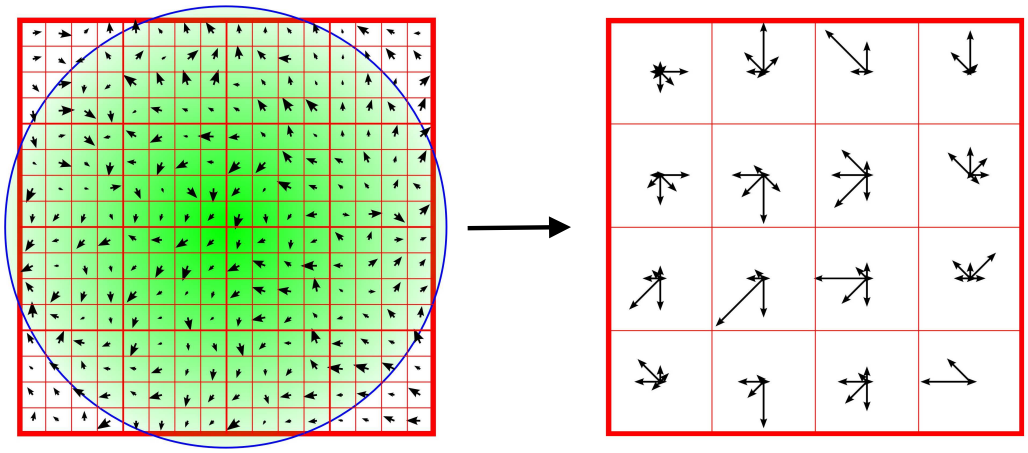
University of Cambridge
Engineering Part IB
Paper 8 Information Engineering

Handout 3: Feature Descriptors

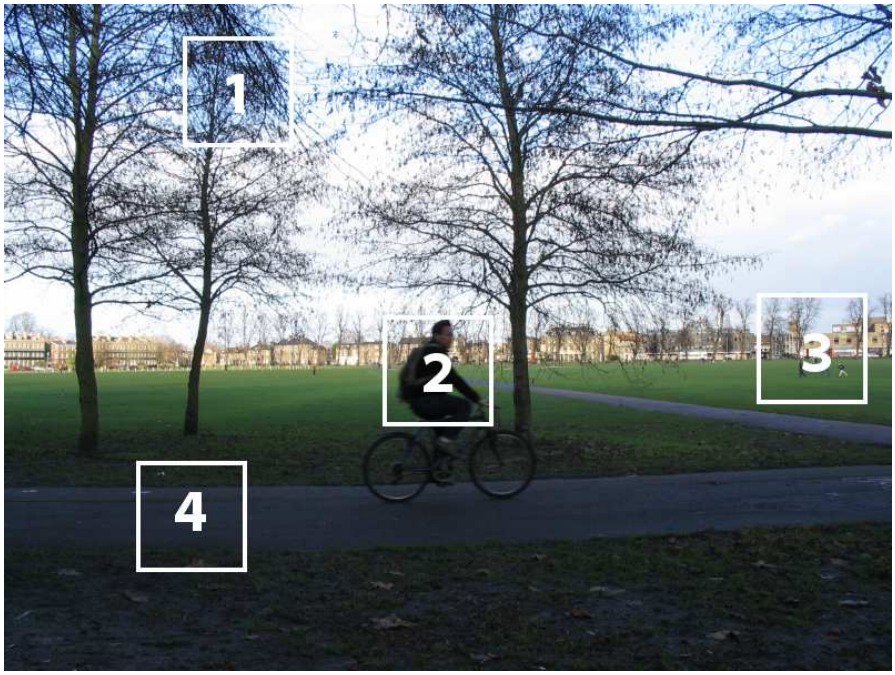


Roberto Cipolla
May 2013

Representing Image Patches

Image patches are distinct and interesting, and take many forms. For example, what is the total number of 100 by 100 pixel patches possible?

$\left(255^3\right)^{10000}$ = a really big number



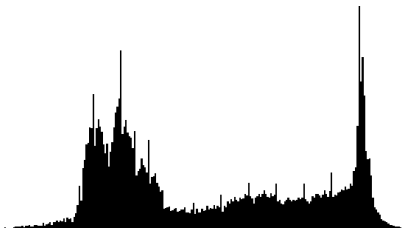
1 2 3 4

Even though the number of patches we can expect to see in natural scenes is much smaller, how can we store information about a patch which represents its distinctive nature?

