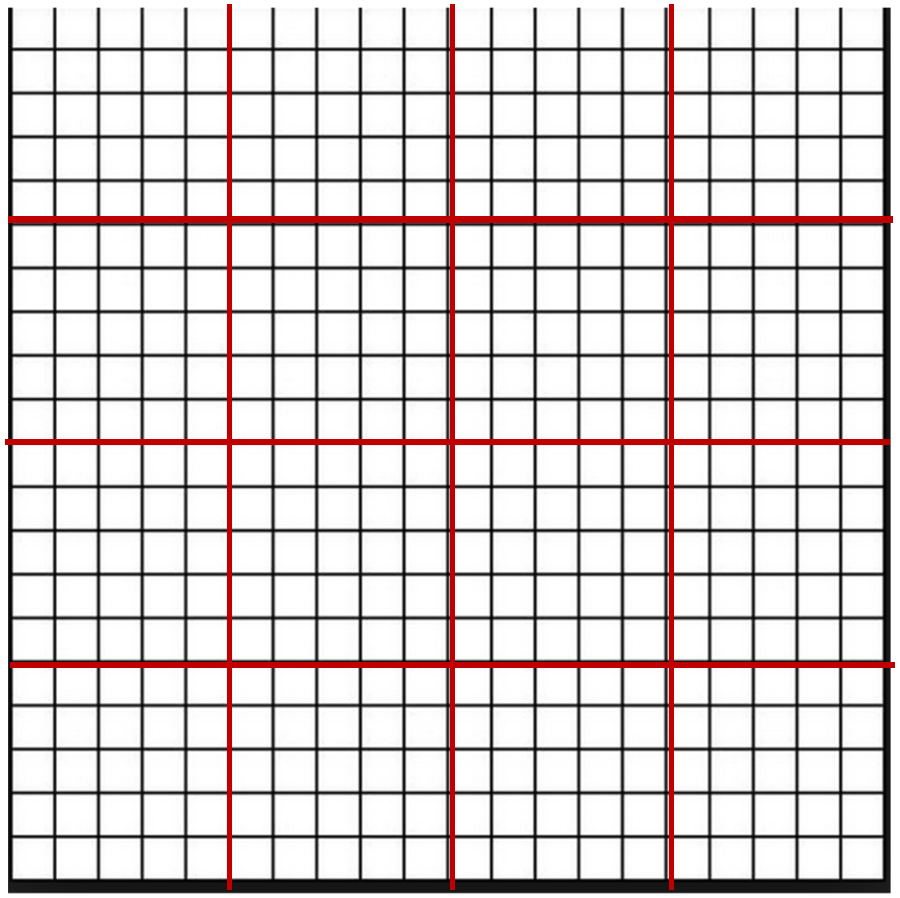




Approximations of gradients
Values 1 and -1 in the masks
Very fast to compute : use sums

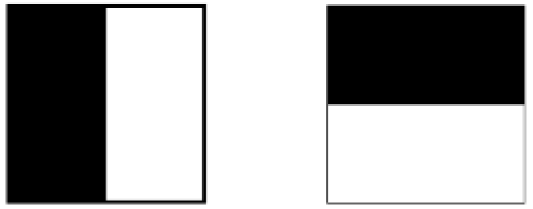
Approximations of second order derivatives,
Used in keypoint detection
Values 1 and -1 in the masks

SURF features

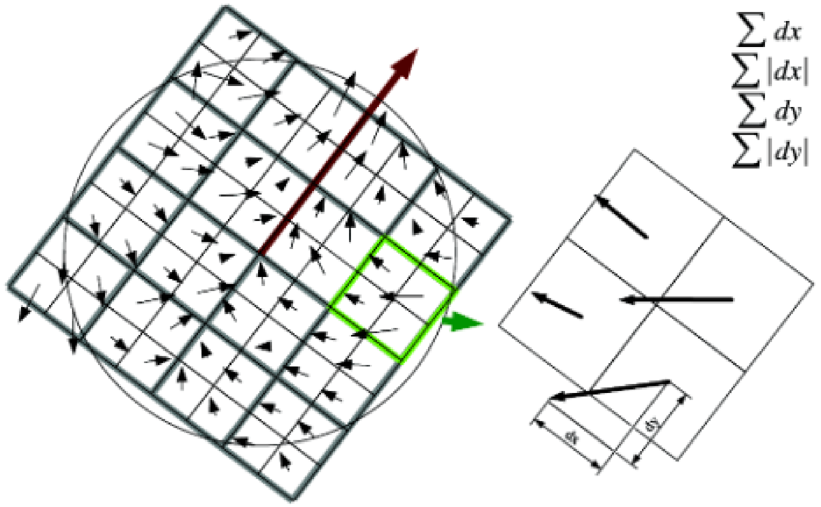


Size s
Neighborhood of size 20sx20s
Divide into 4x4 subregions of 5sx5s

For each point in the grid compute the horisontal and vertical Haar response, denoted dx and dy

