



UNIVERSITÀ DEGLI STUDI DI PALERMO
Dipartimento di Ingegneria Chimica, Gestionale, Informatica, Meccanica
Dottorato di Ricerca in Ingegneria Informatica

TEXTURE AND LOCAL KEYPOINTS ANALYSIS FOR ADVANCED IMAGE INSPECTION

SSD: ING-INF/05

TESI DI

ING. ALESSANDRO BRUNO

TUTOR

CH.SSIMO PROF. EDOARDO ARDIZZONE

COORDINATORE DEL DOTTORATO

CH.SSIMO PROF. SALVATORE GAGLIO

CICLO XXIII

DOTTORATO



ACKNOWLEDGEMENTS

I would like to thank my tutor Prof. Edoardo Ardizzone, for his help, support and for supervising my research work during these three years of PhD.

I wish also to thank Dr. Giuseppe Mazzola, who provided me the guidelines and supported me to achieve the goals of my research.

Index

Abstract.....	7
1. TEXTURE ANALYSIS.....	9
1.1. Introduction.....	9
1.2. Texture and Visual Perception	14
1.3. Texture Classification.....	16
1.4. Texture Analysis Approaches:	18
1.4.1 Statistical Approaches	18
1.4.2 Spectral Methods	19
1.4.3 Structural Methods	19
1.5. Traslation, rotation, scale invariance.....	21
1.6. Texture Descriptors.....	24
1.6.1 Fractals	24
1.6.2 Edge Histogram Descriptor	25
1.6.3 Homogeneous Texture Descriptor	27
1.6.4 Perceptual Browsing Descriptor PBD.....	30
1.7. <i>Statistical moments of Histogram</i>	32
1.7.1 Haralick Descriptor.....	34
1.7.2 Spatial filter based descriptor.....	36
1.7.3 Tamura Descriptors.....	36

1.7.4	Gabor Filter	38
1.7.4.1	Gabor Based Texture Descriptor.....	39
1.7.5	Zernike Moments	40
1.8.	Application Domains	43
2.	LOCAL KEYPOINTS	46
2.1.	Introduction.....	46
2.2.	State of the art.....	51
2.2.1	Local Keypoint Detection.....	51
2.3.	Harris Edge Corner.....	53
2.4.	SIFT (Scale Invariant Feature Transform)	57
2.4.1	Scale-space extrema detection	57
2.4.2	Local Extrema Detection	59
2.4.3	Keypoint Localization	59
2.4.4	Orientation Assignment	61
2.4.5	Local Image Descriptor.....	61
2.5.	PCA-SIFT detector	62
2.6.	SURF (Speeded Up Robust Feature)	63
2.6.1	Scale space representation	65
2.6.2	Interest Point Localization.....	67
2.6.3	Interest Point Description.....	69
2.6.4	Orientation	69
2.6.5	Haar Wavelet Responses	70

2.7. Local Keypoint Comparisons.....	72
3. TEXTURE SCALE DETECTION	78
3.1. Introduction.....	78
3.2. State of the art.....	81
3.3. Keypoint Density Map.....	83
3.4. Texture Scale Detection.....	85
3.5. Texel Extraction	88
3.6. Experimental Results	90
3.7. Conclusions	94
4. VISUAL SALIENCY	97
4.1. Introduction.....	97
4.2. State of the art.....	98
4.3. Proposed Saliency Measure	100
4.4. SIFT Density Maps.....	101
4.5. Saliency Map	105
4.6. Experimental results	108
4.7. Conclusions	111
5. IMAGE FORENSICS.....	116
5.1. Introduction.....	116
5.2. Image Tampering.....	117
5.3. Copy-Move forgeries	118
5.3.1 State of the art.....	120

5.4. Block matching approach	121
5.4.1 Texture Descriptors	123
5.4.2 Evaluation Metrics	124
5.4.3 Experimental Results	127
5.4.4 Conclusions	136
5.5. Keypoint cluster matching approach	137
5.5.1 Features Extraction	137
5.5.2 Keypoint clustering	138
5.5.3 Custers matching	141
5.5.4 Post Processing	142
5.5.5 Experimental Results	145
5.5.5.1 Controlled geometrical transformations	146
5.5.5.2 Not tampered images	147
5.5.5.3 Tampered images	147
5.5.5.4 Temporal Efficiency	148
5.5.6 Conclusions	149
5.6. Triangles matching approach	152
5.6.1 Delaunay Triangulation	152
5.6.2 Triangles Matching	153
5.6.3 Experimental Results	156
5.6.4 Conclusions	165
CONCLUSIONS	166

Abstract

Today, in digital era, digital imaging is rapidly growing. It is going to become an essential medium to retrieve, display and manipulate information. Digital imaging include Computer Vision, Image Processing, Computer Graphics and Visualization.

A picture or an image can be described from different perspectives. High-level description refers to "top-level" aspect, overall systemic features, it is more abstracted and is typically more concerned with the system as a whole, and its goals. High-level description of the image often corresponds with a semantic description of the image content. Face, object, text, are examples of high level features of the image.

Low-level analysis describes individual components, detail rather than overview, rudimentary functions rather than complex overall ones, and it is typically more concerned with individual components within the system and how they operate. Color, texture, edge are examples of low level features of the image.

An important approach to image region description is to quantify its texture content. Properties such as smoothness, regularity and coarseness are measured by this descriptor. Texture indicates visual pattern in real scene (woods surface, wall, waves, ecc.) and synthetic scene. Local Keypoints and the associated local features recently attracted attention for their capability of characterizing salient regions of the image. The neighborhood of local keypoints is rich of *contents*, for this reason these are usually detected in texture regions.

This PhD thesis presents advanced image inspection techniques based on texture and local keypoint analysis. The arguments treated in this thesis are organized in a growing level of abstraction.

The first two chapters give an overview of texture and local keypoints analysis and description. The third chapter presents the argument of texture scale

detection and it includes a new technique for texture scale detection. The fourth chapter shows a new technique for visual saliency based on local keypoints distribution. The fifth chapter deals with image forensics argument and it presents three techniques for a specific image forensics task.

1. TEXTURE ANALYSIS

1.1. Introduction

Texture is a difficult concept to represent, it gives a look measure for visual patterns in images. The identification of specific textures in an image is achieved primarily by modeling texture as a two-dimensional gray level variation. The relative brightness of pairs of pixels is computed such that degree of contrast, regularity, coarseness and directionality may be estimated [1]. However, the problem is in identifying patterns of co-pixel variation and associating them with particular classes of textures such as silky, or rough.

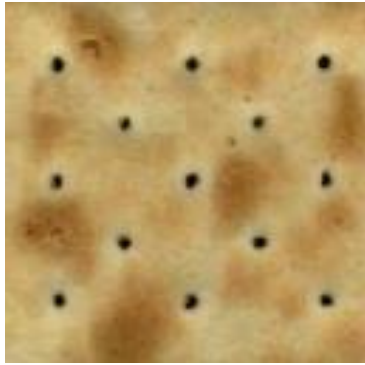
In image processing and computer vision literature many methods and algorithms deal with image features extraction, some of these are based on objects boundaries, some are based on color informations, some methods are based on texture features analysis. The content of an image could be considered as a group of spatial objects spatial entities, such as surfaces, edges, lines and points, with spatial relationships (adjacency, orientation, ecc.). It is very important to analyze the picture regions for the image content description.

An important approach to image region description is to quantify its texture content. Texture analysis methods in the state of the art give no formal definition of texture. Properties such as smoothness, regularity and coarseness are measured by this descriptor (texture). Visual texture is easy to recognize but it is very difficult to define. In computer vision and image processing literature a lot of texture definitions , Coggins [2] compiled a catalogue of this definitions. Some examples are given from [3]:

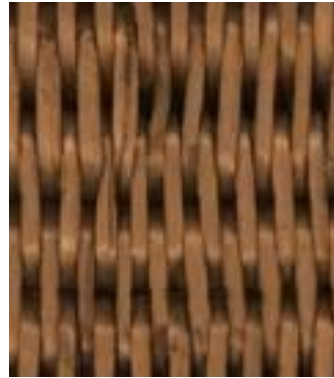
- “We may regard texture as what constitutes a macroscopic region. Its structure is simply attributed to the repetitive patterns in which elements or primitives are arranged according to a placement rule.” [4]
- “A region in an image has a constant texture if a set of a local statistics or other local properties of the picture function are constant, slowly varying, or approximately periodic.” [5]
- “The image texture we consider is nonfigurative and cellular... An image texture is describe by the number and types of its (tonal) primitives and the spatial organization or layout of its (tonal) primitive. A fundamental characteristic of texture: it cannot be analyzed without a frame of reference of tonal primitive being state or implied. For any smooth gray-tone surface, there exists a scale such that when the surface is examined, it has no texture. Then as resolution increases, it takes on a fine texture then a coarse texture.” [6]
- “Texture is defined for our purposes a san attribute of a field having no components that appear enumerable. The phase relations between the components are thus not apparent. Nor should the field contain an obvious gradient. The intent of this definition is to direct attention of the observer to the global properties of the display – i.d., its overall “coarseness”, “bumpiness”, or “fitness”. Phisically, nonenumerable (aperiodic) patterns are generated by stochastic as opposed to deterministic processes. Perceptually, however, the set of all patterns without obvious enumerabl components will include many deterministic (and even periodic) textures.” [7]

- “Texture is an apparently paradoxical notion. On the one hand, it is commonly used in the early processing of visual information, especially for practical classification purposes. On the other hand, no one has succeeded in producing a commonly accepted definition of texture. The resolution of this, we feel, will depend on a richer, more developed model for early visual information processing, a central aspect of which will be representational systems at many different levels of abstraction. These levels will most probably include actual intensities at the bottom and will progress through edge and orientation descriptors to surface, and perhaps volumetric descriptors. Given these multi-level structures, it seems clear that they should be included in the definition of, and in the computation of, texture descriptors.” [8]
- “The notion of texture appears to depend upon three ingredients: (i) some local ‘order’ is repeated over a region which is large in comparison to the order’s size, (ii) the order consists in the nonrandom arrangement of elementary parts, and (iii) the parts are roughly uniform entities having approximately the same dimensions everywhere within the textured region.” [9]

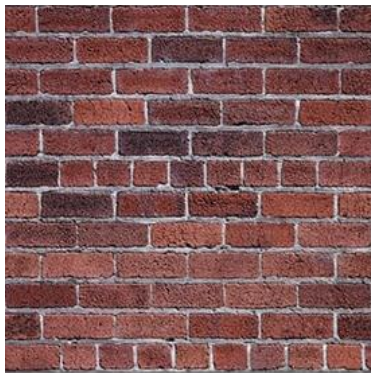
As seen above, Scientific Literature provides many texture definitions, it depends upon the particular kind of applications and upon the motivation. Texture analysis is an important and useful area of study in computer vision and image processing, most natural surface in real scene exhibit texture. A lot of real images consist of many regions, each of which is made up of different texture tokens, the figure-ground separation, for example, is a very interesting issue for computer vision community. Texture can be also defined as a function of the spatial variation in pixel intensities (gray values). Based on textural properties, a variety of materials weave can be detected, a few of examples is reported in fig.1,2.



a)



b)



c)

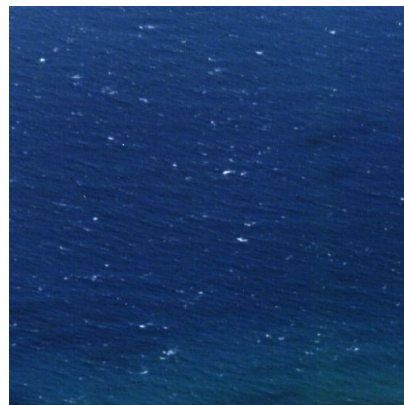


d)

Fig. 1 Four examples of texture patterns on real surfaces. a) biscuit b) wood surface; c) bricks; d) windows



a)



b)

Fig. 2 Two examples of texture patterns on real surfaces. a) prison window b) waves;

Images of real objects often do not exhibit uniform intensities regions. Visual texture is such as the repeated pattern present in wooden surface (Fig.1b).Texture features are also employed for image rapresentation and widely used for comparing images.

