1 State of the art

Since few years color image processing is a major problem, indeed lot of the colour texture discrimination has been explored in a marginal colour way. The problematic is that today we can do color image recognition on numerical images but we don't have good results on nature images.

The CLEF contest is an answer to that problematic, the aim of the challenge is to put in competition some university laboratories and company laboratories. In this contest each laboratories could compare its colour texture feature against all the other challengers.

In this way, we will make a little presentation of what is a Key-points and how we used them. In a first time with the classical descriptors, SIFT, SURF, and opponent SIFT. We choose to present the last one because this is the descriptors used by FINKI, which is the laboratory of the last year contest we choose as reference for their methodology and to compare our results in a first place. In a second time, we will use a new descriptors purposed by Noel Richard. This descriptors is the C_2O, which we going to describe in this part with his strengths and weakness.

1.1 Key-points

1.1.1 Classical

1.1.2 Dense Grid

The dense grid method is an easy way to extract Key-points. First we have to divide the image in k subimages, where k is the number of subimages choose by the user. After you just have to take as Key-points the corner of each subimages.

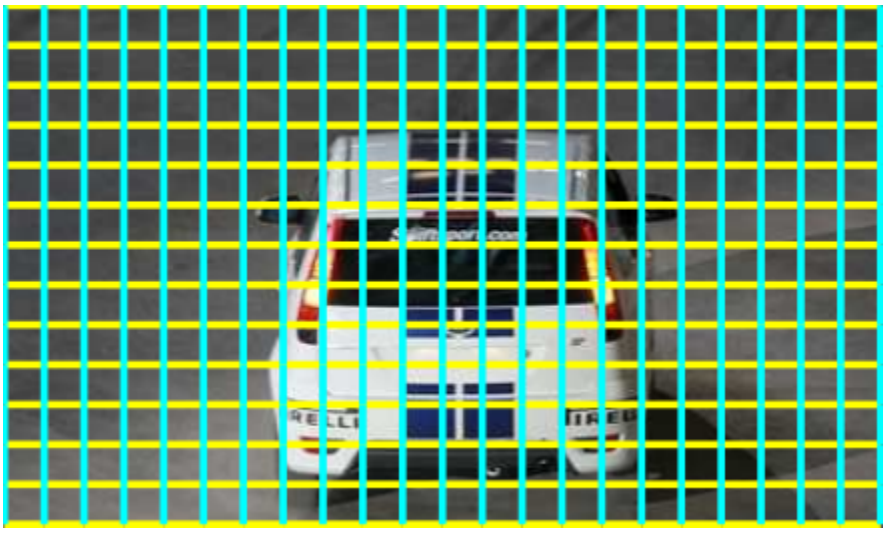

Figure 1: Dense grid

1.2 Descriptors

1.2.1 SIFT

Scale-invariant feature transform (or SIFT) is an algorithm in computer vision to detect and describe local features in images. The algorithm was published by David Lowe in 1999. Applications include object recognition, robotic mapping and navigation, image stitching, 3D modeling, gesture recognition, video tracking, individual identification of wildlife and match moving.

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

SIFT keypoints of objects are first extracted from a set of reference images and stored in a database. An object is recognized in a new image by individually comparing each feature from the new image to this database and finding candidate matching features based on Euclidean distance of their feature vectors. From the full set of matches, subsets of keypoints that agree on the object and its location, scale, and orientation in the new image are identified to filter out good matches. The determination of consistent clusters is performed rapidly by using an efficient hash table implementation of the generalized Hough transform. Each cluster of 3 or more features that agree on an object and its pose is then subject to further detailed model verification and subsequently outliers are discarded. Finally the probability that a particular set of features indicates the presence of an object is computed, given the accuracy of fit and number of probable false matches. Object matches that pass all these tests can be identified as correct with high confidence.