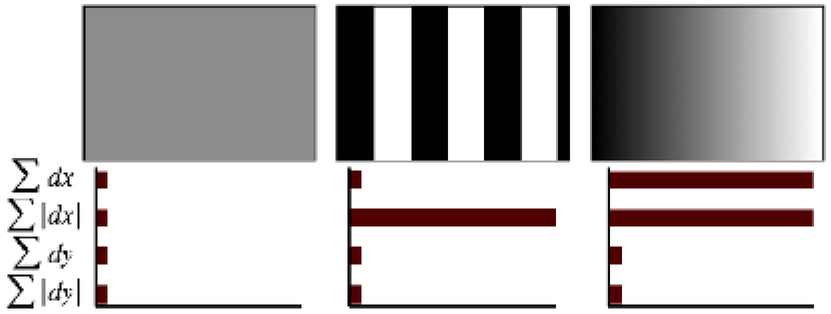


This gives 4



- Sum up dx, dy, |dx| and |dy| over each of the 4x4 sub windows to get 4 features for each 4x4 grid cell = 64 feature values.
- First, weight by a gaussian window to center the response at the center pixel.
- In the figure, the window is oriented along the local orientation, and the gradients (Haar-wavelets) computed according to this orientation.
  U-SIFT (upright SIFT skips this orientation estimation and centering)

Example of SURF features



Example of features for simple regions

More literature on related feature descriptors

- Moment invariants for recognition under changing viewpoint and illumination
- General intensity transformations and differential invariants
- Reliable feature matching across widely separated views
- Design and use of steerable filters
- Multi-scale phase-based local features
- improvement of handwritten japanese character recognition using weighted direction code histogram

Part 2

- Feature extraction from the Fourier domain using a filter bank
- Challenges when designing filters
  - Do we extract the relevant information?
- Relation to convolutional networks for feature extraction

Texture based on filtering

Based on:

- M. Tuceryan and A. Jain, Texture analysis, in Handbook of Pattern Recognition and Computer Vision, C. Chen, L. Pau, W. Wang (editors) :link

- T. Randen and J. H. Husøy, Filtering for texture classification: a comparative study, IEEE Tr. PAMI, vol. 21, no. 4. April 1999.

- A. Jain and F. Farrokhnia, Unsupervised texture segmentation using Gabor filters, Pattern Recognition, vol. 24, 1991. Link:

A taxonomy of texture models

We can characterize texture models into different groups:

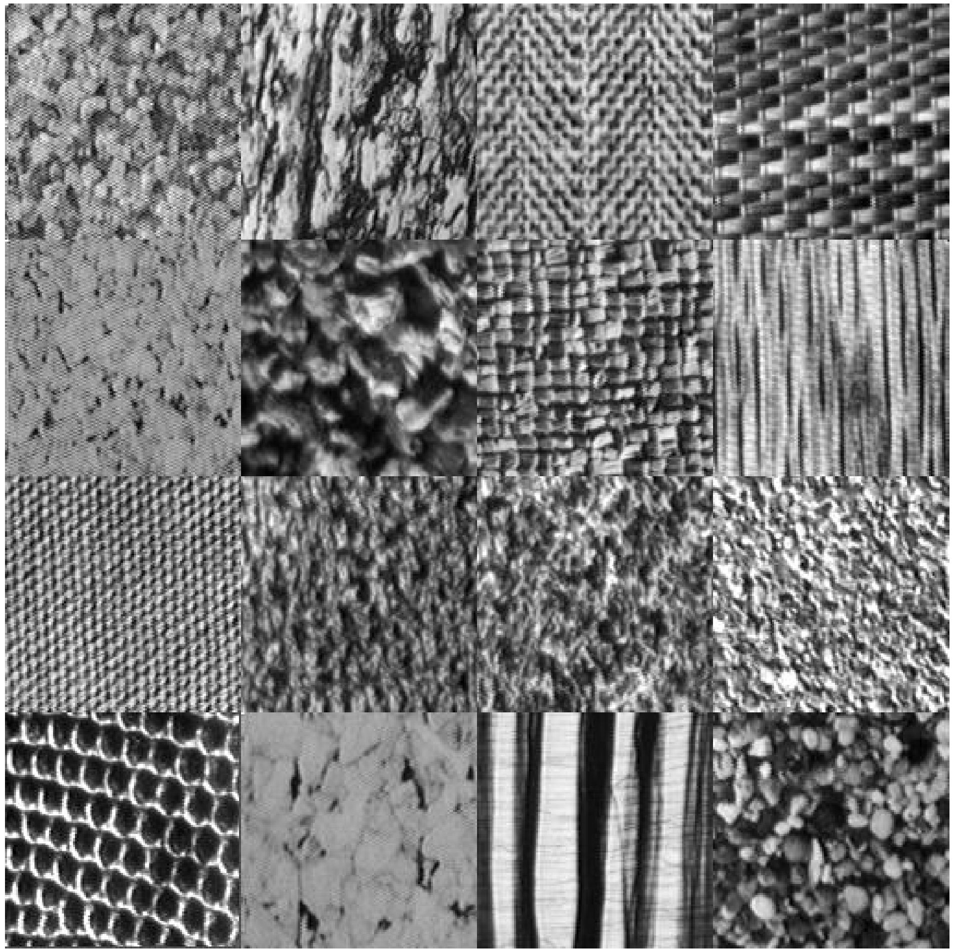
- Statistical models
  - GLCM, GLRL,
  - Autocorrelation features
- Geometrical models
  - Voronoi tesselation, structural models
- Model-based methods
  - Markov random field models
  - Fractals
- Signal-processing methods
  - Frequency-based methods like wavelets, Gabor filters, filter banks, etc.