In this chapter an overview of texture analysis and description approaches is given.

1.2. Texture and Visual Perception

Most natural surfaces exhibit texture and a successful vision system must be able to deal with textured world surrounding it. The psychophysical aspect of texture perception is very important: the detection of a tiger among the foliage is a perceptual task that carries life and death consequences for someone trying to survive in the forest. In psychophysical studies of texture perception the performance of various texture algorithms is evaluated against the performance of the human visual system doing the same task. The image in fig consists of two images, each of them is made up of different texture tokens. The immediate perception of the image does not result in the perception of two different textured regions; instead only one uniformly textured region is perceived.



Fig. 3 A pair of non-preattentively discriminable textures. The figure is from [1]. Texture pairs have identical second order statistics.

Julesz has studied texture perception extensively in the context of texture discrimination [10] [11] [12]. Julesz dealt with texture discrimination problem, more particularly he focused his attention on the discrimination of texture pairs showing similar brightness, contrast, color. Julesz particularly studied first and second order spatial statistics. First order can be computed from the histogram of pixel intensities in the image. This depends only on individual pixel values and not on the interaction or co-occurrence of neighboring pixel values. Second order

statistic is defined as the likelihood of observing a pair of gray values occurring at the endpoints of a dipole of random length placed in the image at a random location and orientation.

Julesz presupposed that two textures are not preattentively discriminable if their second-order statistics are identical, in Fig.3 a pair of textured regions whose second-order statistics are identical.

Julesz, in "theory of textons" [13] states that that textons are "the putative units of pre-attentive human texture perception", related to texture's local features, such as edges, line ends, blobs, etc. Julesz observed that human texture discrimination could be modelled by first-order density of such textons. Researches show that human visual system performs local spatial frequency analysis on retinal images which could be simulated by a computational model using a filter bank [14] [15]. This theory has motivated mathematic models of filter-based human texture perception. Bergen [16] suggests that a texture can be decomposed into a series of sub-band images using a bank of linear filters at different scales and orientations. Each sub-band image is related to certain texture features. Therefore, a texture is characterised by an empirical distribution of the magnitude of filter responses and therefore similarity metrics, e.g., distance between distributions, could be derived for discriminating textures.

As a whole, visual texture perception is an important subject area in vision science. A variety of theories have been developed to understand the mechanisms about human texture perception. Texture research in neuroscience and psychophysics has greatly influenced the counterpart in computer vision. The Julesz Theory, inspired the statistical approach to texture analysis which characterises a texture by image statistical features. The texton theory led to a structural approach that extracts texture primitives as local features for texture description. The filter-bank model has also been introduced into computational texture modelling, resulting in methods that decompose a texture using filters and extend texture analysis into the frequency domain.

1.3. Texture Classification

Texture indicates visual patterns in real and synthetic scenes. In image processing there are different kinds of textures, from stochastic to regular (regular, near-regular, irregular, near stochastic, stochastic). Due to complexity and diversity of natural textures, textures analysis, texture description and texture synthesis are very challenging tasks. In image processing literature there are several approaches for texture inspection, including statistical approaches of autocorrelation function, optical transforms, digital transforms, gray tone co-occurrence, run lengths, and autoregressive models [6]. Structural texture analysis focuses primarily on identifying periodicity in texture or identifying the placement of texels (in this paper we use the words "texel" and "texton" as synonyms). Texton has not a precise mathematical definition, although in 1981 Julesz [13] defined it as the "putative unit of pre-attentive human texture perception" (qualitative definition). In our method we consider a texel as "basic repetitive element of texture pattern". In terms of periodicity and regularity of the structure of texture elements there are 5 kind of textures: regular, near regular, irregular, near stochastic and stochastic.

- Regular textures are simply periodic patterns where the intensity of color and the shape of all texture elements are repeated in equal intervals;
- Near Regular textures are a statistical distortion of a regular pattern;
- Irregular textures present deformation fields from regular patterns;
- Near Stochastic and Stochastic textures show typically dots and shapes randomly scattered over all the image.

Many scenes in real world do not have a regular structure but a near regular one (buildings, wallpapers, floors, tiles, windows, fabric, pottery and decorative arts, animal fur, gait patterns, feathers, leaves, waves of the ocean, and patterns of sand dunes), so for natural scenes the problem is to detect the scale, and to extract the texel, also in case of a near regular periodicity. Texture scale value can be used in many image processing applications such as image segmentation,

texture description, Content Based Image Retrieval and texture synthesis. Texture scale detection is not a young research field in image processing, but today it remains one of more challenging issue.

The following properties play an important role in describing texture: uniformity, density, coarseness, roughness, regularity, linearity, directionality, direction, frequency, phase, scale. Some of these properties are not independent.

1.4. Texture Analysis Approaches:

An important approach to region description is to quantify its texture content. Texture descriptor provides measures of properties such as smoothness, coarseness, regularity, linearity, directionality, direction, frequency, phase, scale. The principal approaches [17] used in image processing to describe texture are: Statistical, Structural, Spectral and Model Based

1.4.1 Statistical Approaches

The statistical approaches extract, what so called, texture feature descriptors based on region histograms their extensions and their moments. These descriptors are used to measure contrast, granularity and coarseness of the image. Texture feature descriptors can be classified into two categories according to the order of the statistical function that is utilized: first-order texture features and second-order texture features [18].

First-order texture features, also known as grey level distribution moments (GLDM), are extracted exclusively from the information provided by the intensity histograms, thus it yields no information about the locations of the pixels. The second-order texture features take into account the specific position of a pixel relative to another. The most popularly used of second-order methods is the grey level co-occurrence matrix (GLCM) method, which depends on constructing matrices by counting the number of occurrences of pixel pairs of given intensities at a given displacement.

The intensity-level histogram of an image is a function showing the number of pixels in the whole image. It gives a measure of the statistical informations contained in the image. It further contains the first-order statistical information about the image. Most often so-called central moments are derived from it to characterise the texture (see section 1.6 Statistical moments of Histogram).

1.4.2 Spectral Methods

Spectral techniques are based on the autocorrelation function of a region or on the power distribution in the Fourier transform domain. Spectral techniques detect texture periodicities. The Fourier transform analyzes the global frequency in the signal (image), a textured image can be decomposed into its frequency and orientation components. Gabor and Wavelet models are based on Fourier transform, in both of them multi-resolution processing is applied.

In eq. 1 $F(u,v)$ is 2-D Fourier Transform of a two-dimensional signal $f(x,y)$.

$$F(u, v) = \int \int_{-\infty}^{\infty} f(x, y) e^{-j2\pi(ux+vy)} dx dy \quad (1)$$

Fourier transform based methods have good performances on textures showing strong periodicity, on the other hand their performance deteriorates on textures showing no periodicity [19]. The spectral approach is not very popular among researchers dealing with texture analysis due to the performance problems and the high computational complexities of the Fourier Transform [20].

1.4.3 Structural Methods

Structural techniques describe the texture by using pattern primitives accompanied by certain placement rules. In structural approach, the texture region is defined to have a constant texture if a set of local statistics or other local properties of the image are constant, slowly varying or approximately periodic. An image texture is described by the number and types of its (tonal) primitives and the spatial organization or layout of the primitives [21]. Texture images are analyzed by identifying the local and global properties of the images under consideration. The basic idea is that a simple "texture primitive" can be used to form more complex texture patterns by means of some rules that limit the number of possible arrangements of the primitive. Extraction of texture elements and inference of the placement rule are two major steps for structural texture

analysis approaches. In structural approach Number of primitives and spatial relationship are not independent.

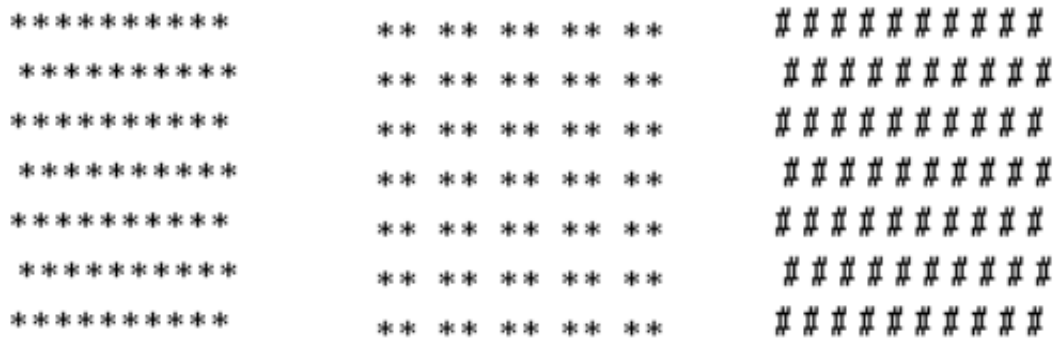


Fig.4 Textures with the same number and same type of primitives does not necessarily give the same texture (the first and the second pattern). The same spatial relationship does not guarantee same texture (the first and the last pattern).

Commonly used element properties are average element intensity, area, perimeter, eccentricity, orientation, elongation, magnitude, compactnes, euler number, moments, etc. [22]. Perimeter and compactness of a primitive are used to characterize the texture by Goyal et al. [23]. Perimeter is in theory invariant to shift and rotation. But in practice, it varies due to quantization error. Histogram is a very useful method for texture analysis, a texture can be characterized by the histogram of its texture elements. Goyal et al. [24] have noted that the texel property may not change under translation and rotation. They use the histogram property to get invariant texture features and propose a method to convert the traditional histogram to an invariant histogram.

