1 Feature detectors

1.1 Overview

Goal of feature/interest point detectors:

- Detect all 'true' features and no false ones (i.e. high precision, recall)
- Well localised points
- Robust to noise
- Computationally efficient

The standard approach is to look for regions large gradient changes in x and y directions. If we denote $I(x, y)$ as the intensity at point x,y, then we can calculate the weighted sum of square intensity difference between two images (or a region of an image shifted) with:

$$E_{wssd} = \sum_i w(x_i, y_i)[I(x_i + u, y_i + u) - I(x_i, y_i)]^2$$

where $w(x_i, y_i)$ is the weighting function, in it's simplest case it would be uniform.

1.2 Harris Corner Detector

Idea: Scan a window over an image and look for significant intensity changes. Use a Taylor series expansion to approximation $I(x_i + u, y_i + v)$:

$$I(x_i + u, y_i + v) = I(x_i, y_i) + u\frac{\partial I}{\partial x}(x_i, y_i) + v\frac{\partial I}{\partial y}(x_i, y_i)$$

This can be represented in matrix form and denoted **M**. In a practical sense the eigenvalues inform us of intensity changes in each direction, such that:

- if λ_1 and λ_2 large then there is a corner ($R > 0$)
- if $\lambda_1 << | >> \lambda_2$ then there is an edge ($R < 0$)
- if both are close to 0, then it is a flat region ($R \approx 0$)

Where $R = Det(M) - ktrace(M)^2$

The harris detector is variant to scale, invariant to photometric changes and covariant to geometric (rotational) changes (meaning that the points remain consistent relative to the image).

1.2.1 Harris: computational steps

1. Calculate gradient of image in x and y directions
2. Calculate I_x^2 and I_y^2 and $I_x I_y$ (M matrix)
3. Gaussian blur the image
4. Calculate R for each point using step 2 results
5. Choose points with R ¿ 0
6. Perform non-maximal suppression to localise

1.3 Difference of Gaussians (SIFT)

1.3.1 Background: Laplacian of Gaussian (Blob detector)

The laplacian of gaussian pics up edge content better than a first order derivative, and traditionally is enhanced to be scale invariant by applying gaussian filters of different sizes to the image first and looking for blobs that are minima/maxima across both space and scale (scale-space). This is a computationally expensive process, and can be estimated using a Difference of Gaussians approach.

1.3.2 Difference of Gaussians approximation

$$Lap = \sigma^2(G_x x(x, y, \sigma) + G_y y(x, y, \sigma)) \approx G(x, y, k\sigma) - G(x, y, \sigma)$$

Steps:

- Empirically, typical values are $k = \sqrt{2}$ and $\sigma = \frac{1}{\sqrt{2}}$
- Apply gaussians with $k\sigma, k^2\sigma, k^3\sigma, ...$ to an image
- Downsample the image and repeat (each repeat is an octave)
- Subtract two consecutive scales to get the difference
- For each min/max found at a scale, compare to neigbouring scales in scale-space and perform non-maximal suppression
- (optional) Outlier rejection via thresholding
- (Make it orientation invariant) Calculate a histogram of gradients to show the direction of the strongest response in that region

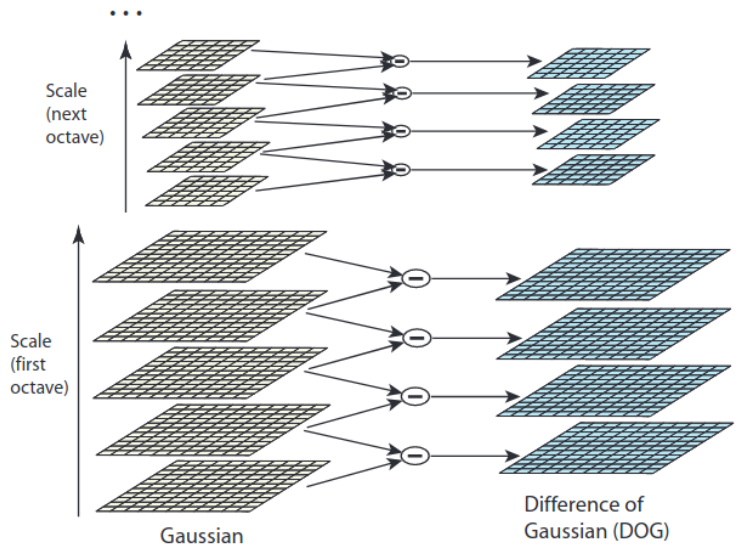

Figure 1: For each octave of scale space, the initial image is repeatedly convolved with Gaussians to produce the set of scale space images shown on the left. Adjacent Gaussian images are subtracted to produce the difference-of-Gaussian images on the right. After each octave, the Gaussian image is down-sampled by a factor of 2, and the process repeated.

2 Feature Extraction

2.1 Overview

Feature extraction is the task of extracting the summary statistics of the images and present them in a compact form. It can be done globally (dense) or locally (around interest points). We care about the scope and the content of the features. It has applications in: Image matching, object recognition, object tracking, scene comparison, stereo calibration, robot navigation, image retrieval...

Goal of feature extraction:

- Repeatable, invariant to transform
- Discriminates enough
- Compact dimensionality
- Computationally efficient (corollary to above)

2.2 SIFT Descriptor

For each interest point:

- Compute 16x16 patches of magnitude and orientation
- Compute HOG for each 4x4 region within the 16x16 (?)
- results in a 128D vector

2.3 Self Similarity Descriptor

Looks for repetition in an image. Steps:

- Take a patch (from interest point, or randomly) and scan it over the image or interest region of the image
- Record the correlation at each point
- Calculate the log-polar plot and obtain vector representation (can then be used to compare against other images)
- Repeat for all patches

2.4 LBP and LBP-TOP

Good for faces and textures. Steps:

- Divide image into blocks
- For each pixel in each block create an 8 bit score where each bit corresponds to a neighbouring pixels thresholded intensity, 1 if neighbour ¿ subject, 0 otherwise.
- Compute the histogram for each block to get the representation for that block
- stack histograms together to preserve spatial information.
- (repeat for different neighbourhood scales to make it scale invariant)
- (LBP-TOP is for video; take 3 orthogonal planes XY, XZ, YZ and compute the same for each)

3 Matching features

A simple / naive method is to search for matches between groups of interest point regions on two images. A slighly improved version is to do this, but look for the top two matches (for each point) and if the relative similarity is ¿ 0.8 reject the candidate match.

4 Image Alignment

The task of finding parameters of a model (e.g. entries in a matrix) that maps one set of points to another. Tricky in image processing due to noise, outliers, many-to-one object matching.. Basic transforms include scale, shear, rotation, translation, affine, and projective.

4.0.1 Example: Panorama stiching

1. Compute interest points and match
2. Compute H (homography matrix) to warp one image to another using matched interest points on edges. use RANSAC method to select the best H:
   (a) Select set of matched points randomly
   (b) Compute the transform for this group
   (c) Find the nearest neighbours (the inliers) and count them up
   (d) Repeat the process n times
   (e) Choose the H that yields the most inliers
3. Warp the image using H