Texture based on filtering

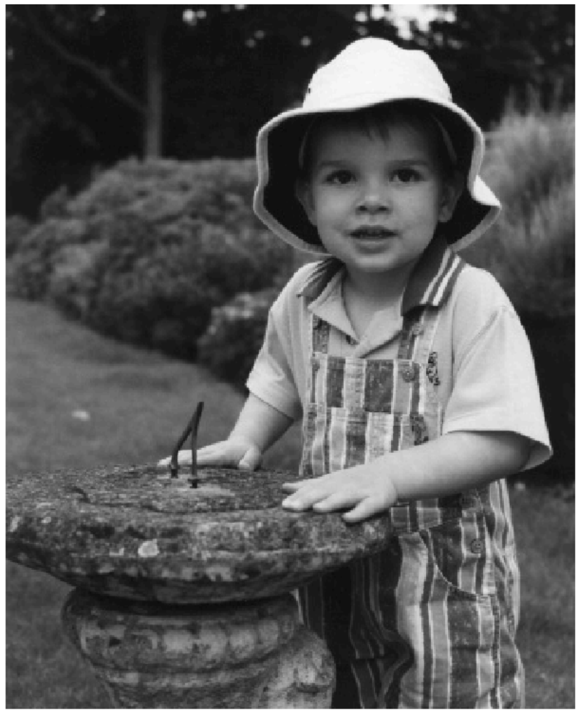
- To discriminate textures containing structures with different spatial frequencies or different orientations, spatial filtering methods are useful.
- The most common approach is to set up a filter bank with different filter, and compute the response to a set of filters with covering different ranges of the frequence spectrum. A special feature extraction function is then used to combine the filter outputs to texture descriptors.
- A simple approach is to use edge detection filters.
- This frequency-based approach is best suited to texture which can be identified as belonging to different regions of the Fourier spectrum.

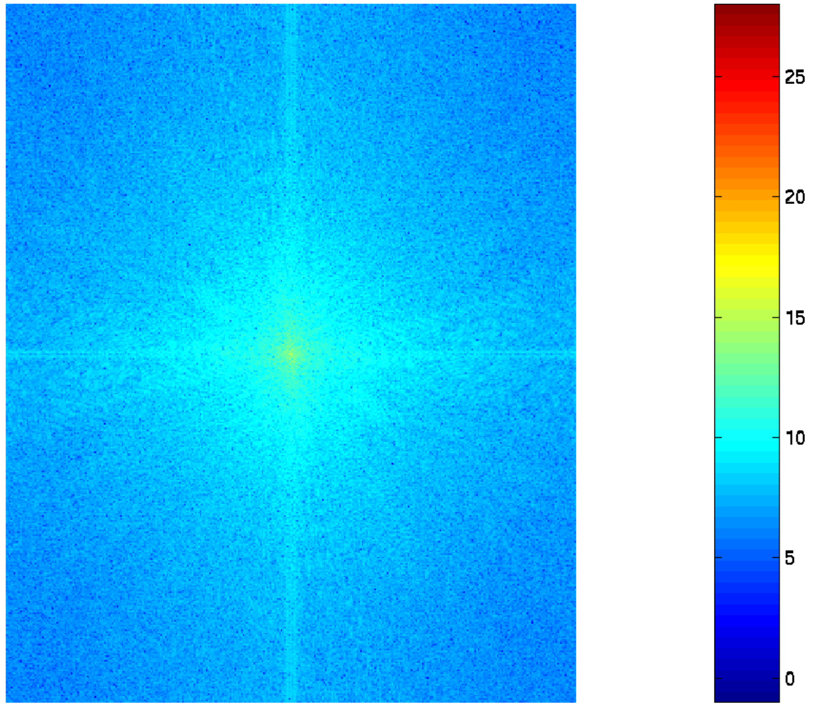
Energy in different part of the Fourier spectrum?