Histograms of Intensities

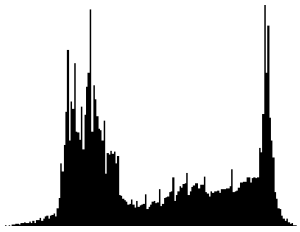
The obvious first idea is to use an intensity histogram to describe the patch. The patch is described using a set of 256 bytes, which is certainly very compact. However, intensity histograms vary considerable when the patch is changed slightly.

Original patch



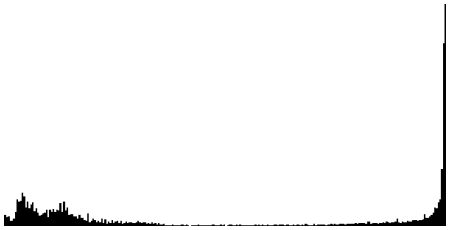
Intensity histogram

Brightness change



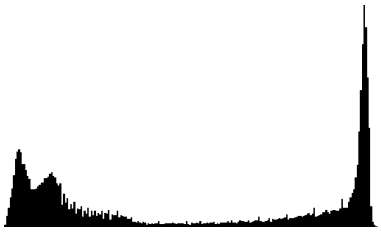
Shifted to the left

Contrast change



Stretched

Various changes



Just different

Feature Descriptors

So, in order to be effective, the description must be robust to changes in the way the patch looks. These descriptions, called **feature descriptors** are n-tuples which represent information about an area of an image. It is in essence a targeted data reduction which gives particular information about an area in a compact form.

Good Properties

In image matching and retrieval, feature descriptors are calculated using scaled areas around interest points and are utilized to find a correspondence between the points in one image and another.

For this purpose, a few properties are desirable:

Robust: Small changes in the following should have little effect:

- Contrast
- Intensity
- Colour
- Resolution

Ideally, small location and perspective changes should also have little effect.

Distinctive: Different patches have different descriptors.

Good Properties

Robustness

A **robust** feature descriptor computed for a patch will be the same even if the patch undergoes certain transformations. The image which is "seen" by a camera and then read by a digital computer is dependent upon a large range of variables, not just those of the lens but also lighting conditions in the environment and the encoding method used, among others. A feature descriptor which exhibits large changes for these conditions is thus not useful for matching between scenes represented in different images.

The way to achieve this robustness is to utilize interest points in computing the descriptor as opposed to the raw image data. Of particular use is the edge-map of an image, which is robust to changes in contrast, colour and intensity.

Good properties

Distinctiveness

In order for a feature descriptor to be **distinctive**, it must be possible for the space in which it lives to be populated uniformly by descriptors computed from different patches. To say it differently, if you chose a random number of image regions and computed the descriptor over those, they would be uniformly distributed over the descriptor space.

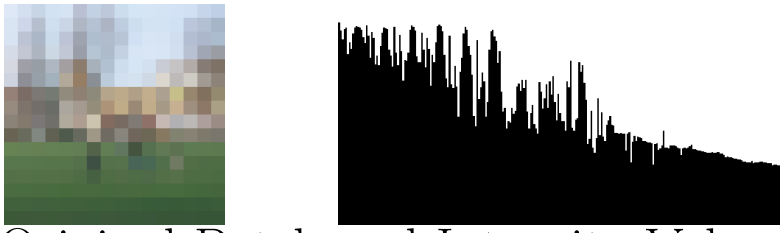
This property is difficult to achieve, with the usual recourse being to use a large dimensional descriptor. The ultimate goal is to make it so that different patches have different descriptors, something which sounds easy but is in reality quite different. Perhaps more importantly, the amount of difference between descriptors must relate directly to the amount of difference between patches.

Intensity Patches

The simplest way to describe an area of an image is just to store the intensity values. You can then compare patches directly using cross-correlation against other patches to find a match.