Algorithm:

- Scale-space extrema detection

  This is the stage where the interest points, which are called keypoints in the SIFT framework, are detected. For this, the image is convolved with Gaussian filters at different scales, and then the difference of successive Gaussian-blurred images are taken. Keypoints are then taken as maxima/minima of the Difference of Gaussians (DoG) that occur at multiple scales. Specifically, a DoG image D(x,y,σ) is given by:

  $$D(x, y, \sigma) = L(x, y, k_i\sigma) - L(x, y, k_j\sigma) \tag{1}$$

  where L(x,y,kσ) is the convolution of the original image I(x,y) with the Gaussian blur G(x,y,kσ) at scale kσ, i.e., L(x,y,kσ)= G(x,y,kσ)*I(x,y)

- Keypoint localization Scale-space extrema detection produces too many keypoint candidates, some of which are unstable. The next step in the algorithm is to perform a detailed fit to the nearby data for accurate location, scale, and ratio of principal curvatures. This information allows points to be rejected that have low contrast (and are therefore sensitive to noise) or are poorly localized along an edge.

Interpolation of nearby data for accurate position

First, for each candidate keypoint, interpolation of nearby data is used to accurately determine its position. The initial approach was to just locate each keypoint at the location and scale of the candidate keypoint. The new approach calculates the interpolated location of the extremum, which substantially improves matching and stability. The interpolation is done using the quadratic Taylor expansion of the Difference-of-Gaussian scale-space function, D(x,y,σ) with the candidate keypoint as the origin. This Taylor expansion is given by:

$$D(X) = D + \frac{\partial D^T}{\partial X}X + \frac{1}{2}X^T\frac{\partial^2 D}{\partial X^2} \tag{2}$$

where D and its derivatives are evaluated at the candidate keypoint and X=(x,y,σ) is the offset from this point. The location of the extremum, $\hat{X}$, is determined by taking the derivative of this function with respect to X and setting it to zero. If the offset $\hat{X}$ is larger than in any dimension, then that's an indication that the extremum lies closer to another candidate keypoint. In this case, the candidate keypoint is changed and the interpolation performed instead about that point. Otherwise the offset is added to its candidate keypoint to get the interpolated estimate for the location of the extremum. A similar subpixel determination of the locations of scale-space extrema is performed in the real-time implementation based on hybrid pyramids developed by Lindeberg and his co-workers.

Discarding low-contrast keypoints

To discard the keypoints with low contrast, the value of the second-order Taylor expansion D(X) is computed at the offset $\hat{X}$. If this value is less than 0.03, the candidate keypoint is discarded. Otherwise it is kept, with final scale-space location Y+$\hat{X}$, where Y is the original location of the keypoint.

Eliminating edge responses

The DoG function will have strong responses along edges, even if the candidate keypoint is not robust to small amounts of noise. Therefore, in order to increase stability, we need to eliminate the keypoints that have poorly determined locations but have high edge responses. For poorly defined peaks in the DoG function, the principal curvature across the edge would be much larger than the principal curvature along it. Finding these principal curvatures amounts to solving for the eigenvalues of the second-order Hessian matrix, H:

$$H = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{y_y} \end{bmatrix}$$

The eigenvalues of H are proportional to the principal curvatures of D. It turns out that the ratio of the two eigenvalues, say α is the larger one, and β the smaller one, with ratio r=$\frac{\alpha}{\beta}$, is sufficient for SIFT's purposes. The trace of H, i.e., D_{xx}+D_{y_y}, gives us the sum

of the two eigenvalues, while its determinant, i.e.,D_{xx} D_{y_y}-D_{xy}^2, yields the product. The ratio R=$\frac{Tr(H)^2}{det(H)}$ can be shown to be equal to $\frac{(r+1)^2}{r}$, which depends only on the ratio of the eigenvalues rather than their individual values. R is minimum when the eigenvalues are equal to each other. Therefore the higher the absolute difference between the two eigenvalues, which is equivalent to a higher absolute difference between the two principal curvatures of D, the higher the value of R. It follows that, for some threshold eigenvalue ratio r_{th}, if R for a candidate keypoint is larger than $\frac{(r_{th}+1)^2}{r_{th}}$, that keypoint is poorly localized and hence rejected. The new approach uses r_{th}=10. This processing step for suppressing responses at edges is a transfer of a corresponding approach in the Harris operator for corner detection. The difference is that the measure for thresholding is computed from the Hessian matrix instead of a second-moment matrix (see structure tensor).