Eichmann and Kasparis [25] discuss the extraction of invariant texture descriptors from line structures based on Hough transform (HT). Line segments in the image correspond to the points in the HT plane. Image rotation is equivalent to the translation in the HT plane along the angle axis. Scaling in the texture is equivalent to the translation in the HT plane along the ρ -axis. At last, Lam et al. [26] use iterative morphological decomposition (IMD) method for scale invariant texture classification. IMD decomposes the image into scale-dependent set of component images. For each component texture statistics features are extracted.

1.5. Traslation, rotation, scale invariance

The majority of texture analysis methods make the explicit or implicit assumption that texture images are acquired from the same viewpoint, the same scale and orientation. It is not easy to ensure that images captured have the same translation, rotations or scaling. Texture analysis should be ideally invariant to viewpoints.

Coordinate system play a very important role in invariant texture analysis. Translations, rotations, and scaling of texture image are caused by linear transform of the image points coordinates. The analysis of transform properties is needed.

The geometric transforms of texture images can be described as the following mathematical model:

$$\begin{bmatrix} x_a \\ y_a \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} d_1 \\ d_2 \end{bmatrix} \quad (2)$$

$$f_a(x_a, y_a) = f(x, y) \quad (3)$$

The function $f(x, y)$ is the original image, $f_a(x_a, y_a)$ is the transformed image. The matrix **D** is the translation factor, and the matrix **C** is the rotation, scaling and skew factor. If C is orthogonal matrix, this implies that only translation and rotation occur in eq. (prima equazione della tabella). One way to handle rotation and scaling is to adopt a suitable transform such that rotation and scaling will appear as a simple displacement in the new coordinates system. The polar coordinates system or log-polar grid is a good way to solve this problem. Rotation in cartesian grid (x,y) corresponds to circular translation in the polar grid (fig.4).

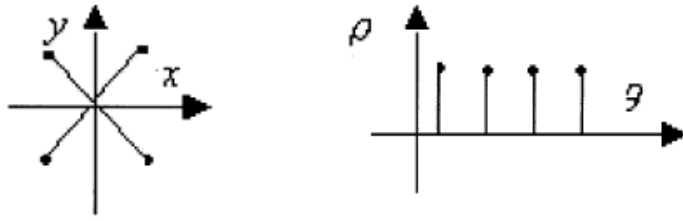


Fig.4 Rotation in cartesian grid (x,y) corresponds to circular translation in the polar grid

Scaling in the (x; y) grid corresponds to a shift along the log axis in the log-polar grid. The drawback of applying these methods for invariant analysis is that the translation invariance is sacrificed. Some shift-invariant methods must be applied in advance (for example the Fourier transform). In the following three paragraphs Statistical, Model Based, and Structural methods are listed in function of rotation, translation, scale invariance.

From the existing literature on invariant texture analysis it is observed that: some statistics texture features such as global mean, variance and central moments, are invariant to translation and rotation. Fourier Spectrum may be used to deal with translation invariance. Polar coordinates can be employed on the frequency domain to obtain rotation invariant features (Gabor Model, Harmonic Expansion). Some texel properties, such as area, perimeter and compactness of the texels, are invariant to translation and rotation. Histogram properties can be used to obtain scale invariance.

Many authors developed statistical methods for invariant texture analysis. Davis [27] describes a tool (polarogram) for image texture analysis and used it to get invariant texture features. A polarogram is a polar plot of a texture statistics as a function of orientation, the features computed from the polarogram are invariant to the orientation of the polarogram. Since the rotation of the original texture results in the rotation of the polarogram, the features are invariant to the rotation of the original texture. Mayorga and Ludeman [28] used polar grid for rotation invariant texture analysis. The features are extracted on the texture edge statistics obtained through directional derivatives among circularly layered data.

Two sets of invariant features are used for texture classification. The first set is obtained computing the circularly averaged differences in the gray level between a pixel. The second computes the correlation function along circular levels. In Harmonic Expansion and Mellin transform [29] [30] [31] an image is decomposed into a combination of harmonic components in its polar form. The projection of the original image to the harmonic operators gives the features of the pattern. The magnitudes of these coefficients are used for rotation invariant pattern analysis. In [32] Duvernoy et al. use the technique of Fourier descriptors to extract the invariant texture signature on the spectrum domain of the original texture. The image is characterized by the optical Fourier spectrum of the texture. A contour line is defined by selecting the points whose energy equals to a given percentage of the central maximum of this spectrum. The energy of the coefficients of the descriptors is invariant to the rotation and change of the scale of the curve.

Tsatsanis and Giannakis [33] employed the high-order statistics to solve the invariant texture classification and modeling problems (They use cumulant and multiple correlation).

Another example of invariant texture descriptor is represented by Zernike moments. The orthogonality of Zernike moments provides an important property: the magnitude of the moments are invariant to rotation [34]. Pietikainen et al. [35] present some features based on center-symmetric auto-correlation, local binary pattern and gray level difference to describe texture images. Most of these features are locally invariant to rotation including linear symmetric auto-correlation measures (SAC), rank order version (SRAC), related covariance measures (SCOV), rotation invariant local binary pattern (LBPROT) features and gray-level difference (DIFF4).

1.6. Texture Descriptors

In this section of the chapter, most used texture descriptors are reported: Fractals, Edge Histogram Descriptor, Homogeneous Texture Descriptor, Perceptual Browsing Descriptor, Statistical Moments of Histogram, Haralick, Tamura, based descriptor, Zernike Moments, Gray level co-occurrence matrix, ecc...

1.6.1 Fractals

Many natural surfaces show roughness and self-similarity at different scale. Fractal surfaces are produced by a number of basic physical processes, ranging from the aggregation of galaxies to the curdling of cheese. The defining characteristic of a fractal is that it has a fractional dimension, from which the word "fractal" is taken [36]. The fractal dimension, particularly, gives a measure of the roughness of a surface. More simply, larger fractal dimension corresponds to rougher texture. **D** indicates the **Fractal Dimension**. Given a bounded set A in Euclidean n-space, the set A is said to be self-similar when A is the union of N distinct (non-overlapping) copies of itself; each of which has been scaled down by a ratio of r.

$$D = \frac{\log N}{\log (1/r)} \quad (4)$$

Pentland in [36] gave evidence that most natural surfaces can be modeled by fractals. Other methods for fractals estimation are in [37] [38]. The limit of this approach is that textured surfaces are not deterministic as described, they have a statistical variation. Statistical approach, such as gray level co-occurrence matrix, outperformed fractals, especially in segmentation and classification texture tasks.

1.6.2 Edge Histogram Descriptor

The Edge Histogram Descriptor is a very simple descriptor that gives a measure of edge density of the image. The normative edge histogram for MPEG-7 [39] is designed to contain only local edge distribution with 80 bins. To localize edge distribution to a certain area of the image, the image space is subdivided into 4x4 sub-images as shown in Fig. 5 for each sub-image, an edge histogram is generated to represent edge distribution. To define different edge types, the sub-image is furthermore divided into small square blocks called image-blocks.

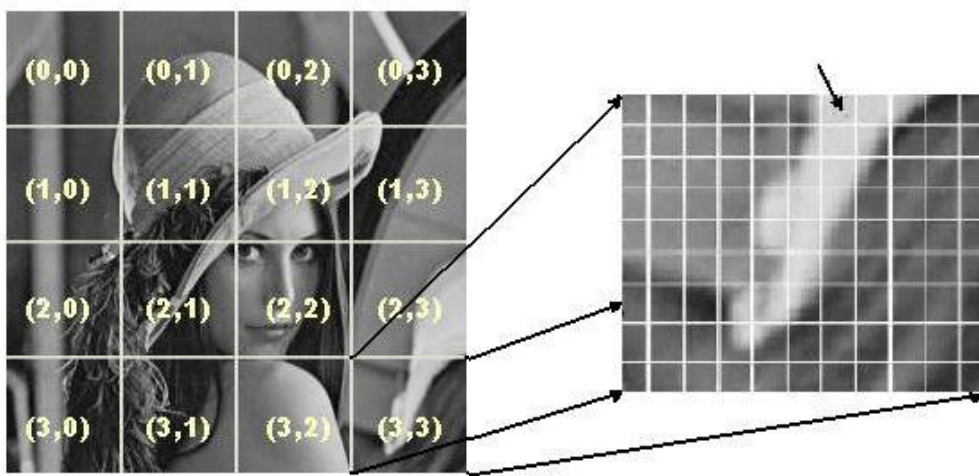


Fig. 5 Subdivision of the image in sub-blocks

Five edge types are defined in the descriptor: four directional edge and a non directional.

- *vertical*,
- *horizontal*,
- *45 degree*,
- *135 degree*
- *non directional* (the edges have not a precise direction belong to this category)

The edges are extracted from image blocks, then total number of edges are counted for each directional edge type. The image is subdivided in 16 sub-images, for each of images 5 bins edge histograms are computed, total number of bins in edge histogram is $16 \times 5 = 80$. The order of block scanning is shown in the Fig.6

Histogram bins	Semantics
Local_Edge [0]	Vertical edge of sub-image at (0,0)
Local_Edge [1]	Horizontal edge of sub-image at (0,0)
Local_Edge [2]	45degree edge of sub-image at (0,0)
Local_Edge [3]	135 degree edge of sub-image at (0,0)
Local_Edge [4]	Non-directional edge of sub-image at (0,0)
Local_Edge [5]	Vertical edge of sub-image at (0,1)
:	:
:	:
:	:
Local_Edge [74]	Non-directional edge of sub-image at (3,2)
Local_Edge [75]	Vertical edge of sub-image at (3,3)
Local_Edge [76]	Horizontal edge of sub-image at (3,3)
Local_Edge [77]	45degree edge of sub-image at (3,3)
Local_Edge [78]	135 degree edge of sub-image at (3,3)
Local_Edge [79]	Non-directional edge of sub-image at (3,3)

Fig. 6 Histogram Bins and Local Edge.

In edge extraction, each of subimages is furthermore subdivided in non-overlapping blocks. Block size depends on spatial resolution of the image. The luminance mean values for the four sub-blocks are used for the edge detection. More precisely, mean values for each of sub-block are convolved with the filters in fig. , to obtain edge magnitude. From edge magnitude values, edge strength is obtained, then if this value is greater than a threshold the block is classified with the corresponding edge type, else the block is classified as belonging at non-directional edge.

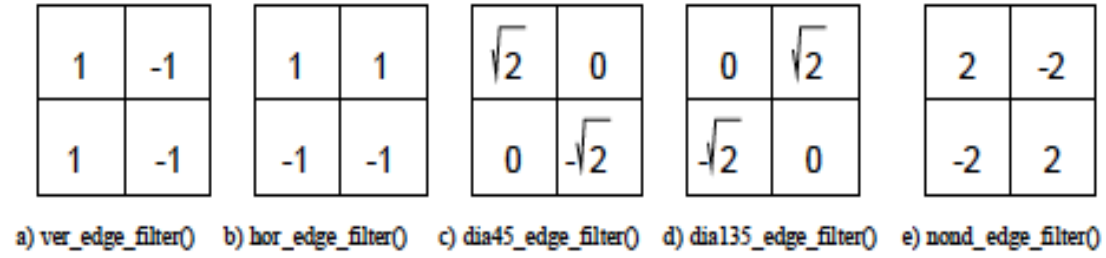


Fig. 7 Edge Filters

Edge Histogram Descriptor is scale invariant, it has small size for an efficient storage of metadata. The limit of EDH for texture description is that it does not give informations about relative spatial positions of pixel.