- Do the images contain information that is distributed in different parts of the Fourier spectrum?
  - Oriented structures
  - Repeated patterns
- If so – extracting features from the Fourier transform can be useful



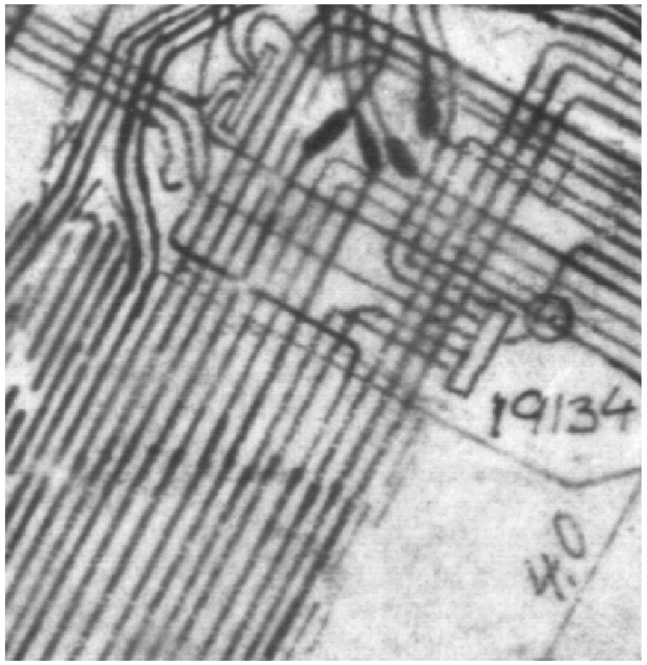
Example images

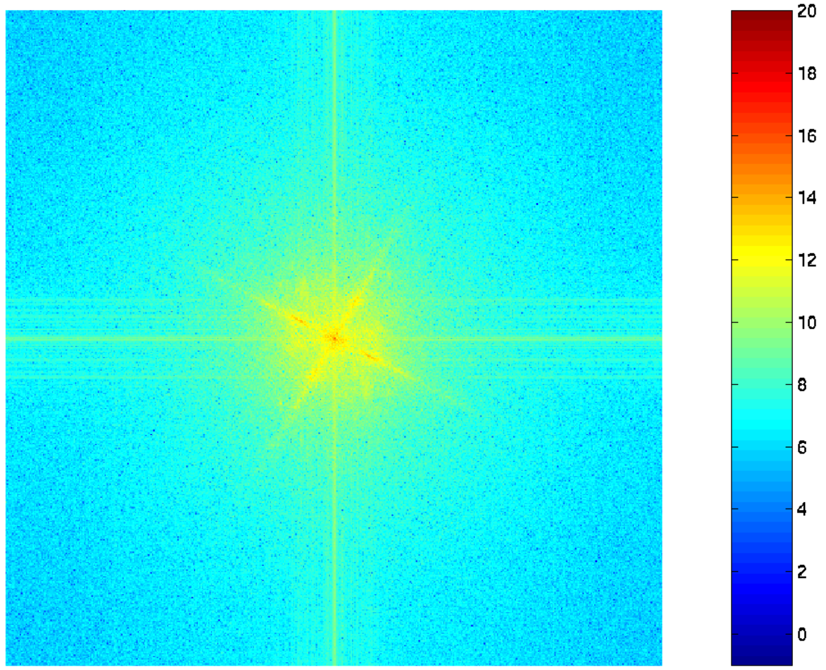




Fourier-spectrum

Do we expect different parts of this image to have different spectra?





This example has repeated, oriented structures

Filtrering in the frequency domain

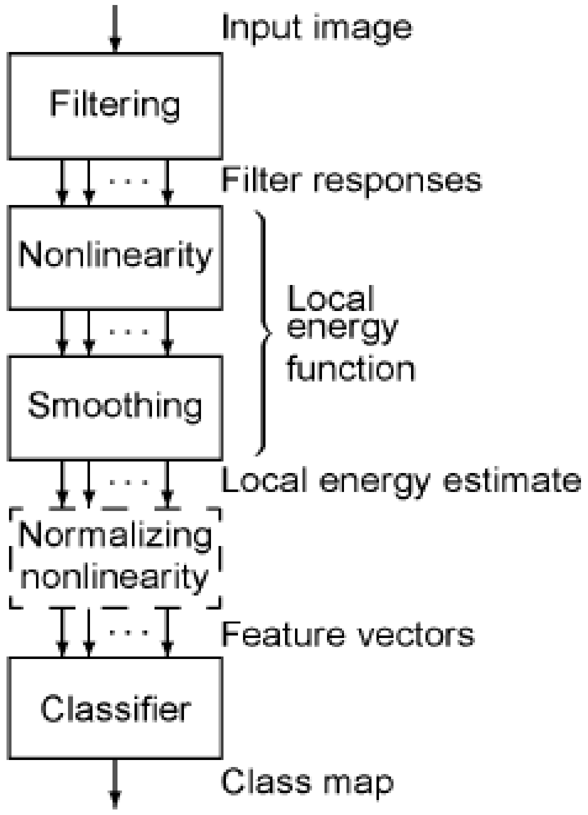
Convolution in the image domain is equivalent to multiplication in the frequency domain.

