A survey on image description algorithms

Nuno Aniceto
nuno.j.aniceto@tecnico.ulisboa.pt

October 17, 2016

Abstract

This survey focuses on the field of computer vision. Several keypoint-detectors and descriptors are presented and a study is made to make an empirical evaluation of these. A simple key-point detector is also implemented on the OpenCV library, focused on the enabling a customized analysis to images by using several filters, thresholds and colour-analysis processes.

1 Introduction

One of the most important features offered by the computer vision field is the possibility of making a machine to recognize objects and scenarios simply by taking a photograph and matching it with memorized images. Some applications are for example: image registration, panorama stitching or visual search.

As inherent enabling systems, keypoint detection and feature extraction are used in order to ‘describe’ each image by a set of ‘key’ distinct features which can be compared in order to detect the likelihood of the images similarities. This is possible through several kinds of algorithms which have been in continuous development and will be briefly presented in the next sections.

Some important basic techniques and concepts are described followed by evaluation metrics. Datasets and software platforms are chosen in order to enable a proper comparison between the descriptors; these are also relevant for the purpose of future improvement over the quality of these detectors given several main issues these face.

1.1 Image Feature Detectors

There are three main classifications for the image feature detectors algorithms: edge-, blob- and point-of-interest-detectors which are introduced below.

a) Edge Detectors: In this class an edge is a set of points that has a high gradient value in the image, that is, regions in where the image changes abruptly by means of surrounding pixels intensity values. These kind of detectors are explored with more detail in the section 2.2.

b) Blob Detectors: Also known as ‘points-of-interest’, a blob is often referred to as a group of connected components which represent a contrast between their surroundings. Similar to edges, blobs are sharp points in the image. A motivation to study and develop techniques in order to detect these points is to make either tracking possible, or object detection - as these points also provide distinct features for that, possibly moving, object. There are several techniques which are used to detect blobs some of which are described below. These detectors are studied on the section 2.1.

c) Point-of-interest Descriptors: Since the earliest paper of Moravec [38], there are ever-evolving studies for development of corner detectors and more generally points of interest. Some of these algorithms are explored with more detail in the section 2.3.
2 Detectors Explored

2.1 Blob Detectors

A blob is a region where each perceived point is very similar to the surrounding points within the same region of the blob – that differ from the points outside, around the blob, by a measurable threshold value. Consider the following example, on an image with several stars or galaxies, a star or a galaxy can be considered a blob. Similar to edges, blobs are sharp points in the image. A motivation to study and develop techniques in order to detect these points is to make either tracking possible, or object detection - as these points also provide distinct features for that, possibly moving, object. There are several techniques which are used to detect blobs some of which are described below.

1) **Laplacian of Gaussian** [LoG] : The first linear scale-space detector, its kernel is given by the Gaussian equation [33, 30] (eq. 1).

\[
G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} \cdot e^{-\frac{(x^2+y^2)}{2\sigma^2}}
\]

Where a point in the image is represented as the coordinate \((x, y)\) and the \(\sigma\) parameter is a scale factor. The figure 1b exemplifies its application on an image.

2) **Difference of Gaussians** [DoG] : An approximation to the previous blob detector, it is faster but it is more ineffective [31, 7]. It is used on the SIFT detector.

\[
L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)
\]

\[
D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma)
\]

Where \(k\) represents a separation factor between the two different scale-spaces. The figure 1c exemplifies its application on an image.

3) **Determinant of Hessian** [DoH] : This blob detector is studied and used by Bay et al. [7] on the SURF corner detector. The figure 1d exemplifies its application on an image.

4) **Hessian-Laplace** : An hybrid strategy that makes use of both Hessian and the Gaussian approaches. It has been proposed by Mikolajczyk and Schmid [36].

There are other blob detectors such as the Maximally Stable Extremal Regions detector [MSER] which is popular but not focused on this paper [34].

![Figure 1: Several blob detectors applied to the same Hubble image](image-url)
The figure 1 presents an application of the aforementioned blob detectors on the original figure 1a. These have been set up with a maximum standard deviation of the largest blobs of \( \sigma = 30 \). The LoG detector has 10 intermediaries values for the standard deviation (\( \sigma \)). These detectors are implemented in the \texttt{scikit-image} python open-source library [51].

The original figure 1a is a photograph taken by the author on August 18th of 2015 at Braga’s Sanctuary on Bom Jesus do Monte.

### 2.2 Edge Detectors

The edge detectors have been in continuous development from the very early paper of Roberts [41] and are still being studied and improved. As an example we have the Scharr (2000) [45], Kroon (2009) [28].

The more widespread and known algorithms are noticeably the first detectors by Roberts (1963) [41] and Sobel (1968) [48, 13] along with the more robust detector of Canny (1986) [11] which is patented. Along with these Costella (2011) [12] proposes an unevaluated novel edge detector.