1.6.3 Homogeneous Texture Descriptor

The current MPEG-7 homogeneous texture descriptor (HTD) is composed of 62 numbers[40]. It consists of the mean, the standard deviation value of an image, energy, and energy deviation values of the Gabor Filter of the image. These are extracted from partitioned frequency channels based on the human visual system (HVS). For reliable extraction of the texture descriptor, Radon transformation is employed.

From psychophysics experiments, the response of the visual cortex is turned to a band limited portion of the frequency domain. In homogeneous texture descriptor the frequency layout allows extracted texture information to be matched with human perception system. The frequency layout consists of sub-bands. In these bands, the texture descriptor components such as energy and energy deviation are extracted.

The subbands are designed by dividing the frequency domain, then texture feature values are computed. The frequency space is partitioned in equal angles of 30 degrees along the angular direction and in octave division along the radial direction. The sub-bands in the frequency domain are called feature channels indicated as C_i in Fig. 8.

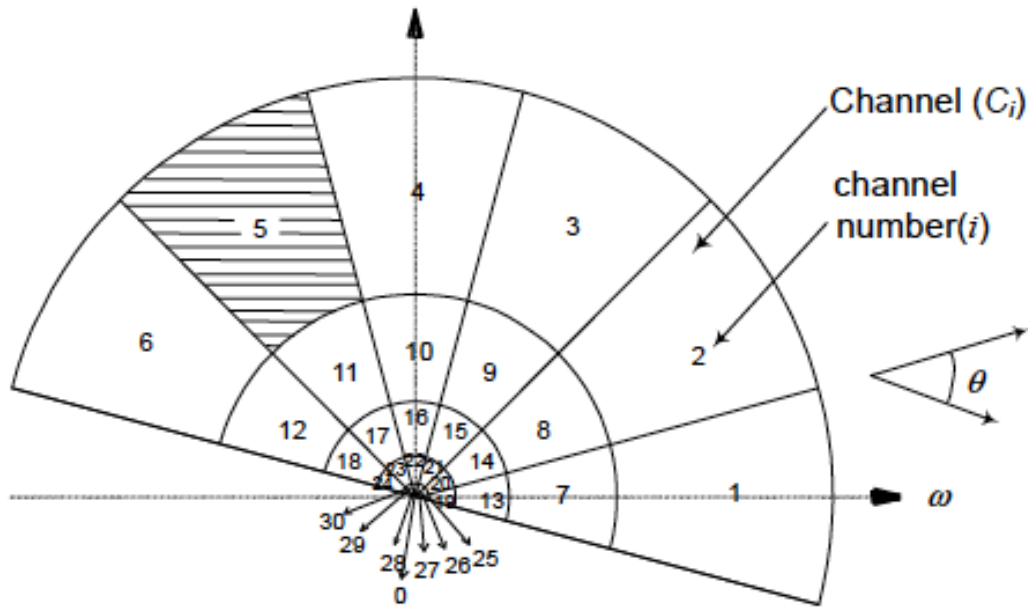


Fig.8 Frequency space partition

The frequency space is partitioned in 30 feature channels. The center frequencies of the feature channels are spaced equally in 30 degrees along the angular direction such as $\theta_r = 30^\circ \times r$, $r = \{0,1,2,3,4,5\}$. Each partitioned region corresponds to a band-limited portion of the frequency domain that is the response of the visual cortex in the HVS. The channels located in the low frequency areas are of smaller sizes while those of the high frequency areas are of larger sizes. Human vision is more sensitive to the change of low frequency area. The center frequencies of the channels in the angular and radial directions are such that:

$$\theta_r = 30^\circ \times r \quad 0 \leq r \leq 5, \quad \omega_s = \omega_0 \cdot 2^{-s} \quad 0 \leq s \leq 4, \quad \omega_0 = 3/4.$$

The equation (5) is the Gabor Wavelet filters.

$$G_{P_{s,r}}(\omega, \theta) = \exp\left[\frac{-(\omega - \omega_s)^2}{2\sigma_{\omega_s}^2}\right] \cdot \exp\left[\frac{-(\theta - \theta_r)^2}{2\sigma_{\theta_r}^2}\right] \quad (5)$$

σ_{ω_r} is the standard deviation in radial direction

$\sigma_{\theta r}$ is the standard deviation in angular direction. ω_r

In HTD, Radon transform is followed by 1-d Fourier transform application. Let $F(\omega, \theta)$ be the result, the energy e_i and the energy deviation d_i of the i^{th} channel is σ_p :

$$e_i = \log[1 + p_i], d_i = \log[1 + q_i] \quad (6)$$

$$p_i = \sum_{\omega=0+}^1 \sum_{\theta=0+}^{360} I_{sr}^2, \quad (7)$$

$$q_i = \sqrt{\sum_{\omega=0+}^1 \sum_{\theta=0+}^{360} \{I_{sr}^2 - p_i\}^2} \quad (8)$$

$$I_{sr} = G_{P_{s,r}}(\omega, \theta) \cdot |\omega| \cdot F(\omega, \theta) \quad (9)$$

$|\omega|$ is the Jacobian between the Cartesian and the Polar.

Homogeneous Texture Descriptor in vector form:

$$HTD = [f_{DC}, f_{SD}, e_1, e_2, \dots, e_{30}, d_1, d_2, \dots, d_{30}]$$

f_{DC} and f_{SD} are **mean** and **standard deviation** of the image.

1.6.4 Perceptual Browsing Descriptor PBD

In perceptual browsing descriptor Regularity, Direction and Coarseness are described. **PBD** has the following syntax: **PBD=[v1,v2,v3,v4,v5]**. v1 represent regularity. v2 and v3 represent the directionality. v4 and v5 are two scales that best represent the coarseness.

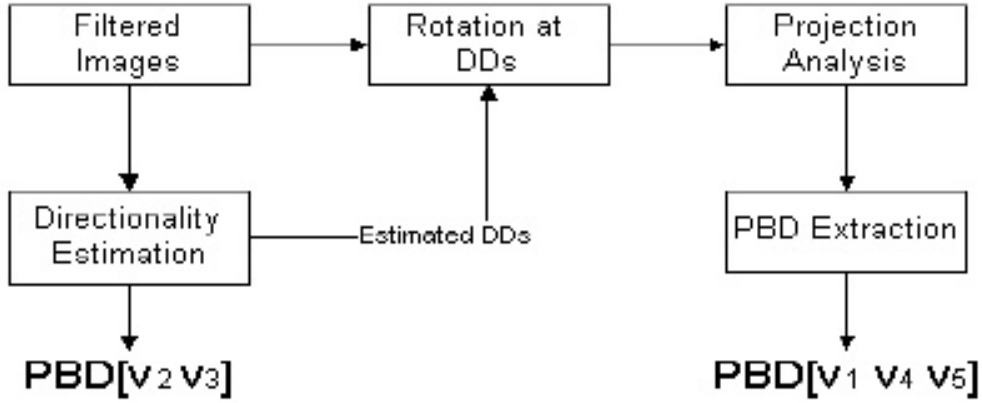


Fig.9 PBD algorithm steps

The image is decomposed into a set of filtered images, $W_{mn}(x,y)$. For $S \times K$ images S directional histograms are used for **Dominant Direction** estimation.

$$H(s, k) = \frac{N(s, k)}{\sum_1^K N(s, k)}, \quad s = 1, \dots, S \text{ and } k = 1, \dots, K \quad (0.1) \quad (10)$$

PDB[v2] and PDB[v3] represent the directions having peaks in $H(s,k)$, in neighboring scale. Furthermore, these direction have two peaks with highest sharpness.

Given a peak, the formola (11) and (12) represent sharpness:

$$P_H^{(m)}(l) = \iint W_m(x, y) \delta(x \cos \theta_{DOI} + y \sin \theta_{DOI} - l) dx dy \quad (11)$$

$$P_v^{(mn)}(l) = \iint W_m(x, y) \delta(x \cos \theta_{d02} + y \sin \theta_{d02} - l) dx dy \quad (12)$$

θ_{d01} and θ_{d02} are the dominant directions. $P(l)$ is the projection corresponding to θ_{d0} .

$p_posi(i)$, $p_magn(i)$ and $v_posi(j)$, $v_magn(j)$ are the positions and magnitudes of the peaks and valley of **NAC(k)**, **Normalized Autocorrelation** (see eq. 13). Given **dis** and **std** representing **mean** and **standard deviation** of successive peaks distances. Projections with std/dis less than a threshold are candidates.

$m^*(H)$ and $m^*(V)$ are the scale index of candidate $P_H^{(mn)}$ and $P_V^{(mn)}$ with maximum contrast. $PBD[v4]=m^*(H)$ and $PBD[v5]=m^*(V)$.

$$NAC(k) = \frac{\sum_{m=k}^{N-1} P(m-k)P(m)}{\sqrt{\sum_{m=k}^{N-1} P^2(m-k) \sum_{m=k}^{N-1} P^2(m)}} \quad (13)$$

$$contrast = \frac{1}{M} \sum_{i=1}^M p_magn(i) - \frac{1}{N} \sum_{j=1}^N v_magn(i) \quad (14)$$

Regularity (v1) is obtained by assigning credits to candidates, for more details see [41].

1.7. Statistical moments of Histogram

The Histogram is one of the most commonly used characteristic to represent the global feature composition of an image. It is invariant to translation and rotation of the images and normalizing the histogram leads to scale invariance. Exploiting the above properties, the histogram is considered to be very useful for indexing and retrieving images.

The intensity-level histogram of an image is a function showing (for each intensity level) the number of pixels in the whole image. This is a concise and simple summary of the statistical information contained in the image. Calculation of the gray level histogram involves single pixels, it further contains the first-order statistical information about the image. Most often so-called central moments are derived from histogram to characterise the texture, as defined by the four following equations:

$$\text{Mean} = \bar{x} = \frac{1}{N} \sum_{i=1}^N x_i \quad (15)$$

$$\text{Variance} = \sigma = \sqrt{\frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})^2} \quad (16)$$

Variance can be considered as a contrast measure, higher values indicate higher gray levels dispersion inside of the image. This implies that analyzed texture region has random distribution if variance has high value.

$$\text{Skewness} = \frac{1}{N-1} \frac{\sum_{i=1}^N (x_i - \bar{x})^3}{\sigma^3} \quad (17)$$

Skewness coincides with third statistical order of histogram. It is zero if the histogram is symmetrical about the mean, it is otherwise either positive or

negative depending whether it has been skewed above or below the mean. Skewness gives a measure of symmetry.

$$\text{Kurtosis} = \frac{1}{N-1} \frac{\sum_{i=1}^N (x_i - \bar{x})^4}{\sigma^4} \quad (18)$$

Kurtosis coincides with the fourth statistical order of histogram, it gives a measure of the flatness of histogram curve with the respect of gaussian distribution.