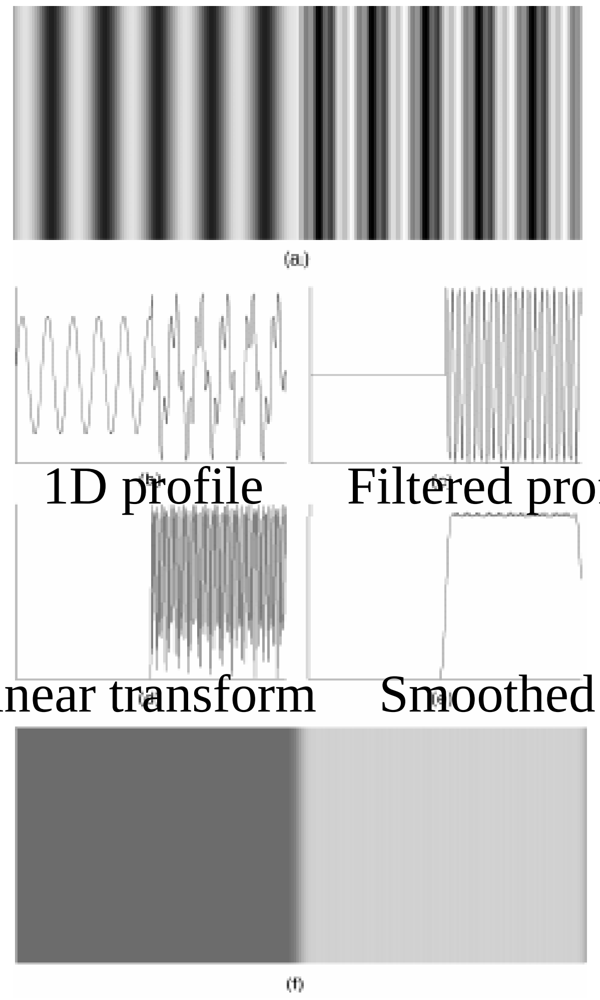
We can design filters looking for different types of structures.

Filtering in the Fourier domain:

1. Compute Fourier transform of the image
2. Compute Fourier transform of the filter
3. Multiply
4. Use inverser Fourier transform to convert back to image domain

Texture based on filter banks





Original

1D profile

Filtered profile

Nonlinear transform

Smoothed

Resulting 2D feature image

Designing the filters in the filter bank

- A filter bank is a collection of spatial filters which covers the most interesting parts of the frequency domain.
- To detect a set of textures, a filter bank with filters that are tailored to the frequency characteristics of the texture is needed.
- The main idea is to partition the frequency space into different regions and apply one filter for each region.
- How do we partion the frequency domain, and how many filters do we use?
- Can we use prior knowledge about the textures to tailor the filters?

Unsupervised filter banks

- Unsupervised means that no information about the textures is used when selecting the filter banks.
- Several approaches have been tried:
  - Laws filter masks
  - Ring and wedge filters
  - Gabor filter banks
  - Wavelet transform
  - Discrete Cosine Transform
  - Quadrature Mirror Filters

Gabor filter banks

- Based on phychophysical experiments for the human visual system.
- Campbell and Robson (1968) proposed that the visual system decomposes the retinal image into a number of filtered images, each of which contains intensity variations over a narrow range of frequency/size and orientation.
- The different filters can be seen a multi-channel filtering.

Gabor filter kernels

- We consider even-symmetric Gabor-filters of the following form:

$$h(x,y) = e^{-\frac{1}{2}\left[\frac{x^2}{\sigma_x^2}+\frac{y^2}{\sigma_y^2}\right]}\cos(2\pi f_0 x)$$

- This yields a filter with orientation 0°. Other orientations are found by rotating the reference coordinate system x,y. (Orientations 0°, 45°, 90°, and 135° are often used.)
- The Fourier-transform of a Gaussian function is a Gaussian, thus the filter frequency respons for each filter is a Gaussian function with a given center frequency and width.
- Gabor filters are claimed to give optimal localization properties both in the spatial and in the frequency domain (mainly because of their shape (Gaussian)).
- Different filter parameters can be choosen.