- Orientation assignment

  In this step, each keypoint is assigned one or more orientations based on local image gradient directions. This is the key step in achievinginvariance to rotation as the keypoint descriptor can be represented relative to this orientation and therefore achieve invariance to image rotation. First, the Gaussian-smoothed image L(x,y,kσ) at the keypoint's scale σ is taken so that all computations are performed in a scale-invariant manner. For an image sample L(x,y) at scale σ, the gradient magnitude, m(x,y), and orientation, θ(x,y), are precomputed using pixel differences:

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2} \tag{3}$$

$$\theta(x,y) = atan2(L(x,y+1) - L(x,y-1), L(x+1,y) - L(x-1,y)) \tag{4}$$

  The magnitude and direction calculations for the gradient are done for every pixel in a neighboring region around the keypoint in the Gaussian-blurred image L. An orientation histogram with 36 bins is formed, with each bin covering 10 degrees. Each sample in the neighboring window added to a histogram bin is weighted by its gradient magnitude and by a Gaussian-weighted circular window with a σ that is 1.5 times that of the scale of the keypoint. The peaks in this histogram correspond to dominant orientations. Once the histogram is filled, the orientations corresponding to the highest peak and local peaks that are within 80% of the highest peaks are assigned to the keypoint. In the case of multiple orientations being assigned, an additional keypoint is created having the same location and scale as the original keypoint for each additional orientation.

- Keypoint descriptor

  Previous steps found keypoint locations at particular scales and assigned orientations to them. This ensured invariance to image location, scale and rotation. Now we want to compute a descriptor vector for each keypoint such that the descriptor is highly distinctive and partially invariant to the remaining variations such as illumination, 3D viewpoint, etc. This step is performed on the image closest in scale to the keypoint's scale. First a set of orientation histograms is created on 4x4 pixel neighborhoods with

8 bins each. These histograms are computed from magnitude and orientation values of samples in a 16 x 16 region around the keypoint such that each histogram contains samples from a 4 x 4 subregion of the original neighborhood region. The magnitudes are further weighted by a Gaussian function with equal to one half the width of the descriptor window. The descriptor then becomes a vector of all the values of these histograms. Since there are 4 x 4 = 16 histograms each with 8 bins the vector has 128 elements. This vector is then normalized to unit length in order to enhance invariance to affine changes in illumination. To reduce the effects of non-linear illumination a threshold of 0.2 is applied and the vector is again normalized. Although the dimension of the descriptor, i.e. 128, seems high, descriptors with lower dimension than this don't perform as well across the range of matching tasks and the computational cost remains low due to the approximate BBF (see below) method used for finding the nearest-neighbor. Longer descriptors continue to do better but not by much and there is an additional danger of increased sensitivity to distortion and occlusion. It is also shown that feature matching accuracy is above 50% for viewpoint changes of up to 50 degrees. Therefore SIFT descriptors are invariant to minor affine changes. To test the distinctiveness of the SIFT descriptors, matching accuracy is also measured against varying number of keypoints in the testing database, and it is shown that matching accuracy decreases only very slightly for very large database sizes, thus indicating that SIFT features are highly distinctive.

1.2.2 SURF

In computer vision, Speeded Up Robust Features (SURF) is a local feature detector that can be used for tasks such as object recognition or 3D reconstruction. It is partly inspired by the scale-invariant feature transform (SIFT) descriptor. The standard version of SURF is several times faster than SIFT and claimed by its authors to be more robust against different image transformations than SIFT.

SURF uses an integer approximation of the determinant of Hessian blob detector, which can be computed with 3 integer operations using an integral image. For features, it uses the sum of the Haar wavelet response around the point of interest. These can also be computed with the aid of the integral image.

SURF descriptors can be used to locate and recognize objects, people or faces, make 3D scenes, track objects and extract points of interest.

SURF was first presented by Herbert Bay et al. at the 2006 European Conference on Computer Vision. An application of the algorithm is patented in the US.