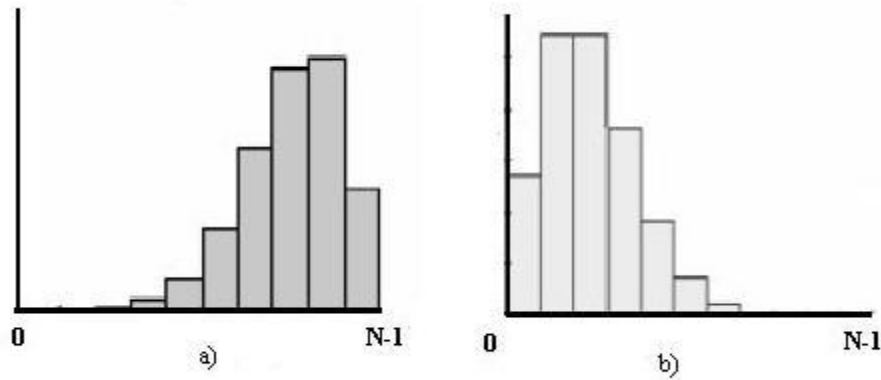


Fig.10 Example of right asymmetric (a) and left asymmetric histogram (b)

Statistical Moments of intensity level histogram are used for texture analysis in the following vector form (v):

$$v = [\text{mean}, \text{variance}, \text{skewness}, \text{kurtosis}]$$

1.7.1 Haralick Descriptor

Haralick Descriptor is co-occurrence matrix based. A **co-occurrence matrix**, F is used to describe the patterns of neighboring pixels in an image at a given distance d . In the features extraction phase 4 matrices are needed to describe different orientations. More precisely, four matrices include Vertical, horizontal and diagonal in both directions: P^0 , P^{90} , P^{45} and P^{135} . The co-occurrence matrices are symmetric matrices with $M \times M$ dimensions, M is the number of possible gray levels of the image. These matrices can be represented by a tridimensional data, third dimension varies depending on dimensions of the original input image.

In fig.11 the construction of co-occurrence matrix for $d=1$, in fig .12 a 3-dimensional representation of co-occurrence matrix for a given orientation.

Four descriptors are computed after the construction of co-occurrence matrices: Contrast, Energy, Homogeneity, Correlation. Haralick descriptor is totally composed of 16 components vector as depicted in fig.13.

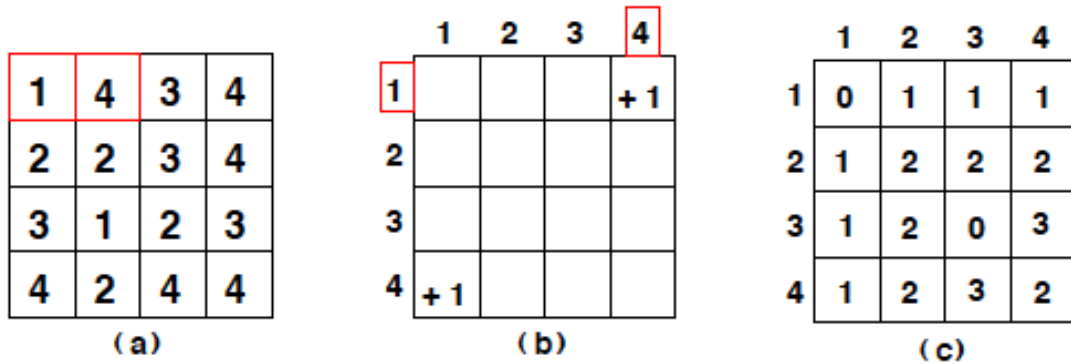


Fig.11 Co-occurrence matrices

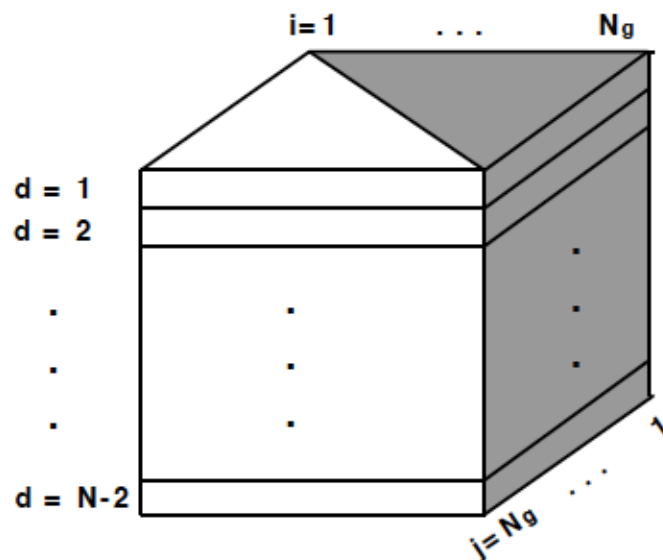


Fig.12 A tridimensional representation of co-occurrence matrix for a given orientation

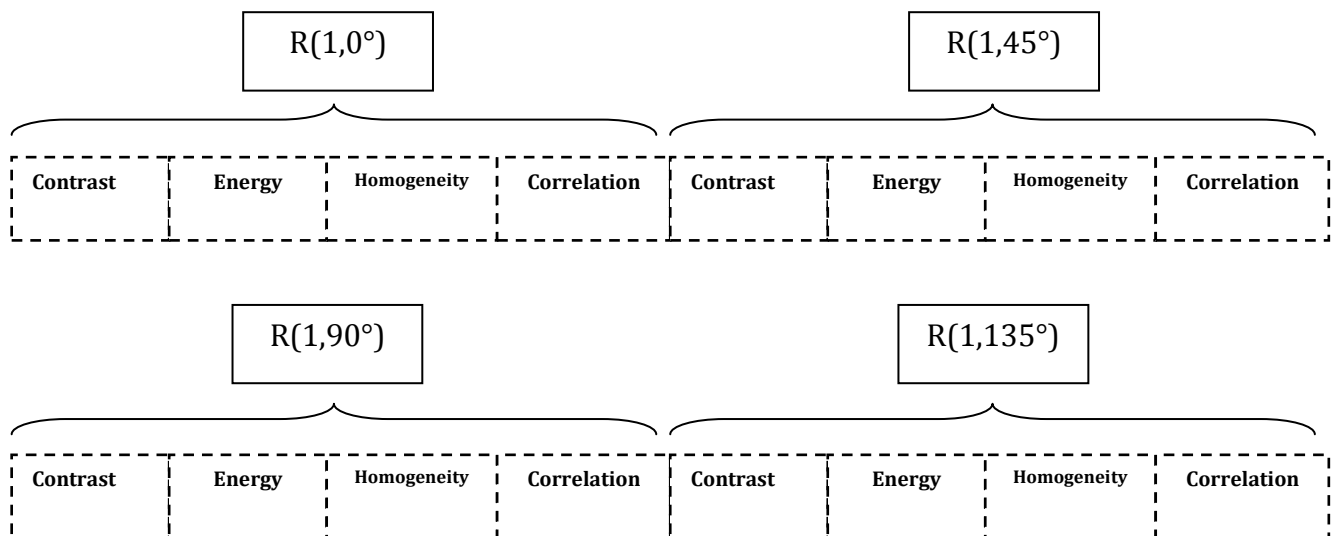


Fig.13 Haralick descriptor components.

For a more detailed description see [42].

1.7.2 Spatial filter based descriptor

This texture descriptor analyzes the mean local distribution of the edges inside of an image block, it takes into account five directional components, in order:

- **vertical,**
- **horizontal,**
- **diagonal - 45°,**
- **diagonal - 135°,**
- **not directional.**

Last category include the edges have not a dominant direction in the image.

Every block is filtered by the following spatial filter mask. Every mask is corresponding to a specific direction as in fig.7.

The image is then subdivided in blocks with the same size of the filter masks. Each of block of the image is filtered with the five spatial filter masks (directional). It is returned the sum of pixel (for each of directional filter) of filtered sub-blocks of the image. Then each of block of the image is represented by five component vector; each of component represents the five categories of directional edge.

1.7.3 Tamura Descriptors

Textural features corresponding to human visual perception are very useful for optimum feature selection and texture analyzer design. Tamura et al defined six textural features: Coarseness, Contrast, Directionality, line-likeness, regularity and roughness. Tamura et al [4] found the first three features to be very important. The three features, **coarseness**, **contrast**, and **directionality**, are defined as follows:

- **Coarseness** has direct relationship to scale and repetition rates, it aims to identify the largest size at which a texture exists, even where a smaller

micro texture exist. It takes averages at every point over neighbourhoods the linear size of which are powers of 2. For a neighbourhood of size $2^k * 2^k$ at the point (x,y) the average is:

$$A_k(x, y) = \sum_{i=x-2^{k-1}}^{x+2^{k-1}-1} \sum_{j=y-2^{k-1}}^{y+2^{k-1}-1} f(i, j) / 2^{2k} \quad (19)$$

At each point one takes differences between pairs of averages corresponding to non overlapping neighbourhood on opposite sides of the point in both horizontal and vertical orientations. The coarseness measure is then the average of $S_{opt}(x,y) = 2^{k_{opt}}$

- **Contrast** feature aims to capture the dynamic range of gray levels in an image, together with the polarization of black and white distribution. Dynamic range of gray levels is measured by standard deviation of gray levels. The polarization of black and white distribution is measured by standard deviation and kurtosis of gray levels of the image (α_4). Given kurtosis, mean and the variance, contrast measure is:

$$F_{con} = \sigma / (\alpha_4)^n \quad \text{where} \quad \alpha_4 = \mu_4 / \sigma^4 \quad (20)$$

μ_4 is kurtosis, μ is the mean σ^2 is the variance. With $n=4$ Tamura gives the closest agreement to human measurements.

- **Directionality**. This is a global property, it gives a measure of the total degree of directionality. Two simple masks are used to detect edges in

the image. At each pixel the angle and magnitude are calculated. Edge probabilities histogram is built up by counting all points with magnitude greater than a threshold and quantising by the edge angle. The histogram will reflect the degree of directionality. To extract a measure from **Hd** the sharpness of the peaks are computed from their second moments.

Linelikeness, Regularity and Roughness features are not considered important because derived from Contrast, Coarseness and Directionality. These are not very distinctive features.