Frequency respons for a Gabor filter bank

•Jain and Farroknia suggests a set of filters with center frequencies $\frac{\sqrt{2}}{2^6}, \frac{\sqrt{2}}{2^5}, \frac{\sqrt{2}}{2^4}, \frac{\sqrt{2}}{2^3}, \frac{\sqrt{2}}{2^2}$

and orientations

0°, 45°, 90°, and 135°

This gives a almost uniform

coverage of the spectrum.

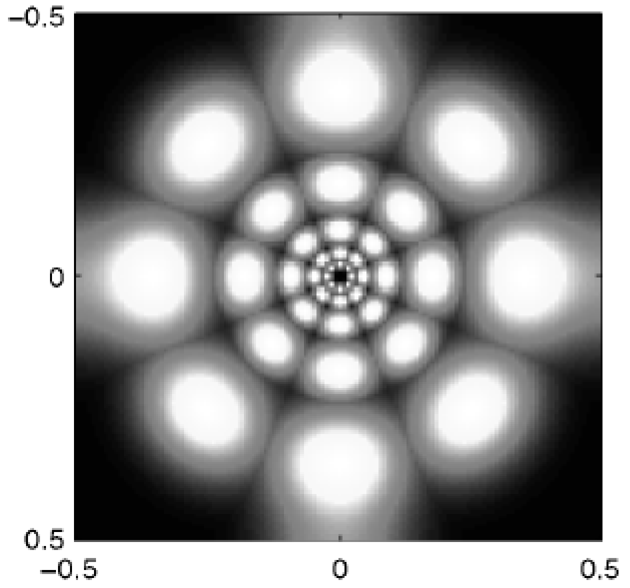


Fig. 5. The frequency response of the dyadic bank of Gabor filters. The maximum amplitude response over all filters is plotted. Each filter is represented by one center-symmetric pair of lobes in the illustration. The axes are in normalized spatial frequencies.

Wavelet filter banks

- The Gabor function in the spatial domain is an example of a wavelet.
- The discrete wavelet transform corresponds to a filter bank with particular filter parameters and subband decomposition.
- There are many familites of wavelets. A common family is the Daubechies family (Daubechies 1992)

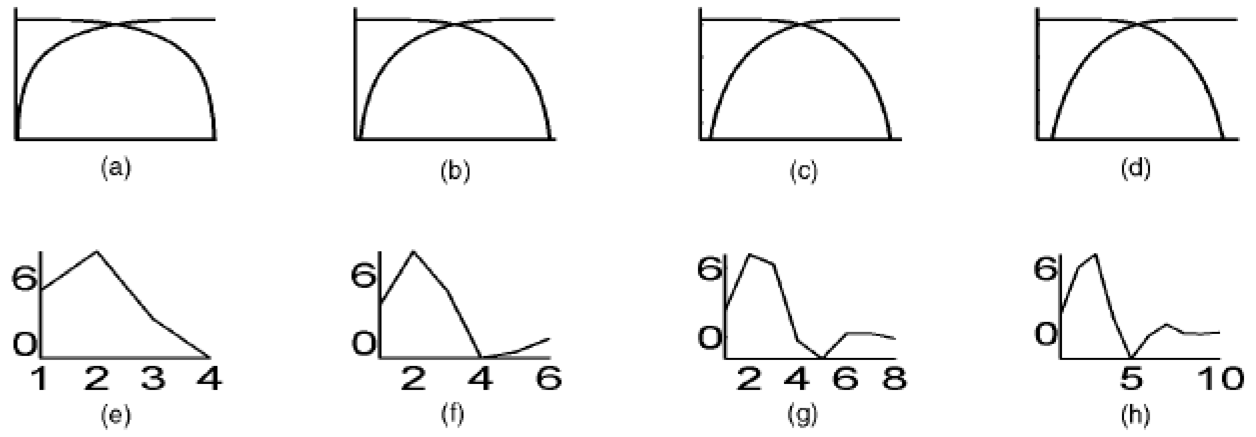


Fig. 7. The two-channel filter bank dB amplitude responses for the wavelet filters (a) "Daubechies 4," (b) "Daubechies 6," (c) "Daubechies 8," (d) "Daubechies 10," and their respective unit pulse responses (e)-(h).

From the output of a filter bank to texture features

- The result after applying a filter bank with M filters to an image is M filtered images.
- If texture is computed in a local window, M subimages result from each window position.
- These cannot be used as feature vectors directly. We try some kind of feature extraction to the filtered images. We are looking for a feature extraction step that will give constant feature values for regions with equal texture, and different for region with different texture.
- There is no evident way to do this. A common approach is to first perform a non-linear transform, then to smooth the resulting image.
- The success of the texture model will depend on the success of this step.