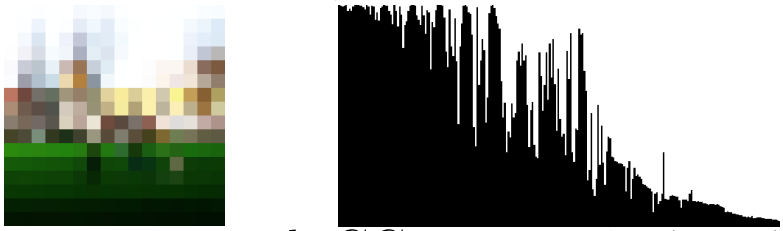
$$CC(P_1, P_2) = \sum_{i}^{N} P_1[i]P_2[i]. \qquad (1)$$

In this raw form it is not very robust to changes.

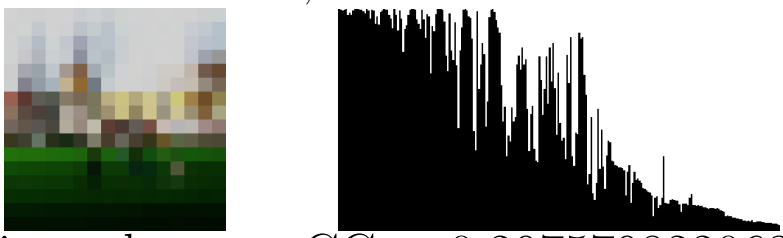


Original Patch and Intensity Values

Brightness Decreased, CC = 0.262720397078039



Contrast increased, CC = 0.380413705374859



Various changes, CC = 0.297579822063629

Zero Normalized Patches

Brightness changes are essentially changes in the mean brightness value. While the mean changes, the distribution of the intensity values around the mean stays the same. Thus, by giving the intensity values a zero mean, they become immune to brightness change.

$$\mu = \frac{1}{N} \sum_{x,y} I(x,y)$$
$$Z(x,y) = I(x,y) - \mu$$

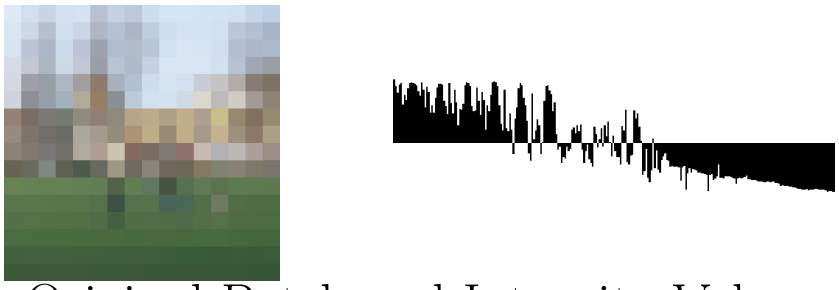
However, the intensity values are still affected by contrast changes. Contrast is essentially a change in the variance of the distribution of the intensity values around the mean.

Thus, to deal with contrast all that is required is to divide each value by the standard deviation of the intensity value distribution.

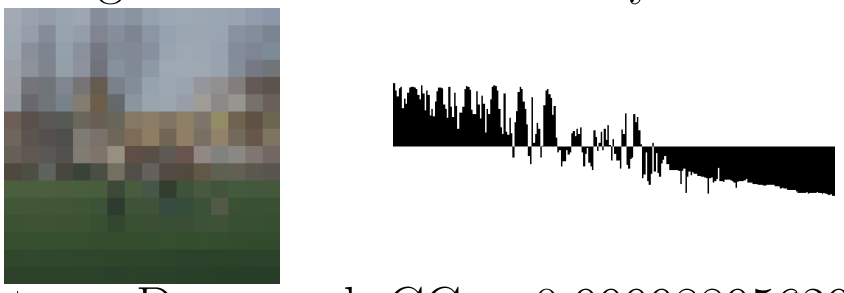
$$\sigma^2 = \frac{1}{N} \sum_{x,y} Z(x,y)^2$$
$$ZN(x,y) = \frac{Z(x,y)}{\sigma}$$

Zero Normalized Patches, cont.