1.7.4 Gabor Filter

Gabor filter is one of the most popular signal processing based approaches for texture feature extraction. It enables filtering in the frequency and spatial domains. A bank of filters at different scales and orientations allows multichannel filtering of an image to extract frequency and orientation information. The feature is computed by filtering the image with a bank of orientation and scale sensitive filters, the mean and standard deviation are computed in frequency domain.

Given an image **I(x,y)**, a gabor filter **g** designed according to [43] results in Gabor Wavelet transform:

$$W_{mn}(x, y) = \int I(x, y) g_{mn}^*(x - x_1, y - y_1) dx_1 dy_1 \quad (21)$$

The mean and standard deviation of the magnitude **|W_{mn}|** are used to for the feature vector. The outputs of filters at different scales will be over differing ranges. For this reason each element of the feature vector is normalised using the standard deviation of that element across the entire database

1.7.4.1 Gabor Based Texture Descriptor

Multi-Channel filtering approach [44] is inspired by a multi-channel filtering theory. In the early stages perception, human visual system decomposes the retinal image into a number of filtered images, each of which contains intensity variations over a narrow range of frequency (size) and orientation. This approach to texture analysis is intuitively appealing because the dominant spatial-frequency components of different textures are different. Simple statistics of gray values in the filtered images are used as texture features.

A bank of Gabor filters is used to characterize the channels. More particularly, the filter set forms an approximate basis for a wavelet transform, with the Gabor function as the wavelet. The channels are represented with a bank of two-dimensional Gabor filters. A two-dimensional Gabor function consists of a sinusoidal plane wave of some frequency and orientation, modulated by two-dimensional Gaussian envelopes.

In the following formula, a canonical version of Gabor filter in spatial domain:

$$h(x, y) = \exp \left\{ -\frac{1}{2} \left[\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2} \right] \right\} \cos(2\pi u_0 x + \phi), \quad (22)$$

μ_0 and ϕ represent the frequency phase of the sinusoidal plane wave along the x-axis (0° orientation) and $\sigma_x \sigma_y$ are the space constants of the Gaussian envelope along the x and y-axis. The frequency and orientation selective properties of a Gabor filter are more explicit in its frequency domain representation. The Fourier transform of the Gabor function is:

$$H(u, v) = A \left(\exp \left\{ -\frac{1}{2} \left[\frac{(u-u_0)^2}{\sigma_u^2} + \frac{v^2}{\sigma_v^2} \right] \right\} + \exp \left\{ -\frac{1}{2} \left[\frac{(u+u_0)^2}{\sigma_u^2} + \frac{v^2}{\sigma_v^2} \right] \right\} \right) \quad (23)$$

where:

$$\sigma_u = 1/2\pi\sigma_x, \sigma_v = 1/2\pi\sigma_y, \text{ and } A = 2\pi\sigma_x\sigma_y \quad (24)$$

The Fourier domain representation in eq. 23 specifies the amount by which the filter modifies or modulates each frequency components of the input image.

Each Gabor filter is a discrete realization of eq. 23.

Four orientation θ values are used: $0^\circ, 45^\circ, 90^\circ, 135^\circ$. The restriction to four orientations is made for computational efficiency. Given an image with a width N_c pixels (N_c is power of 2), the following radial values are used for frequency μ_0 :

$$1\sqrt{2}, 2\sqrt{2}, 4\sqrt{2}, \dots, (N_c/4)\sqrt{2}.$$

The orientation and frequency bandwidths of each filter are: 45° and 1 octave respectively.

If the size of Gabor filter masks is 16×16 , $N_c = 5$ and number of orientation is equal to 4, feature vector will have 40 coefficients:

$$f = (\mu_{00} \sigma_{00}, \mu_{01} \sigma_{01}, \dots, \mu_{40} \sigma_{40}) \quad (25)$$

In eq. 25 μ_{00} and σ_{00} are mean and variance values respectively of block of filtered image. The vector coefficients are corresponding to fixed orientation and frequency values.

1.7.5 Zernike Moments

Moment of the image is a powerful statistic tool for pattern recognition, it is defined as the projection of a function $f(\mathbf{x}; \mathbf{y})$ onto a monomial $\mathbf{x}^p \mathbf{y}^q$:

$$M_{pq} = \int \int x^p y^q f(x, y) dx dy. \quad (26)$$

Zernike moments are the projection of the function $f(\mathbf{x}; \mathbf{y})$ onto Zernike polynomials.

$$z_{mm} = [(n+1)/\pi] \int \int_U f(x, y) V_{nm}^*(x, y) dx dy, \quad (27)$$

Zernike polynomials $V_{nm}(\mathbf{x}; \mathbf{y})$ are one of an infinite set of polynomials that are orthogonal over the unity circle $\mathbf{U}: \mathbf{x}^2 + \mathbf{y}^2 \leq 1$

$$\int \int_U V_{pq}(x, y) V_{nm}^*(x, y) dx dy = \frac{\pi}{n+1} \delta_{pn} \delta_{pm} \quad (28)$$

$$\delta_{ij} = \begin{cases} 1, & i=j, \\ 0, & \text{otherwise.} \end{cases}$$

$$V_{nm}(x, y) = V_{nm}(\rho, \theta) = R_{nm}(\rho) \exp(jm\theta) \quad (29)$$

ρ e θ represent polar coordinates inside of the unity circle, R_{nm} are ρ polynomials.

Zernike moments are orthogonal moments. Their kernel is Zernike polynomials defined over the polar coordinate space inside of unit circle.

Wang and Healey [45] used Zernike Moments for texture analysis because these descriptors perform well in practice to obtain geometric invariance. In texture analysis, texture is characterized using Zernike moments of multispectral correlation functions. The correlation function is invariant to translation for its nature, the scale invariance is obtained by normalization of correlation functions. Zernike Moments are extracted from each correlation to form a set of translation and rotation invariants.

1.8. Application Domains

Texture analysis methods have been utilized in a variety of applications domains. Image inspection, Medical Image Analysis, Document Processing, Remote Sensing, Image Segmentation, Visual Saliency, Image Forensics. The role that texture plays in these examples varies depending upon the application. Texture processing for inspection problems involve, for example, defect detection in images of textiles, automated inspection of carpet wear and automobile paints. In medical image analysis, texture properties play an important role. Sutton and Hall [46] discuss the classification of pulmonary disease using texture features, Landeweerd and Gelsema [47] extracted first-and second order statistics to differentiate types of white cell bloods. Texture segmentation methods for preprocessing document images to identify region of interests [48,49] (i.e. postal address recognition, analysis and interpretation of maps). Similar methods are used for locating text blocks in newspapers (see fig.14).



Fig.14 - Texture Segmentation for locating bar code in newspaper image

Texture analysis has been furthermore used to classify remotely sensed images. Haralick et al. [50] used gray level co-occurrence features to analyze remotely sensed images; Land use classification where homogeneous regions with different types of terrains need to be identified is an important application. In visual saliency texture properties are used to detect the common and rare visual aspects that are involved in visual perception. Image Forensics deals with image forgeries detection, many methods used texture features to detect the tampering inside of the image [51] (see fig. 15). Some of the aspects inherent in texture analysis applications, will be further discussed in next chapters.

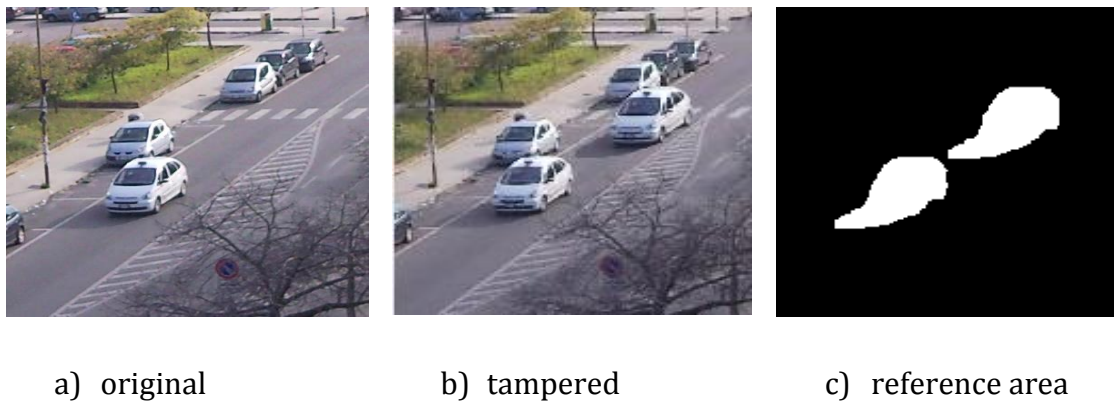


Fig. 15 Copy move detection by block matching with texture features.

2. LOCAL KEYPOINTS

2.1. Introduction

Many computer vision and image processing tasks rely on feature extraction. Local keypoint, or interest point are such features. These are widely used in Image retrieval, image forensics, visual saliency, image registration, camera calibration, object recognition. In scientific literature there is not a unique definition about local keypoint, in this section some definitions are reported and local keypoint properties are discussed. From now on, "interest point" and 'local keypoint' will indicate the same meaning.

- "By Keypoints, we mean typically blobs, corners, and junctions. These features have been also referred to as interest or salient points in the literature. In human vision, these localised features, along with edges, are perceived as privileged cues for recognising shapes and objects and are widely used in computer vision for various applications including object recognition, stereo matching, content based image retrieval, mosaicing, motion tracking." [52]
- "A line segment is considered as having some thickness. That means any line segment in a form image consists of a group of black pixels which has a rectangular shape. Generally the height of a horizontal line segment or the

width of a vertical line segment is very small. In defining a keypoint, we ignore two short edges of the line segment so a line segment can be considered as a group of black pixels associated with two long edge lines beside its two sides as shown in fig 16. The end of a long edge line is referred to as an end keypoint. When several horizontal and vertical line segments touch or overlap together the intersection points of two edges lines form another kind of keypoint, a corner keypoint." [53]

- "Interest points are locations in the image where the signal changes two-dimensionally. Example include corner, T-junctions, as well as locations where the texture varies significantly." [54] (T-junction can be an occlusion or the result of reflectance discontinuity).
- Each keypoint is defined by its center \mathbf{K} , its scale σ and its orientation ω and is denoted by a circle on the image with radius $\rho = 6\sigma$. [55]

The first definition is very generic, it only emphasizes the importance of local keypoints for computer vision and image processing tasks. The second definition is focused on geometric viewpoint, infact it is based on geometric elementary concepts. In third definition keypoint concept is based on signal intensity, some visual examples are reported in fig.16.

