SIFT: Scale-invariant feature transform

It is an algorithm to detect and describe local features in images. The algorithm was published by David Lowe in 1999.

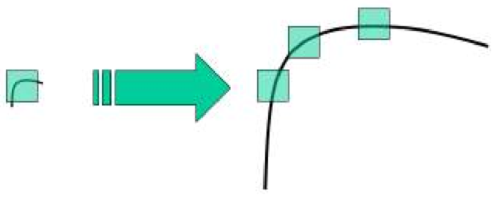
The algorithm is patented in the US; the owner is the University of British Columbia.

The SIFT algorithm extract keypoints and compute its descriptors. (This explanation is just a short summary of this paper). Like previous works (e.g. Harris corner detector) it is rotation-invariant, which means, even if the image is rotated, we can find the same corners. It is obvious because corners remain corners in rotated image also.

There are mainly four steps involved in SIFT algorithm. We will see them one-by-one.

Detection of scale-space extrema

We can't use the same window to detect keypoints with different scale. Larger corners require larger windows.

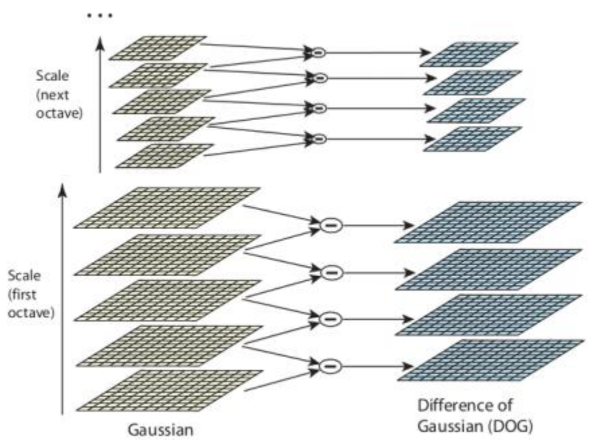


Scale-space filtering is used. In it, Laplacian of Gaussian is found for the image with various σ values. LoG acts as a blob detector which detects blobs in various sizes due to change in σ. σ acts as a scaling parameter.

So, we can find the local maxima across the scale and space which gives us a list of (x,y,σ) values which means there is a potential keypoint at (x,y) at σ scale. But LoG is costly, so SIFT algorithm uses Difference of Gaussians which is an approximation of LoG. Difference of Gaussian is obtained as the difference of Gaussian blurring of an image with two different σ, let it be σ and kσ. This process is done for different octaves of the image in Gaussian Pyramid. It is represented in next image.

Lowe purposes the following formula for image detection:

$$D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) = L(x, y, k\sigma) - L(x, y, \sigma).$$



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.

Maxima and minima of the difference-of-Gaussian images are detected by comparing a pixel (marked with X) to its 26 neighbors in 3x3 regions at the current and adjacent scales (marked with circles).

Regarding different parameters, the paper gives some empirical data which can be summarized as, number of octaves = 4, number of scale levels = 5, initial $\sigma=1.6$, $k=\sqrt{2}$ as optimal values.

Local extrema detection - Keypoint Localization

Once potential keypoints locations are found, they have to be refined to get more accurate results.

They used Taylor series expansion of scale space to get more accurate location of extrema, and if the intensity at this extrema is less than a threshold value (0.03 as per the paper), it is rejected. This threshold is called contrastThreshold in OpenCV

DoG has higher response for edges, so edges also need to be removed. For this, a concept similar to Harris corner detector is used. They used a 2x2 Hessian matrix (H) to compute the pricipal curvature. We know from Harris corner detector that for edges, one eigen value is larger than the other. So here they used a simple function.

If this ratio is greater than a threshold, called edgeThreshold in OpenCV, that keypoint is discarded. It is given as 10 in paper.

So it eliminates any low-contrast keypoints and edge keypoints and what remains is strong interest points

Orientation Assignment

Now an orientation is assigned to each keypoint to achieve invariance to image rotation. A neigbourhood is taken around the keypoint location depending on the scale, and the gradient magnitude and direction is calculated in that region. An orientation histogram with 36 bins covering 360 degrees is created. (It is weighted by gradient magnitude and gaussian-weighted circular window with σ equal to 1.5 times the scale of keypoint. The highest peak in the histogram is taken and any peak above 80% of it is also considered to calculate the orientation. It creates keypoints with same location and scale, but different directions. It contribute to stability of matching.

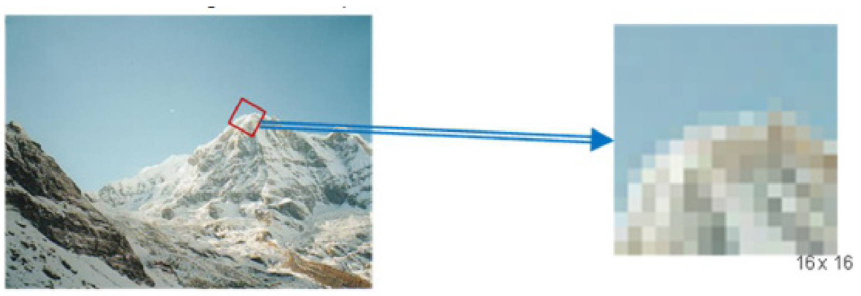
Keypoint Descriptor

Now keypoint descriptor is created. A 16x16 neighbourhood around the keypoint is taken. It is devided into 16 sub-blocks of 4x4 size. For each sub-block, 8 bin orientation histogram is created. So a total of 128 bin values are available. It is represented as a vector to form keypoint descriptor. In addition to this, several measures are taken to achieve robustness against illumination changes, rotation etc.