Jain and Farrokhnias feature extraction approach

- First, each filter is subjected to a nonlinear transform using a tanh function

$$\psi(t) = \tanh(\alpha t) = \frac{1 - e^{-2\alpha t}}{1 + e^{-2\alpha t}} \quad \text{where } \alpha \text{ is a constant}$$

- This results in a threshold-like function and gradual changes in the filtered images are converted to square-like blobs.
- Then, they compute the average deviation from the mean in small overlapping windows

$$e_k(x, y) = \frac{1}{M^2} \sum_{(a,b)} \left| \psi\left(r_k(a, b)\right) \right| \text{ where } r_k \text{ is the filtered image no. k}$$

The smoothing step

- The method is sensitive to the window size M.
- In filter design, there is a trade-off between good frequency localization (filters with large spatial width) and filters with high spatial resolution (filters which covers a broad part of the frequency spectrum).
- Too smooth low frequencies, we need a filter with large window size, while smaller window sizes can be used for higher frequencies.
- Thus, let the filter size be a function of the center frequency of the window.

- Let f_0 be the radial center frequency for the a bandpass filter in the filter bank.
- Use the following corresponding Gaussian lowpass filter

$$h_G(n) = \frac{1}{\sqrt{2\pi}\sigma_s} e^{-\frac{1}{2}\frac{n^2}{\sigma_s^2}}$$

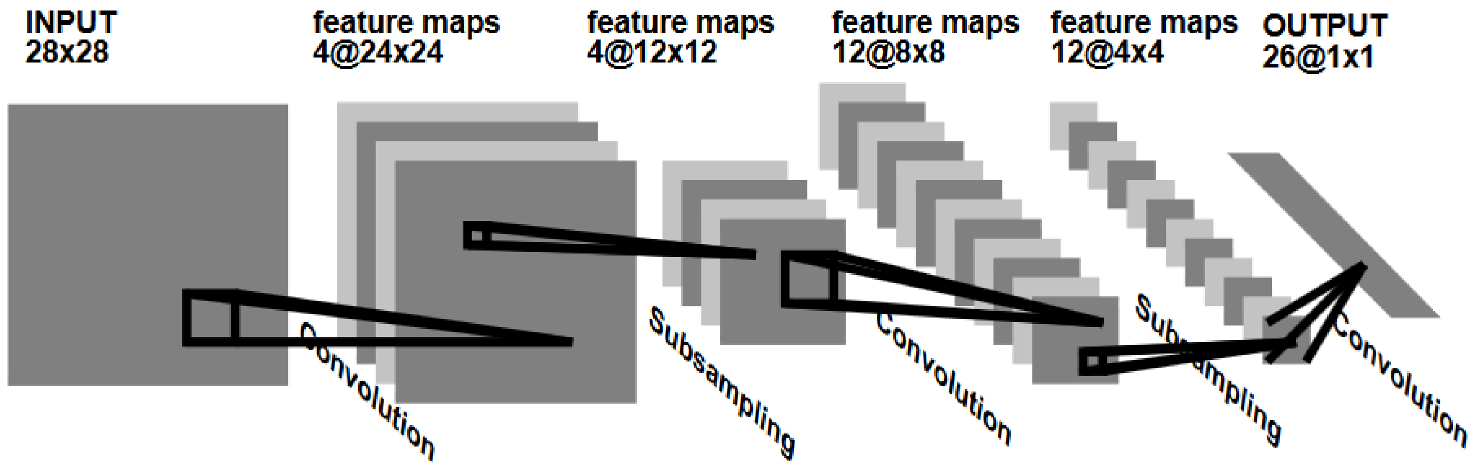
$$\text{where } \sigma_s = \frac{1}{2\sqrt{2} f_0}$$

- Other choices are also used.

The texture segmentation or classification step

- After the filtering, nonlinear transform and smoothing, a set of K feature images result. How do we use these to discriminate between various textures?
- It is possible to use them as input to a regular feature selection process. They can be used either individually or as a multivariate feature vector.