#### 2.2.1 Gradient Approximation by Convolution Matrix

Often that edge detectors rely on a square matrix with certain properties which is applied with the 2-dimensional convolution operator on the original image pixels to provide another image function that represents an approximation to a partial derivative of the original image in a certain orientation defined by the convolution matrix (please see equation 5). Notice that we make use of the discrete convolution operator given that the images do belong to a discrete domain (equation 4).

\[
(f \ast g)(n) \equiv \sum_{\tau=-\infty}^{\infty} f(\tau) \cdot g(n - \tau), \tau \in \mathbb{Z} \tag{4}
\]

In the equation 5, the \( G_{(x,y)} \) function represents the resulting image filtered pixels at a given position \((x, y)\) and the \( I_{(x,y)} \) function represents the original image pixel at the same given position. The convolution matrix is represented by \( M \) and can be defined by a squared \( n \times n \) shape where there is a need to represent an anchor point (middle). It is pretended that this matrix serves as a filter to define an approximation of the gradient of the image.

The main purpose of this calculation of the image gradient (shown in equation 5) is that of revealing the high frequency points of the image – eg: blobs and edges – which can be later filtered and used as distinct feature points for the image (please see the 2.2.2 section). For edge-detection purposes this matrix has certain properties: it is symmetric and the sum of its elements is zero: it is a zero-sum matrix. This matrix is usually obtained via the external product of a given kernel vector \( K \) whose absolute values are symmetric (as in equation 6) for \( 3 \times 3 \) matrices with a second vector depicted below in the same equation in which the latter is giving a positive weight value to the upper row and its symmetric negative value to the last row. Notice that it is also a zero-sum vector: its absolute values are symmetric, this is: \( v_1 = -v_3 \).

\[
M_x = K \times \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix} \quad \text{and} \quad M_y = -M_x^T = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix} \times K \tag{6}
\]

The symmetric transpose of this product is another matrix which represents an approximation to the images gradient in the other 2D orientation and thus the horizontal and the vertical filters for that same kernel are obtained. As explained above in equation 5 these matrices do represent a similarity with the partial derivatives of a function \( I_{(x,y)} \). The equation

---

7 shows an example for the horizontal derivative.

\[ G_x(x, y) = \frac{\delta}{\delta x} I_{(x, y)} \approx M_x \ast I_{(x, y)} \]  
\hspace{1cm} (7)

2.2.2 Sobel, Prewitt, Kayyali and Scharr Filters

The Sobel Filter is an "Isotropic 3x3 Image Gradient Operator" [48, 13]. The kernel vector for the Sobel filter is \((1 \ 2 \ 1)\). Other detectors such the Prewitt [40], the Kayyali [23] and the Scharr [45] do rely on the smoothing kernels: \((1 \ 1 \ 1)\); \((6 \ 0 \ -6)\) and \((3 \ 10 \ 3)\), respectively.

The Kayyali proposes two matrices (please see equation 8) by analysing diagonals, thus they are named SENW (south east - north west) and NESW (north east - south west).

\[ K_{senw} = \begin{bmatrix} +6 & 0 & -6 \\ 0 & 0 & 0 \\ -6 & 0 & +6 \end{bmatrix}; \quad K_{nesw} = \begin{bmatrix} -6 & 0 & +6 \\ 0 & 0 & 0 \\ +6 & 0 & -6 \end{bmatrix} \]  
\hspace{1cm} (8)

The Scharr detector is based on the Sobel approach and is considered as one edge-detector filter that provides an excellent approximation for the rotation invariance [4, 52, 28]. Its filters are shown below (please see equation 9).

\[ G_x = \begin{bmatrix} +3 & 0 & +3 \\ -10 & 0 & +10 \\ -3 & 0 & +3 \end{bmatrix} \]  
\hspace{1cm} (9)

2.2.3 Kroon Filters

Kroon [28] presents a filter that is computed for an optimal for rotation invariance shown below in equation 10. For solving this computation a quasi-Newton optimizer was used\(^3\). Kroon then shows an approximation to this kernel, given that it’s values are not integers but decimals instead, so that the resulting kernel: \((17 \ 61 \ 17)\).

\[ M^K = \begin{bmatrix} -17 & 0 & +17 \\ -61 & 0 & +61 \\ -17 & 0 & +17 \end{bmatrix} \]  
\hspace{1cm} (10)

2.2.4 Kirsch Filters

A very interesting filter is the Kirsch [26] detector which defines a contrast function by applying filters in several directions of the image, thus making it more sensitive\(^4\). This contrast function leads to the filters in equation 11.

\[ g_0 = \begin{bmatrix} +5 & +5 & +5 \\ -3 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix}; \quad g_1 = \begin{bmatrix} +5 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix}; \ldots \]  
\hspace{1cm} (11)

By application of each of these filters on the original image we have 8 image gradients (please see equation 12). The result of this method is the unique gradient \(G\) depicted in the equation 13.

\[ G_i(x, y) = g_i \ast I(x, y), \quad \forall i \in [0; 7] \]  
\hspace{1cm} (12)

\[ G(x, y) = \max \{G_i(x, y)\}, \quad \forall i \in [0; 7] \]  
\hspace{1cm} (13)

A summary of filters are described in the table 1. The images below depict the differences between an original cropped photograph and the magnitude of the filters described above. The cropped region is the same Costella [12] does in order to compare his results to these detectors results. Notice that the photograph has been publicly used and published by David Kennedy (CC BY-SA)\(^5\).