Algorithm and features:

- Detection

  The SURF algorithm is based on the same principles and steps as SIFT but details in each step are different. The SIFT approach uses cascaded filters to detect scale-invariant characteristic points, where the difference of Gaussians (DoG) is calculated on rescaled images progressively.

  Integral image

In SURF, square-shaped filters are used as an approximation of Gaussian smoothing. Filtering the image with a square is much faster if the integral image is used. The integral image is defined as:

$$S(x,y) = \sum_{i=0}^{x} \sum_{j=0}^{y} I(i,j) \tag{5}$$

The sum of the original image within a rectangle D can be evaluated quickly using the integral image. I(x, y) added over the selected area requires four evaluations S(x, y) (A, B, C, D).

Points of interest in the Hessian matrix

SURF uses a blob detector based on the Hessian matrix to find points of interest. The determinant of the Hessian matrix expresses the strength of the response and is an measure of local change around the point.

Blob structures are detected in places where the determinant of the Hessian is maximal. In contrast to the Hessian-Laplacian detector by Mikolajczyk and Schmid, SURF also uses the determinant of the Hessian for selecting the scale, as it is done by Lindeberg. Given a point p=(x, y) in an image I, the Hessian matrix H(p, σ) in p at scale σ, is defined as follows:

$$H(p,\sigma) = \begin{pmatrix} L_{xx}(p,\sigma) & L_{xy}(p,\sigma) \\ L_{xy}(p,\sigma) & L_{y_y}(p,\sigma) \end{pmatrix}$$

where $L_{xx}(p,\sigma)$ is the convolution of second order derivative $\frac{\partial x}{\partial x^2} g(\sigma)$ with the image in the point x, y similarly with $L_{xy}(p,\sigma)$ and $L_{y_y}(p,\sigma)$.

The Gaussian filters are optimal for scale space analysis, but in practice should be quantized and clipped. This leads to a loss of repeatability image rotations around the odd multiple of $\frac{\pi}{4}$. This weakness is true for Hessian-based detectors in general. Repeatability of peaks around multiples of $\frac{\pi}{2}$. This is due to the square shape of the filter. However, the detectors still work well, the discretization has a slight effect on performance. As real filters are not ideal, in any case, given the success of Lowe with logarithmic approximations, they push the approximation of the Hessian matrix further with square filters. These second order Gaussian filters' approximate can be evaluated quickly using integral images. Therefore, the calculation time is independent of the filter size. Here are some approaches: G_{y_y} and G_{xy} (1)

The box filters 9x9 are approximations of a Gaussian with $\sigma = 1.2$ and represents the lowest level (higher spatial resolution) for computerized maps BLOB response. Is denoted D_{xx}, D_{y_y}, D_{xy} . The weights applied to the rectangular regions are maintained by the efficiency of the CPU.

Images are calculated :

- D_{xx}(x, y) from I (x, y) and G_{xx} (x, y)
- D_{xy}(x, y) from I (x, y) and G_{xy} (x, y)

– D_{y_y}(x, y) from I (x, y) and G_{y_y} (x, y)

Then, the following image is generated:

$Det(Haprox) = D_{xx}D_{y_y} - (wD_{xy})^2$

1.2.3 Opponent SIFT

The opponent SIFT descriptors is an algorithm which use the same method than the classical SIFT descriptor. The only difference is that we calculate three descriptors for each Key-points. There are obtained from the color opponent channel, defined as

$$O_1 = \frac{R-G}{\sqrt{2}}, O_2 = \frac{R+G-2B}{\sqrt{6}}, O_3 = \frac{R+G+B}{\sqrt{3}}. \quad (6)$$

The opponent SIFT describes the three opponent color spaces, we can see that the two first channels O_1, O_2 contain some intensity information, but they are not invariant to changes of lights intensity. The last channel will contain the intensity information.

The strength of these method is that that use a color spaces, and we can see directly information of that with the algorithm of the SIFT descriptor. The weakness is that, in this approach we use the RGB space to make this computation.

1.2.4 C_2O