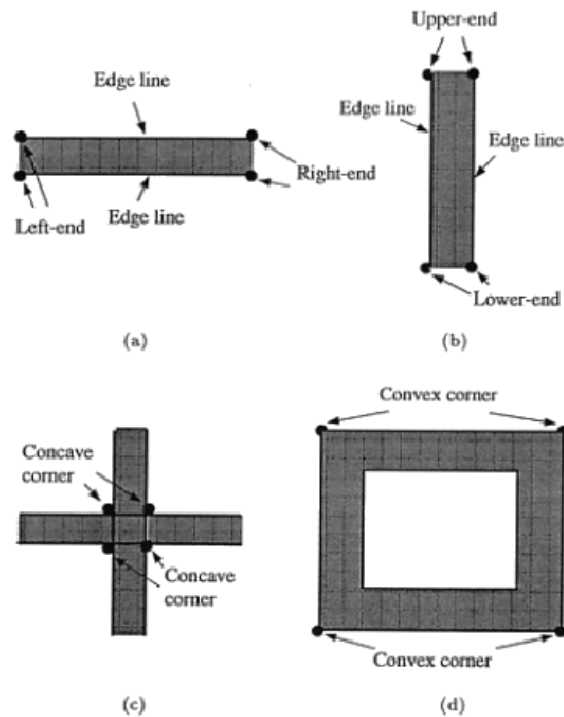


Fig.16- Local keypoint - Geometric viewpoint

Mathematical definition, spatial position, stability and repeatability characterize local keypoint as it follows:

- Keypoint *position* in image space is mathematically defined,
- The neighborhood of interest point is rich of *contents*, (keypoints are usually detected in texture regions, in regions with reflectance discontinuity).
- it is *stable* under local and global perturbations in the image. Perturbations include perspective transformations, affine transformations, scale changes, rotations and/or translations, illumination/brightness variations.
- Most of interest point are detected by multiscale analysis

Local Keypoint detection is correlated with blob and corner detection. In early works about corner features, the goal was obtaining robust, stable and defined image features for object detection and recognition. Usually corner detectors are sensitive not only to corners, but to image regions which show high grade of variation in all directions. In [56] “A corner is defined as a location where a triangle with specified opening angle and size can be inscribed in the curve”. Rangarajan et al. [57] defined a corner as the junction point of two or more straight line edges.

Blob features are clusters of similar pixels in the image plane and can be arise from similarity of color, texture, motion and other signal based metrics [58]. Blob features are used for region detection, in [59] Lindeberg blob is defined by a maximum of the normalized Laplacian in scale-space.

The interest points are correlated with regions of interest which have been used for object detection. Regions of interest are often formulated as the result of blob detection.

For the most common types of blob detectors each blob descriptor has defined point, which may correspond to a local maximum in the operator response.

Interest point and corner have a well-defined position and can be robustly detected. An interest point can be a corner but it can also be an isolated point of local intensity maximum or minimum, line endings, or a point on a curve where the curvature is locally maximal. Most corner detection methods detect interest points in general, rather than corners in particular. For this reason if only corners are needed to be detected, corner detection results must be filtered by a local analysis to determine which of these are real corners.

A criterion for local keypoint descriptor is the repeatability rate. Repeatability means that the points are obtained independently of changes in the imaging conditions. Every local keypoint is described by a feature vector that represents its neighborhood. A good interest point descriptor should be robust against noise, displacements, geometric and photometric deformations. Small dimensions of

descriptor are desirable for a fast interest point extraction and description because the dimension of the descriptor has impact on the time it takes. On the other hand, higher dimensional feature vectors are, in general, more distinctive than their lower-dimensional counterparts.

Another criterion for interest point descriptor evaluation is the information content, it gives a measure of the distinctiveness of the local graylevel pattern at an interest point. The most important interest point detectors have been also developed for a rich neighborhood description.

This chapter is focused on scale changes, translation and rotation invariant detectors and descriptors. In the next paragraphs a review of works in interest point detection and description is given.

2.2. State of the art

Many methods for image retrieval adopt global or local features, this choice has long been an issue of research in image and video retrieval fields. Local keypoints and the associated local features recently attracted attentions for their capability of characterizing salient regions which are invariant to certain geometric and photometric transformations [60], [61]. Over the past few years, keypoint-based approaches have shown success in object categorization [62], duplicate copy detection [63], semantic concept detection [64] and video search [65]. Figure 17 shows an example of keypoints detected inside of an image. The keypoints jointly describe the image content by characterizing the parts which are salient and informative enough to tolerate possible transformations (scaling, translation, rotation).

2.2.1 *Local Keypoint Detection*

Local Keypoint Detectors can be divided into contour based methods, signal based methods and methods based on template fitting. A briefly presentation is done for each cathegory. Contour based method idea is either to search for maximal curvature of inflexion points along the contour chains, or to search for intersection points after some polygonal approximation.

In signal based methods a measure indicates the presence of an interest point directly from signal. A signal based method uses the autocorrelation function of the signal [66], the squared first derivatives of the signal are averaged over a window. The eigenvalues of the resulting matrix are the principal curvatures of the autocorrelation function. If these curvatures are high, an interest point is detected. Förstner in his work [67] uses autocorrelation for pixel classification, the pixel is classified into three cathegories: region, contour, interest point. Heitger et at. [68]

extract 1D directional characteristics by convolving the image with orientation-selective Gabor filters. To obtain 2D characteristics, they compute first and second derivatives of the 1D characteristics. In template fitting method, the image signal is fitted to a parametric model of a specific type of interest point (a corner or a vertex). Another method for interest point detection is Lindeberg method [69], he introduced the concept of automatic scale selection. A local keypoint has its characteristic scale. The determinant of Hessian matrix and Laplacian to detect blob-like structures. In [70] Mikolajczyk et al. refined Lindeberg method creating robust and scale invariant feature detectors with high repeatability. They used a scale adapted Harris measure or the determinant of the Hessian Matrix to select the location, the Laplacian to select the scale. In [60] Lowe performed SIFT (Scale Invariant Feature Transform) , as it transforms image data into scale invariant coordinates relative to local features. SIFT (Scale Invariant Feature Transform) descriptors are generated by finding interesting local keypoints, in a greyscale image, by locating the maxima and the minima of Difference-of-Gaussian in the scale-space pyramid. SIFT algorithm takes different levels (octaves) of Gaussian blur on the input image, and computes the difference between the neighboring octaves. Information about orientation vector is then computed for each keypoint, and for each scale. Briefly, a SIFT descriptor is a 128-dimensional vector, which is computed by combining the orientation histograms of locations closely surrounding the keypoint in scale-space. The most important advantage of SIFT descriptors is that they are invariant to scale and rotation, and relatively robust to perspective changes. Several other scale invariant interest point detectors have been proposed. PCA-SIFT [71] uses PCA instead of histogram to normalize gradient patch. As a consequence of PCA utilization, the feature vector is significantly smaller than the standard SIFT feature vector. In [61] Bay et al. performed SURF descriptor, (Speeded Up Robust Features), SURF algorithm is slightly different than SIFT algorithm. SIFT builds an image pyramid, filtering each layer with Gaussians of increasing sigma values and taking the difference. SURF creates a stack without down sampling for higher levels in the pyramid, so the resulting images have the

same resolution. Kadir and Brady [72] performed a salient region detector through entropy maximization of the edge based region (region detector proposed by Jurie and Schmid [73]). From comparisons of detectors, [74] [59] Hessian based detectors result more stable than Harris based.

In the next paragraphs the most used local keypoint detector and descriptors will be discussed.

2.3. Harris Edge Corner.

Harris revisited Moravec's corner detector [75], it considers a local window in the image, and determines the average changes of the image intensity that result from shifting the window by a small amount in various directions.

- If the windowed patch is constant in intensity, then all shifts will result in a small change (E);
- if the windowed patch covers an edge, then a shift along the edge will result in a small change;
- A shift perpendicular to the edge will result in a large change.

This detector simply looks for local maxima in $\min(E)$ above some threshold value. Moravec's corner detector suffers from some problems, the set of shifts is considered at every 45 degrees, so the response of E is anisotropic. The response is noisy because the window is rectangular and binary. The operator responds too readily to edges because only the minimum of E is taken into account. The change E for the small shift (x,) can be written as:

$$E(x,y) = (x,y) M (x,y)^T \quad (30)$$

where

$$\mathbf{M} = \begin{bmatrix} \mathbf{A} & \mathbf{C} \\ \mathbf{C} & \mathbf{B} \end{bmatrix} \quad (31)$$

Let α and β be the eigenvalues of \mathbf{M} . α and β will be proportional to the principal curvatures of the local autocorrelation function, these form a rotationally invariant description of \mathbf{M} .

Harris detector is based on the second moment matrix, called **auto-correlation matrix**. This matrix must be adapted to scale changes to make it independent from image resolution.

Mikolajczyk et al. [74] formulated **Harris-Laplace** detector in which the scale adapted version of auto-correlation matrix is defined by:

$$\begin{aligned} \mu(\mathbf{x}, \sigma_I, \sigma_D) &= \begin{bmatrix} \mu_{11} & \mu_{12} \\ \mu_{21} & \mu_{22} \end{bmatrix} \\ &= \sigma_D^2 g(\sigma_I) * \begin{bmatrix} L_x^2(\mathbf{x}, \sigma_D) & L_x L_y(\mathbf{x}, \sigma_D) \\ L_x L_y(\mathbf{x}, \sigma_D) & L_y^2(\mathbf{x}, \sigma_D) \end{bmatrix} \end{aligned} \quad (32)$$

where σ_I is the **integration scale**, σ_D is the **differentiation scale** and L_a is the derivative computed in a direction. In a local neighborhood of a point, the gradient distribution is described by the matrix. Gaussian smoothing is applied to derivatives in the neighborhood of the point. Two principal signals of the neighborhood of a point are represented by the eigenvalues of the matrix. The signal change is significant in the orthogonal directions, corners, junctions etc. Harris detector is based on this principle. It combines the trace and the determinant of the second moment matrix:

$$cornerness = \det(\mu(\mathbf{x}, \sigma_I, \sigma_D)) - \alpha \text{trace}^2(\mu(\mathbf{x}, \sigma_I, \sigma_D)) \quad (33)$$

the Locations of interest points are determined by local maxima of coarseness.

Mikolajczyk and Schmid performed a newer version [74] of Harris detector combining the reliable Harris detector [66] with automatic scale selection [69] to obtain a scale invariant detector. More particularly, Harris descriptors are Gaussian derivatives that are computed in local neighborhood of interest points. The local neighboring patches of interest points are normalized, then the Harris Descriptors are computed as Gaussian derivatives are computed on the patches. These descriptors are invariant to rotation, affine transformations. Rotation invariance is obtained by steering derivatives in gradient direction [76], affine transformation invariance is obtained by dividing the first order derivatives by the higher order derivatives. The stability of the descriptor average gradient direction is used [77].

