SIFT features

DEPARTMENT OF ELECTRONICS AND ELECTRICAL COMMUNICATION ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY, KHARAGPUR

Joshua Peter Ebenezer

June 1, 2017

Abstract

This article aims to give a brief overview of SIFT features, what they are, their properties, and how to find them

Contents

1 Introduction

SIFT features were introduced by David G. Lowe back in 2004 [1], and have found wide applications in many areas. SIFT features are invariant to image scale and rotation, and fairly robust to affine distortion, change in 3D viewpoint, addition of noise, and change in illumination. SIFT stands for 'Scale invariant feature transform' and has been patented by the University of British Columbia in the US.

2 Motivation

Interest points are features in an image that are considered to be distinctive enough to be used for image to image correspondence, which allows object detection and tracking. A point is classified as an interest point, if

- it has a clear, preferably mathematically well-founded, definition,
- it has a well-defined position in image space,
- the local image structure around the interest point is rich in terms of local information contents (e.g.: significant 2D texture), such that the use of interest points simplify further processing in the vision system,
- it is stable under local and global perturbations in the image domain as illumination/brightness variations, such that the interest points can be reliably computed with high degree of repeatibility.
- The notion of interest point should include an attribute of scale, to make it possible to compute interest points from real-life images as well as under scale changes.

As the name suggests, interest points are points that are 'interesting' about the image, or that stand out from the image. Interest points are also known as keypoints, and these two terms will be used interchangeably in the article. They can be used in a variety of ways, because identifying an object in an image from a given training set opens up many possibilities. Consequently, a lot of research has gone into describing the properties of keypoints mathematically, as well as detecting them.

3 The Algorithm

3.1 Overview

The algorithm is presented step by step, in the following manner:

1. **Scale space peak detection**: Potential locations for finding features
2. **Key point localization**: Accurately locating the key points
3. **Orientation assignment**: Assigning an orientation to each key point
4. **Key point descriptor**: Describing each keypoint by a 128 dimension vector

3.2 Scale space peak detection

3.2.1 Generating the scale space

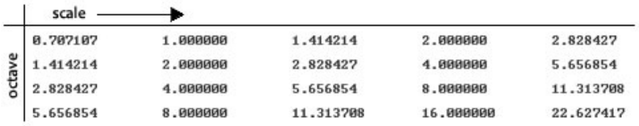
The first stage in the algorithm is to find the location and the scale at which interest points occur. This ensures that the feature points are scale invariant. This can be accomplished by searching for stable features across all scales, in the scale space. If for example, we have an image of a tree, and details such as its leaves and twigs are visible, but we do not need features at that scale, and we only need to see the tree. We would blur the image with a kernel that does not introduce details that weren't there earlier, but removes the fine details of the leaves and the twigs. It has been shown that the Gaussian kernel is the only possible scale-space function. The scale here refers to the standard deviation σ of the Gaussian function. A larger σ value would imply a bigger kernel size for the Gaussian, and blur out finer details because it acts as a low pass filter with a higher cut-off frequency than a Gaussian with a lower σ value.

Zero-crossings are found as points at which the second spatial derivative becomes zero. A zero-crossing at the $(n+1)$th derivative is identified as an extrema of the nth derivative, where the $(n+2)$nd derivative is non-zero. Here we deal with n=1, which corresponds to inflection points in the image. Different values of σ give different location of zero-crossings in the image, and we sweep the space (i.e. across the image) as well as the value of σ to generate a scale-space description.

As σ increases, zero-crossings at smaller σ values may disappear. However, as σ decreases, new zero-crossings may appear, but existing zero-crossings cannot disappear. The Gaussian function is the only function that satisfies both this condition and the condition of well-behavedness.

To generate the scale-space description, we first generate several 'octaves' of the image. To construct an octave, we subsample the image at the lower octave, with the lowest octave having the original image. Every second pixel is sampled, both row-wise and column wise, to construct an image that is half the size of the image in the lower octave in terms of each dimension, and four times smaller in terms of total number of pixels.

If the blur (scale) in a particular image is σ, then the scale in the next image in that same octave is $k\sigma$, where k is an empirical constant chosen as $\sqrt{2}$. The image in the octave immediately above the current octave has a lower resolution, so it will be more sensitive to filtering. Hence the effect of smoothing the image in the higher octave by a filter with the scale σ would actually be equivalent to smoothing the image at the lower octave with a filter of scale $\sigma/2$, because the lower the scale the sharper the filter. So the first image in each octave is filtered with a filter of scale $k^2\sigma$ (i.e. 2σ), where σ is the scale used to filter the first image of the lower scale.



	scale →				
octave	0.707107	1.000000	1.414214	2.000000	2.828427
	1.414214	2.000000	2.828427	4.000000	5.656854
	2.828427	4.000000	5.656854	8.000000	11.313708
	5.656854	8.000000	11.313708	16.000000	22.627417

Figure 3.1: Scales in octaves

Any number of octaves and scale levels within each octave can be used, and this depends on the size of the image, but Lowe suggests that four octaves and five blur(scale) levels be used. So the scale of the filter at the highest image in an octave would differ from the scale at the lowest image in that same octave by a factor of k^5.