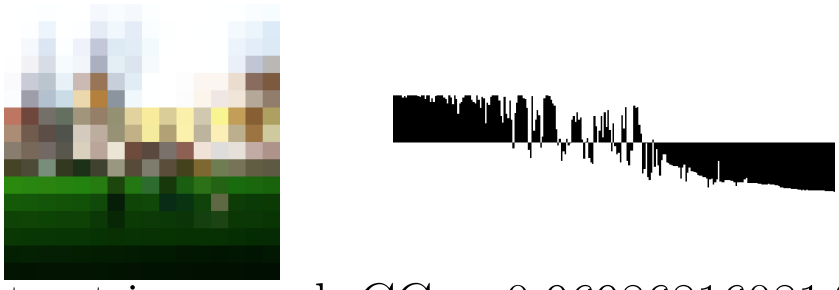
The resulting collection of zero-mean, unit variance intensity values is known as a zero-normalized patch, and can be accurately matched using simple cross-correlation. The size of the descriptor grows with the size of the patch, and thus can be quite big. Even so, while not a data reduction, it is a useful way to represent these areas.



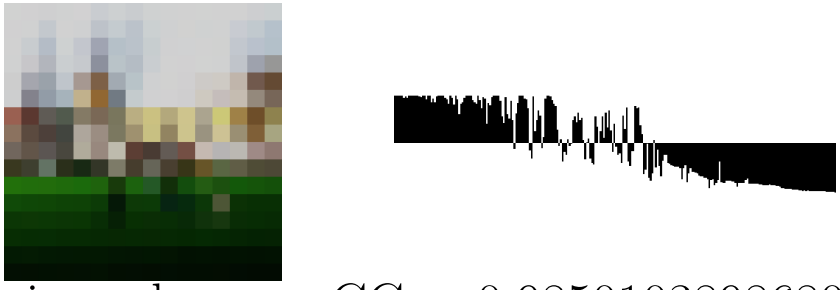
Original Patch and Intensity Values



Brightness Decreased, CC = 0.999988956295594

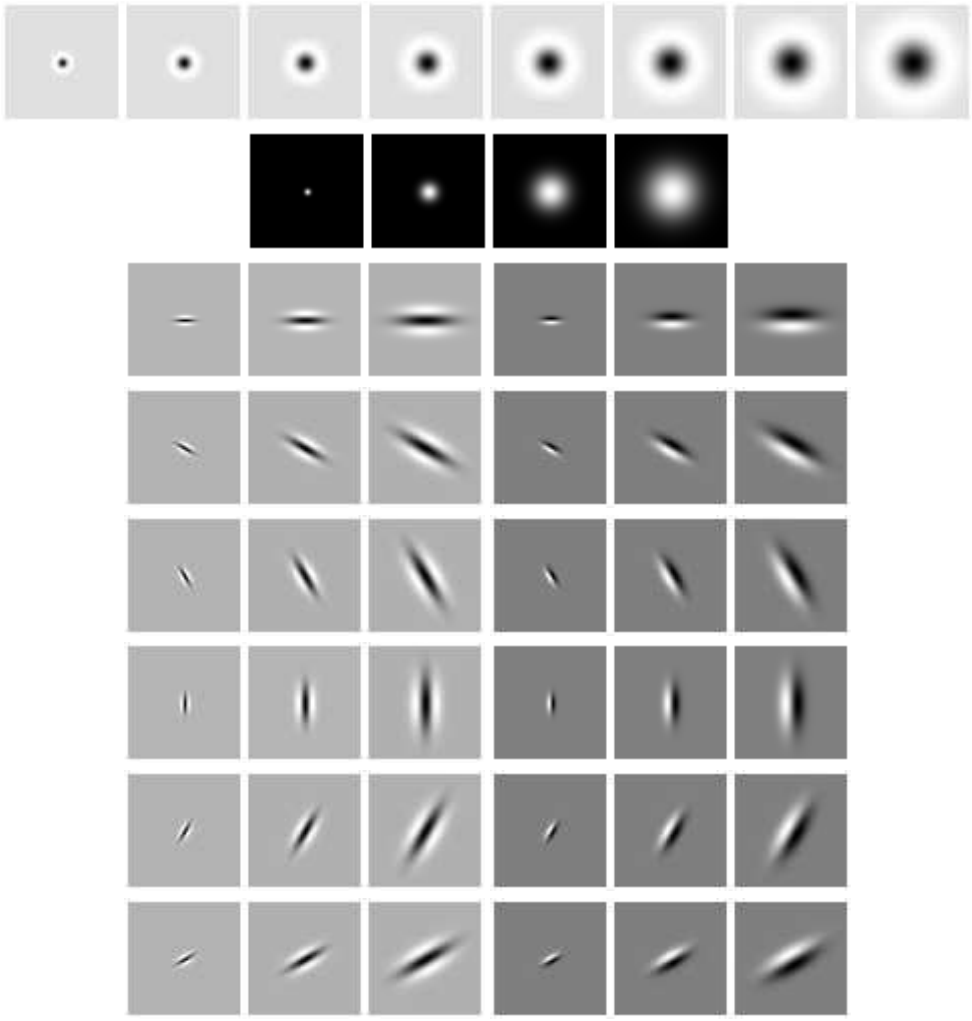


Contrast increased, CC = 0.969868160814465



Various changes, CC = 0.985010389868036

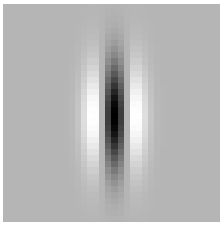
Filter Banks



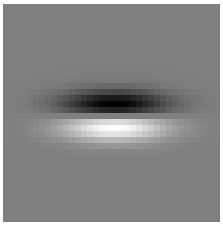
This is an example filter bank. It consists of 8 Laplacian of Gaussian filters and 4 Gaussian filters at different scales to provided non-oriented responses, and 36 oriented filters at 6 different angles, 3 