Histograms of Oriented Gradients (HOG) based descriptors

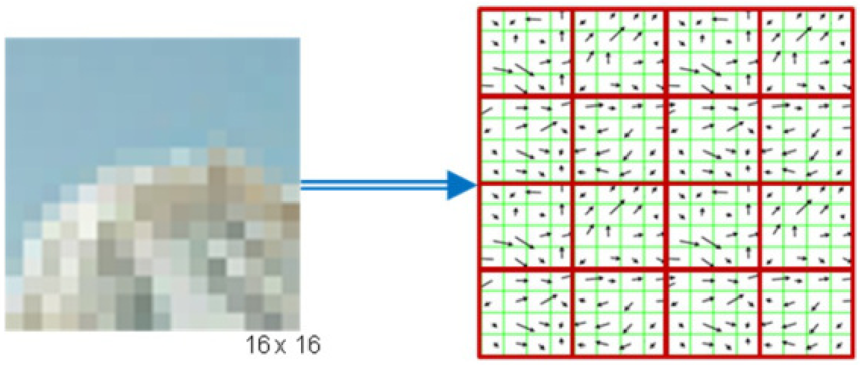
1. First, detect keypoints using a detector, which also detects scale and orientation of the keypoint.

2. Next, for a given keypoint, warp the region around it to canonical orientation and scale and resize the region to a squre of pixels.

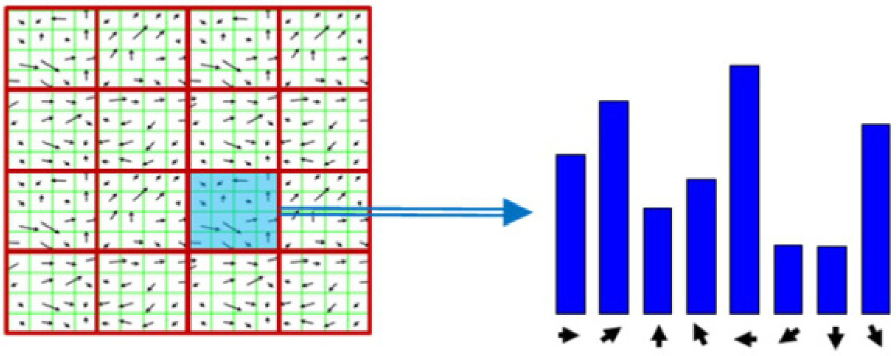


3. Compute the gradients for each pixels (orientation and magnitude).

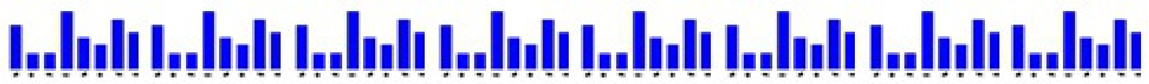
4. Divide the pixels into 16, 4X4 pixels squares.



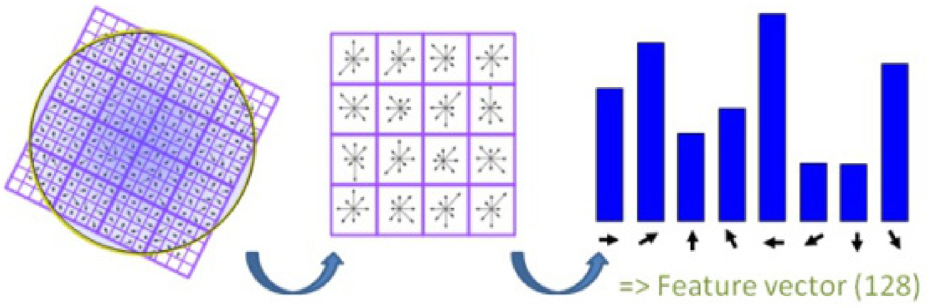
5. For each square, compute gradient direction histogram over 8 directions



6. concatenate the histograms to obtain a 128 (16*8) dimensional feature vector:



SIFT descriptor illustration:



SIFT is invariant to illumination changes, as gradients are invariant to light intensity shift. It's also somewhat invariant to rotation, as histograms do not contain any geometric information.

Keypoint Matching

Keypoints between two images are matched by identifying their nearest neighbours. But in some cases, the second closest-match may be very near to the first. It may happen due to noise or some other reasons. In that case, ratio of closest-distance to second-closest distance is taken. If it is greater than 0.8, they are rejected. It eliminates around 90% of false matches while discards only 5% correct matches, as per the paper.