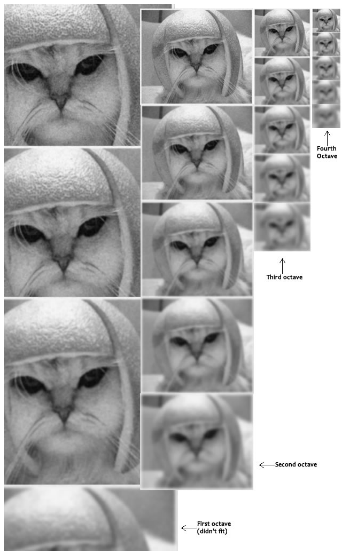


Figure 3.2: Different octaves for an image

3.2.2 Estimating the Laplacian of the Gaussian

Therefore, the scale-space of an image is defined as the function, $F(x,y,\sigma)$, that is the convolution of a Gaussian $G(x,y,k\sigma)$ at a variable scale σ with an image $I(x,y)$.

$$F(x,y,\sigma) = G(x,y,\sigma) * I(x,y) \tag{3.1}$$

Finding the zero-crossings involves finding the zeros of the second spatial derivative of the image, or the Laplacian. The Laplacian $L(x,y)$ of an image $I(x,y)$ is given by

$$L(x,y) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} \tag{3.2}$$

Generally, the Laplacian of the Gaussian(LoG) is used, because the Laplacian is generally very sensitive to noise as it finds the second derivative, and the Gaussian is applied as a pre-processing step to smoothen the image and remove noise to some extent. Lindeberg [2] showed that the scale-normalized Laplacian of Gaussian (LoG) is required for true scale invariance. The Gaussian is represented as:

$$G(x,y,\sigma) = \frac{1}{2\pi\sigma^2} e^{\frac{x^2+y^2}{2\sigma^2}} \tag{3.3}$$

The σ^2 in the denominator introduces a scale dependency. Hence we normalize the Laplacian of the Gaussian, $\nabla^2 G$, by multiplying it with the scale dependent factor σ^2.

Finding the LoG can be an expensive operation, but an approximation can be used for the LoG instead. We define $D(x, y, k\sigma)$ as the difference in the responses of the image to the Gaussian function $G(x, y, k\sigma)$ and $G(x, y, \sigma)$, at scales that are a factor of k apart.

$$\begin{aligned} D(x, y, k\sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= F(x, y, k\sigma) - F(x, y, \sigma) \end{aligned} \tag{3.4}$$

Consider the analogue of the heat diffusion equation for the Gaussian, with σ as the independent variable

$$\frac{\partial G}{\partial \sigma} = \sigma \nabla^2 G \tag{3.5}$$

This can be approximated as

$$\frac{\partial G}{\partial \sigma} \approx \frac{G(x, y, k\sigma) - G(x, y, \sigma)}{k\sigma - \sigma} \tag{3.6}$$

It is thus seen that the scale normalized Laplacian of the Gaussian (LoG), can be approximated as the difference of the Gaussians (DoG).

$$G(x, y, k\sigma) - G(x, y, \sigma) \approx (k-1)\sigma^2 \nabla^2 G \tag{3.7}$$

The factor $(k-1)$ is the same for different scales and only depends on the factor k.

Now, as seen in (3.6), the LoG can be approximated as the DoG. We simply subtract images at a lower scale, from the image at the scale immediately above it in the same octave, and generate DoGs. The computationally intensive task of finding the Laplacian of the Gaussians is replaced by a subtraction, with scale-space normalization (multiplication of the Laplacian by σ^2) being taken care of by the DoG. The factor $(k-1)$ depends on the scales we are dealing with, but this is not very relevant because we are not actually looking for the absolute value of the extrema (zero-crossings of the DoGs) but only the locations of the extrema.

3.2.3 Finding the extremas

The extrema is located within a search space that has the 8 adjacent neighbours of a particular pixel, as well as the 9 pixels directly above it in terms of location in the scale space (i.e. on the scale that is k times higher than the scale at which the pixel is) and the 9 pixels below it in the scale space that has a scale k times lower than the current scale.