different scales, and 2 different phases. The two phases of oriented filters are first and second derivatives of Gaussians on the minor axis and elongated Gaussians on the major axis, and thus detect edges or bars respectively along their major axes.

The descriptor is simply the concatenated responses of all of the filters in the filter bank at a pixel, and is particularly adept at richly encoding texture.

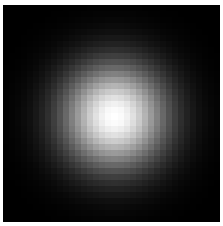
Filter Banks



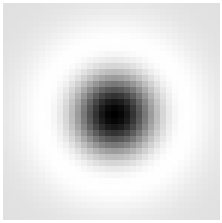
Bar



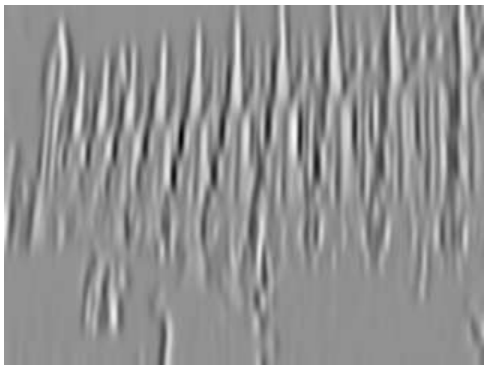
Edge



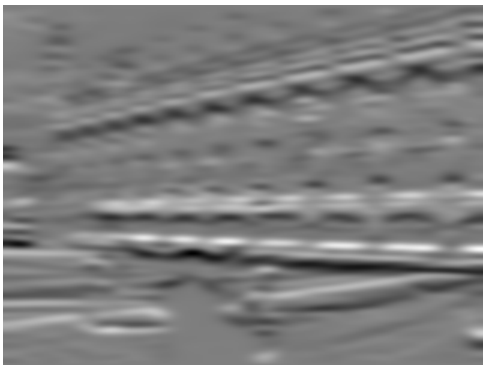
Brightness



Blob

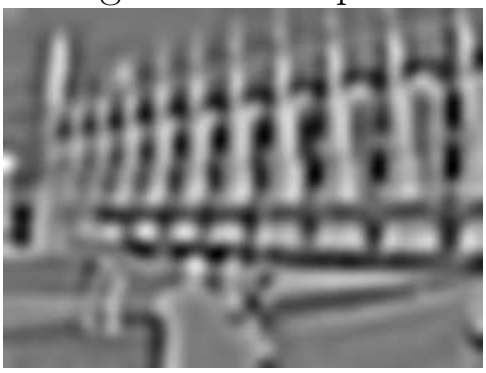


Bar Filter Response



Edge Filter Response

Intensity Filter Response



Blob Filter Response

Filter Banks, cont.

Since filter banks respond to basic image features such as blobs, edges, bars, and average brightness, they are innately immune to most changes in an image.