![Figure 2: The original reference image.](http://en.wikipedia.org/wiki/File:Bikesgray.jpg) – accessed at October 17, 2016


(a) The Sobel horizontal filter
(b) The Kirsch filter
(c) Result of Kayyali NESW filter
(d) Kroon kernel, 8 directions
(e) Scharr filter, 8 orientations
(f) The result of the Costella filter

Figure 3: Edge-detector filter results along with the application of the Kirsch method.

<table>
<thead>
<tr>
<th>Detector</th>
<th>kernel mask</th>
<th>filter matrices</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prewitt [40]</td>
<td>(1 1 1)</td>
<td>(1 0 -1) and (0 0 0)</td>
</tr>
<tr>
<td>Sobel and Feldman [48]</td>
<td>(1 2 1)</td>
<td>(-1 0 +1) and (0 0 0)</td>
</tr>
<tr>
<td>Scharr [45, 52]</td>
<td>(3 10 3)</td>
<td>(-3 0 +3) and (0 0 0)</td>
</tr>
<tr>
<td>Kayyali [23]</td>
<td>(+6 0 -6)</td>
<td>(+6 0 -6) and (0 0 0)</td>
</tr>
<tr>
<td>Kroon [28]</td>
<td>(17 61 17)</td>
<td>(-17 0 +17) and (0 0 0)</td>
</tr>
</tbody>
</table>

Table 1: Table describing the filter masks for several edge-detectors.
2.3 Image Descriptors

The work of Hans Moravec [38] marks the first step of recognition of interest points stating that regions with high differences in the several directions such as corners, or blobs, are better suited as heuristics instead of making use of simple edges or uniformly coloured regions of that image.

The image-descriptors generate feature vectors that represent each keypoint context in order to describe an image with more data related to each keypoint, thus improving precision in matching.

2.4 SIFT

Scale-Invariant Feature Transform (SIFT) is a big milestone due to the high number of features detected and their distinctiveness [9, 33].

SIFT relies on a cascade approach wherein each stage filters the potential key-points received from the previous step and selects or adds more information to the actual key-points before passing it to the next step.

2.5 SURF

Bay et al. [7] tests, proposes and applies a Fast-Hessian scheme which shows to be faster than the difference-of-Gaussian (DoG) approach (used in SIFT), the Harris-Laplace and the Hessian-Laplace blob detectors. This algorithm makes use of box filters that are applied directly to a given resolution of the image. This process eliminates the need of calculating the image pyramid, thus reducing the required time to compute the features. It also makes use of an approximation of block patterns to calculate the gradient orientation. As explained above, SIFT makes use of an histogram in order to detect the orientation of the gradient for a given keypoint.

2.6 KAZE and A-KAZE

The A-KAZE descriptor is an improvement over the previous KAZE algorithm in terms of performance and efficiency - thus it is coined as an Accelerated KAZE feature detector. It makes use of a scheme coined Fast Explicit Diffusion that is easier to implement, faster and more efficient than previous AOS schemes (used in KAZE) [4, 5]. It is also introduced a Modified Local-Difference Binary (M-LDB) descriptor that is both rotation- and scale-invariant which makes use of the gradient information from the scale-space and allows for further distinctiveness.

![Figure 4: Test to the scale invariance mechanisms used on the Lena Söderberg cropped image](image-url)
3 Evaluation and Conclusion

3.0.1 Evaluation

Results: The figure 4 shows a relationship between the diverse algorithms. As the scale factor is increased, the SIFT and KAZE algorithms show to demand more time than AKAZE and SURF. The BRISK and ORB algorithms are the fastest, appearing to detect and compute keypoints and descriptors on a constant-time.

3.0.2 Conclusions

Conclusions For the analysis to the descriptors, the results for detection show better scores for the AKAZE algorithm on the tests to blur robustness, rotation- and scale-invariance, with MSER showing good scores as well. The method proposed by Alcantarilla et al. [4] shows a surprising behaviour. By making a locally-adaptive smoothing, reduces noise and maintains the main features of the image.

In our keypoint detector the Kirsch method has been developed and the Gaussian pyramid is used for scale-space invariance along with colour conversions and analysis but it’s early stage cannot compete yet with these algorithms.

References


Acknowledgments

References