Therefore, a total of 26 pixels need to be searched. The validity of the choice of a 3 x 3 kernel, and 3 scales, is explored in the paper. In terms of scale, three adjacent scales are chosen because repeatability is a maximum at three scales. Repeatability is a measure of how many key-points in the original image find a correspondence in the image after it has been subjected to a range of transformations. Number of key-points increase as number of scales sampled increases, and this may be required

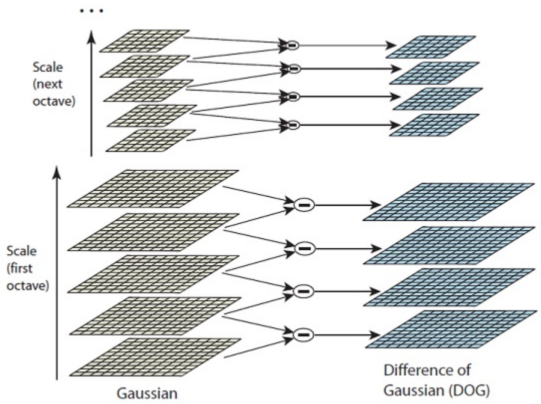


Figure 3.3: Finding the DoGs

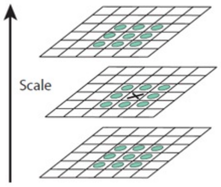


Figure 3.4: Finding the extremas

for some applications, but three scales are chosen for computational efficiency. In terms of spatial frequency, there is no minimum spacing of sampling that will detect all extrema, because extrema can be arbitrarily close. In any case, extrema that are very close to each other are unstable to perturbations. It is found that repeatability increases with σ, but using large values of σ significantly affects computational cost. The value of σ is chosen as 1.6, which provides close to optimal repeatability.

3.3 Accurate key-point localization

3.3.1 Rejecting minor extrema

The next step would be to fit the key-points found to nearby points so as to find the location, scale and ratios of the principal curvatures of the key-points. This would allow us to reject certain points on the basis of various criteria.

An approach suggested by Brown and Lowe [3] was to fit a quadratic 3D function to the key-point and points near it, so as to find the exact location of the maximum. The maximum may not exactly be at the location that was found by the search described in the previous section. It may be in-between pixels and may have to be interpolated. This is done using a first order Taylor Series expansion of the scale space function $F(x, y, \sigma)$ about the key-point.

$$F(\mathbf{x}) = F + \frac{\partial F^T}{\partial \mathbf{x}}\mathbf{x} + \frac{1}{2}\mathbf{x}^T \frac{\partial F}{\partial \mathbf{x}}\mathbf{x} \tag{3.8}$$

Taking the derivative of this function and setting it to zero gives the location of the extremum as

$$\hat{\mathbf{x}} = -\frac{\partial^2 F^{-1}}{\partial \mathbf{x}^2}\frac{\partial F}{\partial \mathbf{x}} \tag{3.9}$$

If the offset $\hat{\mathbf{x}}$ is more than 0.5 (i.e. half a pixel), then it means the extremum should lie at another pixel, and the Taylor series expansion is performed about that point. If the value obtained in (3.9) is plugged into (3.8), then the value obtained for F is

$$F(\hat{\mathbf{x}}) = F + \frac{1}{2}\frac{\partial F^T}{\partial \mathbf{x}} \tag{3.10}$$

In the paper, all extrema with values of $F(\hat{\mathbf{x}})$ lesser than 0.03 are discarded.

3.3.2 Rejecting edges

The DoG function will give a strong response along edges, though locations across edges are poorly determined and sensitive to noise. Edges will thus need to be removed. Such a poorly defined point will have a curvature that is large across the edge, but small along it. The principal curvature can be found from the 2 x 2 Hessian applied at the location and scale of the interest point.

$$\mathbf{H} = \begin{bmatrix} F_{xx} & F_{xy} \\ F_{yx} & F_{yy} \end{bmatrix} \tag{3.11}$$

Derivatives are taken as nearest neighbour differences.

The eigenvalues of the Hessian are proportional to the curvatures of F. Since we are only concerned with the ratio of the eigenvalues, and not the absolute values, we can find it borrowing concepts from Harris and Stephens' [4] corner detector. Let the larger eigenvalue of the Hessian be α and the smaller one β. Then,

$$\mathrm{Tr}(\mathbf{H}) = \alpha + \beta \tag{3.12}$$

$$\mathrm{Det}(\mathbf{H}) = \alpha\beta \tag{3.13}$$

If the ratio of α and β is r, then

$$\begin{aligned} \frac{\mathrm{Tr}(\mathbf{H})^2}{\mathrm{Det}(\mathbf{H})} &= \frac{(\alpha+\beta)^2}{\alpha\beta} \\ &= \frac{(r+1)^2}{r} \end{aligned} \tag{3.14}$$

The minimum value of $(r+1)^2/r$ occurs when $r = 1$, and is 4. As r increases, the value of the function increases. Hence, some threshold value can be chosen for r, say r_0, and key-points for which the Hessian does not satisfy

$$\frac{\mathrm{Tr}(\mathbf{H})^2}{\mathrm{Det}(\mathbf{H})} < \frac{(r_0+1)^2}{r_0} \tag{3.15}$$

are rejected as edges. r_0 is chosen as 10 in the paper.