Original patch

Filter Bank Response

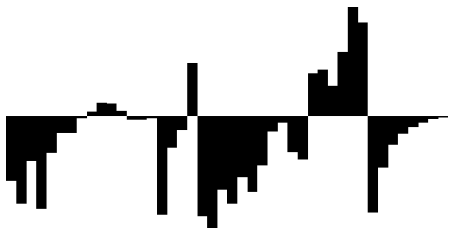
Brightness change

No change

Contrast change

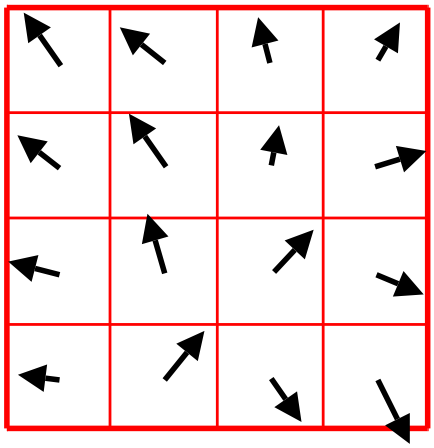
Small Change

Various changes



Medium Change

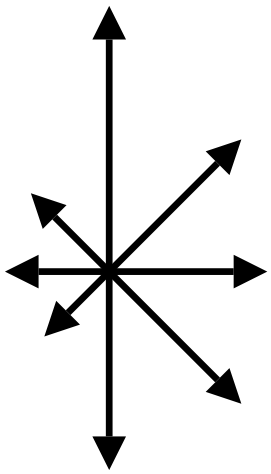
Orientation Histograms



Gradient Grid

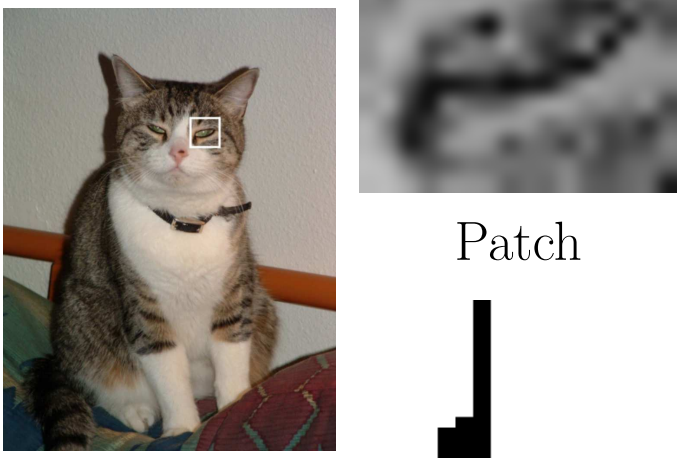
If you look at every edge in a patch of pixels, you will find that each has a distinct orientation, or way that it is facing. If you look at all of these and weight them by the strength of the edge, you get something that looks like this patch.

These can in turn be binned together into an orientation histogram, as shown on the right. Since this histogram is built using edges, which are robust to contrast and brightness changes and can be detected at different scales, and also incorporates orientation data (thus adding robustness to orientation) this makes them a very strong candidate for a descriptor.



Dominant Orientation

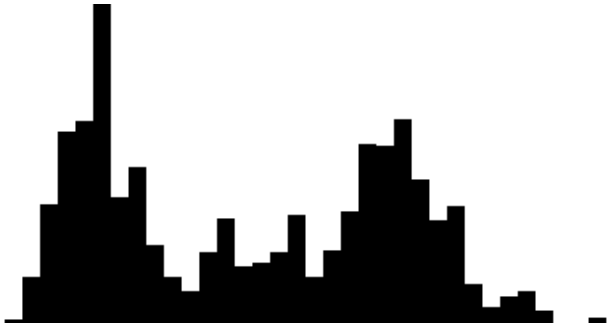
We find the dominant orientation by way of an **edge histogram**. This is a formed by binning all of the edge orientations in the neighborhood of an interest point. The maximum bin of this histogram is an approximation of the true dominant orientation, with the true value found through interpolation (by fitting a parabola to the values of the bin and its two neighbors). If there is no clear maximum, then the interest point is given several dominant orientations (i.e. several copies of the interest point with different orientations are used.)