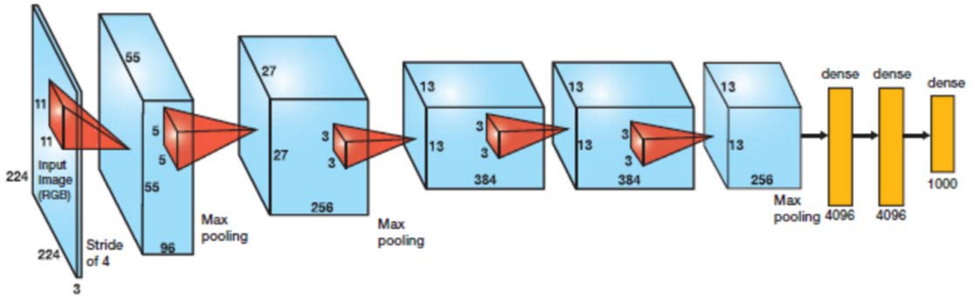
Similarity to convolutional networks

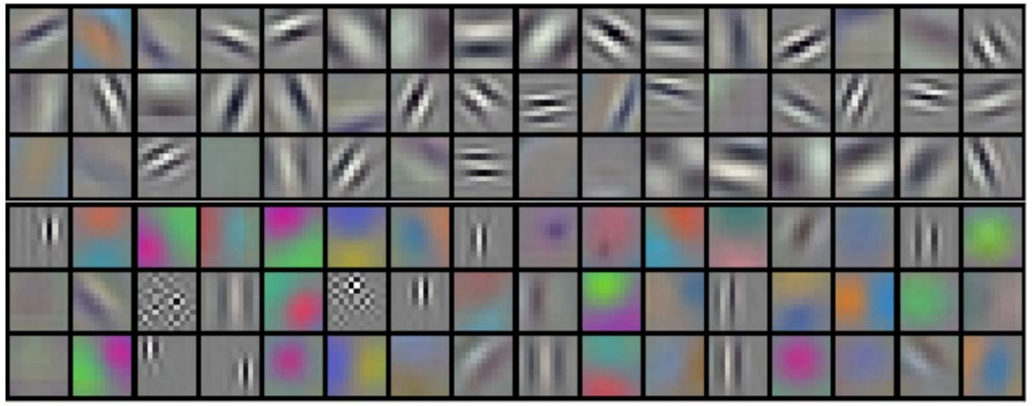


Select D layers
Select Ki filters at each layer
Each filter is a convolution applied to every position in the image
Nonlinear activation after each convolution
The filter coefficients are automatically learned from large data sets

Visualizing and Understanding ConvNets

- AlexNet, the winner of the ImageNet classification challenge 2012.
- Filter bank of size (11x11x3)x96 for the first convolutional layer:
- Visualizing the learnt weights of the first layer

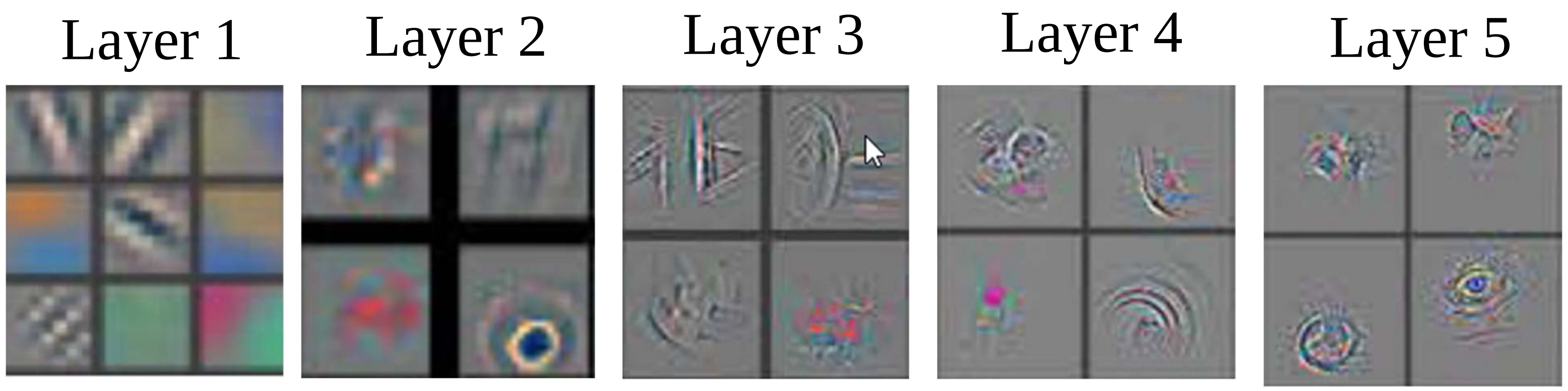




Figures copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012.

Visualizing and Understanding deeper layers

- Visualization what kind of input will have the largest response



Zeiler M.D., Fergus R. (2014) Visualizing and Understanding Convolutional Networks

What kind of features should I use?

- If your application involves natural images, and you have a lot of training data to adapt a pretrained convolutional net, try it!
- If your application either:
  - Uses non RGB images, e.g. ultrasound, sonar etc.
  - Has limited training data
  - Has prior assumptions about the problem
  - Has very skewed classes

  Then traditional feature extraction methods can be best.

  Always analyze your problem! What kind of information do I think is useful.

Learning goals

- Understand HOG feature extraction
- Understand SURF feature extraction
- Have a brief overview only of extracting features based on a filter bank.
  - Mathematical details not important.