3.4 Orientation assignment

The gradient magnitude and the orientation are found in the following way:

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2} \tag{3.16}$$

$$\theta(x,y) = \tan^{-1}[\frac{L(x,y+1) - L(x,y-1)}{L(x+1,y) - L(x-1,y)}] \tag{3.17}$$

The 360° possible values the orientation angle can take are divided into 36 bins of 10° each. The magnitude and orientation are calculated for all the pixels around the key-point, in a window with size 1.5σ, where σ is the size of the scale at which the key-point is located. Also, the intensity values used to find the gradient are first smoothed by a Gaussian filter of scale 1.5σ. A histogram is created, with the bins grouping angles, and being weighted by their magnitudes. A peak will be found for a particular bin representing a particular range of angles that is 10° in width. The key-point is assigned this bin as its orientation. It may so happen that a secondary peak occurs in the histogram, that is within 80% of the value of the first peak. In that case, another key-point is created, and assigned the secondary peak as its orientation. Hence, a single key-point may be split into two if it has a secondary peak that is within 80% of the first peak.

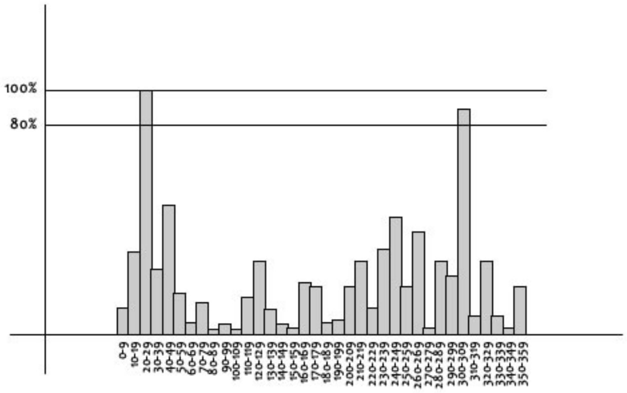


Figure 3.5: Splitting a single key-point

3.5 Key point descriptor

Now the key-points need to be assigned a descriptor. First, we consider a 16 x 16 window around each key-point. Each such window is further divided into 16 4 x 4 blocks. The gradient magnitudes and orientations for every pixel in the 4 x 4 blocks are calculated, and stored in 8 bin histograms. Each bin represents a range of angles between 0° and 360 °, and is of equal width. However, we weight the magnitudes by a Gaussian function centred at the key-point, with scale σ equal to one half of the length of the descriptor window, or in other words it would be a function that tapers off at the edges of the 16 x 16 window around the key-point. To achieve rotational invariance, coordinates of the descriptor and the orientations of all the pixels surrounding the key-point are rotated relative to the key-point orientation.

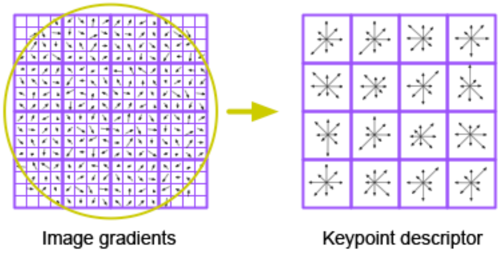


Figure 3.6: Finding the descriptor

Now we have 8 bin histograms for each 4 x 4 block, and we have a total of 16 such blocks around each key-point. All of these are concatenated to create a 128 bin vector to describe each key-point. This vector is normalized to make it robust to illumination changes. Illumination changes may change all intensity values by some factor, but this will not change the description of the key-point as it is normalized. Non linear illumination changes are taken care of by thresholding the values of the normalized unit vector to each be no larger than 0.2, and then re-normalizing the vector to unit length.

References

[1] Lowe, D.G, "Distinctive image features from scale-invariant keypoints', International Journal of Computer Vision, Volume 60 Issue 2, November 2004, Pages 91 - 110

[2] Lindeberg, T., 1994, "Scale-space theory: A basic tool for analysing structures at different scales", Journal of Applied Statistics, 21(2):224-270.

[3] Brown, M. and Lowe, D.G. 2002, "Invariant features from interest point groups". In British Machine Vision Conference, Cardiff, Wales, pp. 656-665.

[4] Harris, C. and Stephens, M., 1988, A combined corner and edge detector, In Fourth Alvey Vision Conference, Manchester, UK, pp. 147-151.