Source Image

Patch

Edge Detection

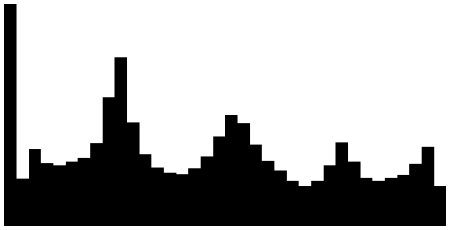


Orientation Histogram

Orientation Histograms

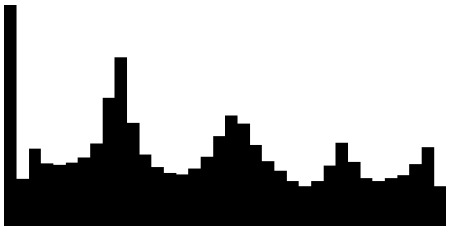
The direction of the edge will always be the same regardless of the strength of the edge. Thus, edges which are detectable always end up in the same bins. In order to deal with changes in edge intensity, the descriptor is normalized. The descriptors shown below have 36 bins.

Original patch



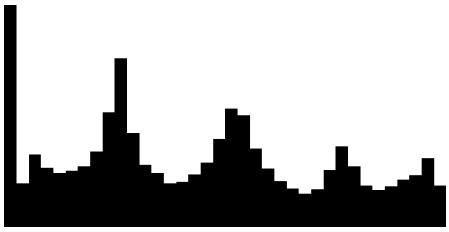
Orientation Histogram

Brightness change



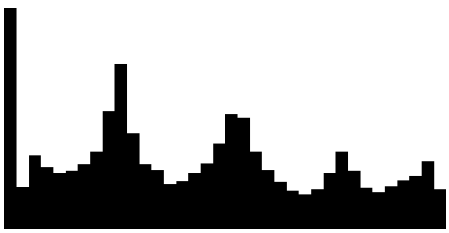
No change

Contrast change



No change

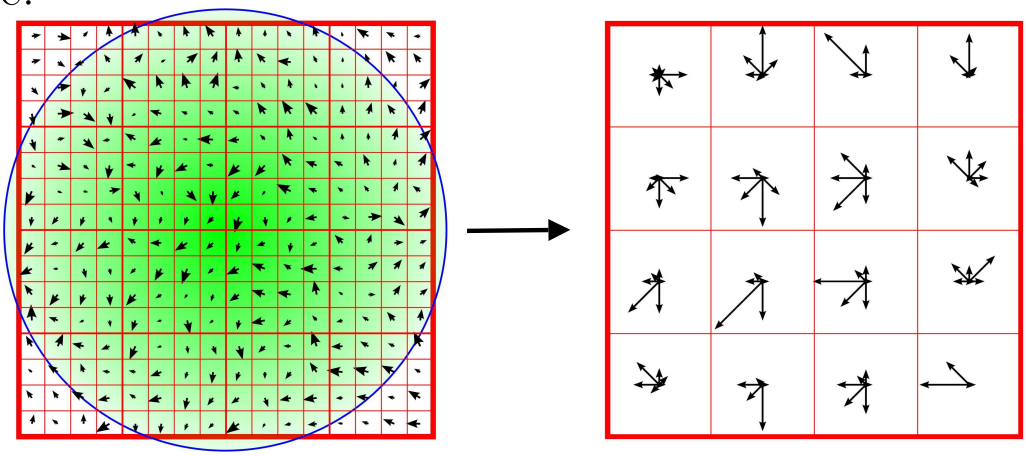
Various changes



No change

SIFT

SIFT stands for **S**cale-**I**nvariant **F**eature **T**ransform. It uses a collection of orientation histograms to create a robust and descriptive representation of a patch. This $N \times N$ patch (typically, $N = 16$) is extracted at the scale of the interest point. Thus, while the patch size never changes the area of the actual image it represents changes depending on scale.



The $N \times N$ patch is split into c cells, and within each cell the intensity gradient at every pixel is calculated and the directions binned into a histogram weighted by their magnitude and a Gaussian window with a σ of .5 times the scale of the feature centered on the patch. This weights the inner pixels (those closer to the interest point) to avoid possible occlusion problems.

SIFT

If the bins are centered on d directions (typically 8) in each of c cells (typically 16) , the resulting descriptor is a $d \times c$ vector (typically **128**D).

By dividing the patch into cells, a particular gradient can move around to some degree within the descriptor window and still contribute to the same directional histogram. Once the $d \times c$ vector has been extracted, it is normalized to provide invariance to gradient magnitude change. One final step is performed to help minimize the effects of non-affine lighting changes by thresholding all values in the unit vector to .2 and then renormalizing.