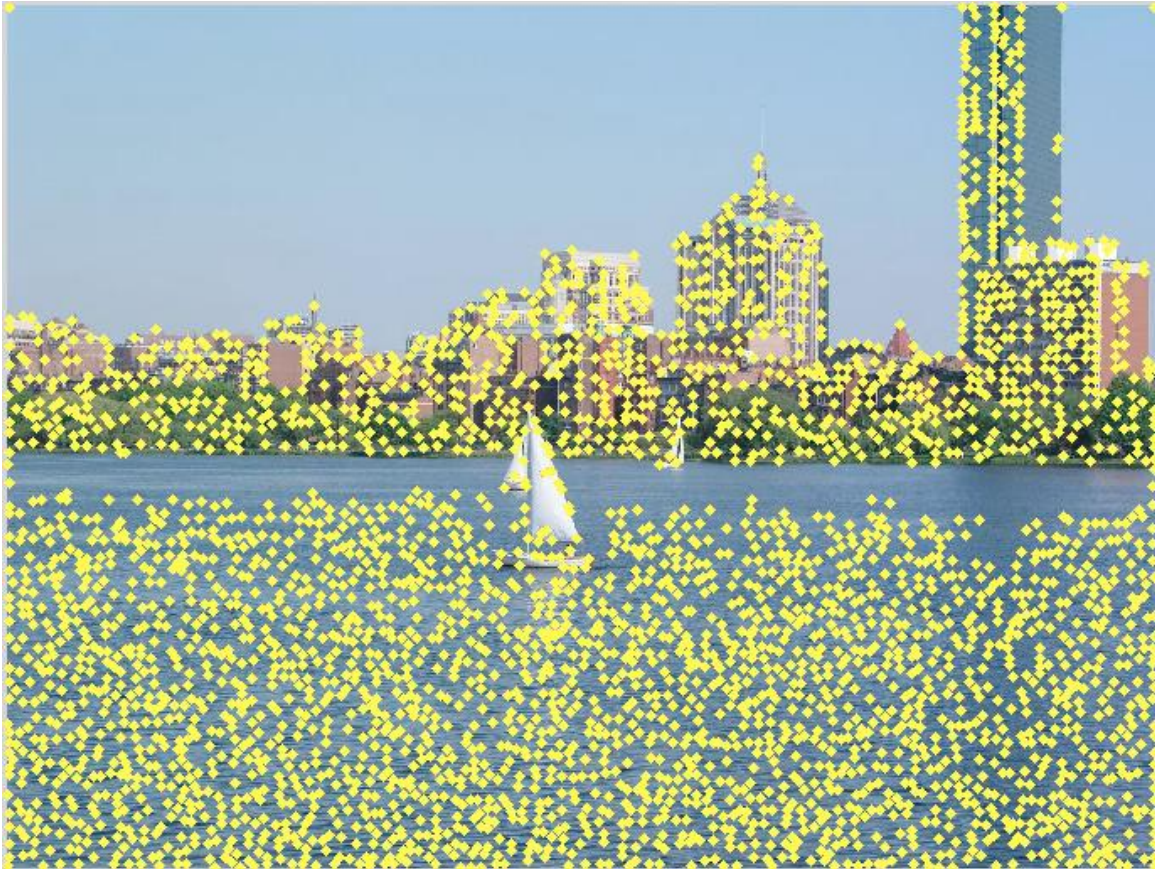


Fig. 17 Harris Keypoints - original version



Table 18 Harris-Laplace Keypoints

2.4. SIFT (Scale Invariant Feature Transform)

The major steps of SIFT algorithm are:

- Scale-space extrema detection;
- Keypoint Localization;
- Orientation Assignment

SIFT approach transforms image data into scale invariant coordinates relative to local features, it generates large numbers of features that densely cover the image over the full range of scales and locations (the numbers of keypoints extracted it depends on various parameter and on image content).

2.4.1 *Scale-space extrema detection*

In SIFT method keypoint detection is obtained using a cascade filtering approach that uses efficient algorithms to identify candidate locations that are examined in further detail. The method aims to find locations and scales that can be repeatably assigned under differing viewpoints of the same object. The scale space of an image [78] is defined as a function, $L(x,y,\sigma)$, that is produced from the convolution Gaussian $G(x,y, \sigma)$ with an input image, $I(x,y)$:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y), \quad (34)$$

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}, \quad (35)$$

Stable keypoints locations in scale space are obtained in Lowe method in the difference-of-Gaussian function convolved with the image, $\mathbf{D}(\mathbf{x}, \mathbf{y}, \sigma)$, which can be computed from the difference of the two nearby scales separated by a constant multiplicative factor k :

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma) \end{aligned} \quad (36)$$

The difference-of-Gaussian function provides a close approximation to the scale normalized Laplacian of Gaussian $\sigma^2 \nabla^2 G$. Lindeberg [78] showed that the normalization of the Laplacian with the factor σ^2 is required for true scale invariance.

$\nabla^2 G$ can be computed from the finite difference approximation to $\partial G / \partial \sigma$, using the difference of nearby scales at $k\sigma$ and σ :

$$\sigma \nabla^2 G = \frac{\partial G}{\partial \sigma} \approx \frac{G(x, y, k\sigma) - G(x, y, \sigma)}{k\sigma - \sigma} \quad (37)$$

$$\sigma \nabla^2 G \approx G(x, y, k\sigma) - G(x, y, \sigma) \quad (38)$$

When the difference-of-Gaussian function has scales differing by a constant factor it already incorporates the factor σ^2 of scale normalization (required for scale invariance). Adjacent image scales are subtracted to produce the difference-of-Gaussian images. Once a complete octave has been processed, Gaussian image, that has twice the initial value of σ , is resampled by taking every second pixel in each row and column.

2.4.2 Local Extrema Detection

Local Maxima and minima of $D(x,y,\sigma)$ are detected after that each sample point is compared to its eight neighbors in the current image and nine neighbors in the scale above and below. Local Maxima and minima are selected if sample point is larger than all of these neighbors or smaller than all of them. The determination of frequency of sampling in the image and scale domains is an important issue. This is needed to reliably detect the extrema. The best choices can be determined experimentally, by studying a range of sampling frequencies.

The scale-space difference-of-Gaussian function has a large number of extrema and it would be very expensive to detect them all. The detection of the most stable and useful subset even with a coarse sampling.

2.4.3 Keypoint Localization

The Keypoints candidates have been found by comparing pixels with their neighbors. Subsequent step is to perform data fitting for location, scale, and ratio of principal curvatures. This step rejects points with low contrast or poorly localized along edge. The location of extremum, \hat{x} is determined by taking the derivative of the

Taylor expansion of function $D(\mathbf{x})$, shifted so that the origin is at the sample point $\mathbf{x}=(x,y,\sigma)$:

$$D(\mathbf{x}) = D + \frac{\partial D^T}{\partial \mathbf{x}} \mathbf{x} + \frac{1}{2} \mathbf{x}^T \frac{\partial^2 D}{\partial \mathbf{x}^2} \mathbf{x} \quad (39)$$

$$\hat{\mathbf{x}} = -\frac{\partial^2 D^{-1}}{\partial \mathbf{x}^2} \frac{\partial D}{\partial \mathbf{x}} \quad (40)$$

The Hessian and the derivative of D are approximated by using differences of neighbors of sample point. The value of the function D at the extremum is used for rejecting unstable extrema with low contrast. Generally, all extrema with value less than 0.03 were discarded.

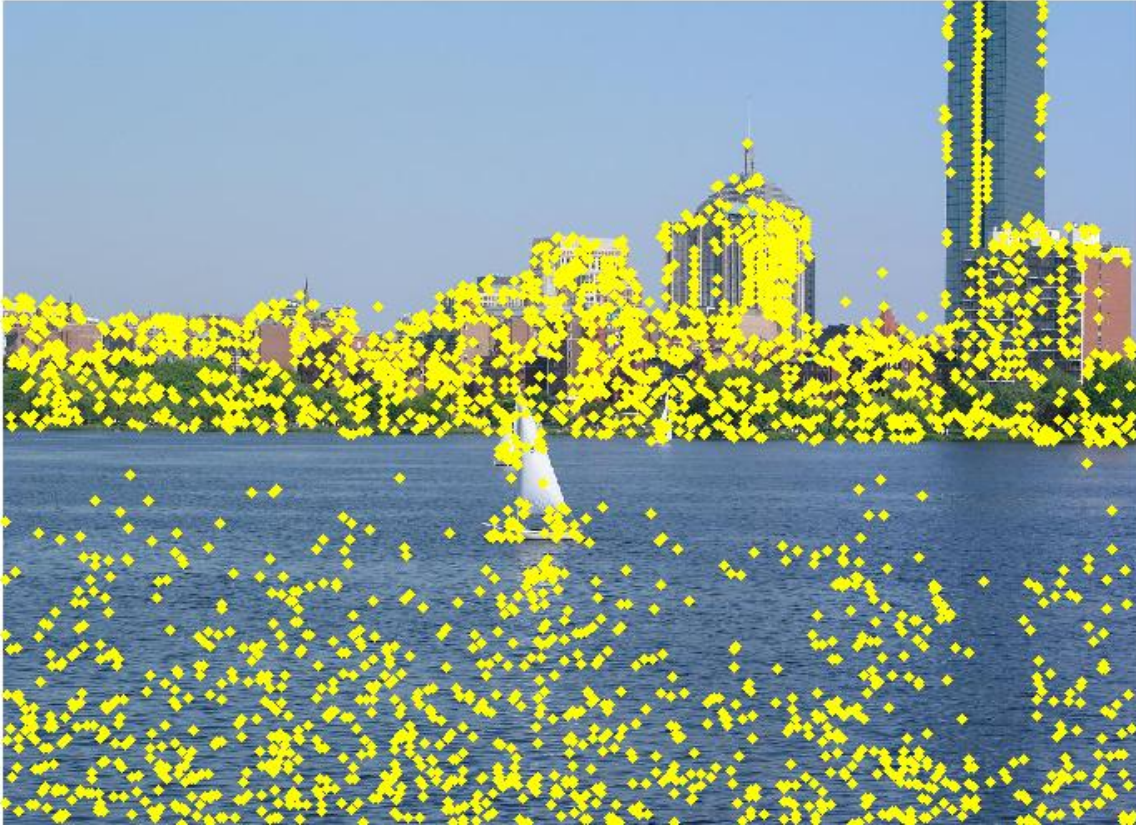


Fig. 19 SIFT keypoints

2.4.4 Orientation Assignment

A consistent orientation, based on local image features, is assigned to each keypoint. The scale of keypoint is used to select the Gaussian Smoothed image with the closest scale, so all the computations are performed in a scale-invariant manner. For each image sample, $L(x,y)$, at this scale, the gradient magnitude, $m(x,y)$, orientation $\theta(x,y)$, is precomputed using pixel differences:

$$m(x, y) = \sqrt{(L(x + 1, y) - L(x - 1, y))^2 + (L(x, y + 1) - L(x, y - 1))^2} \quad (41)$$

$$\theta(x, y) = \tan^{-1}((L(x, y + 1) - L(x, y - 1))/(L(x + 1, y) - L(x - 1, y))) \quad (42)$$

An orientation histogram is formed from the gradient orientations of sample points within a region around the keypoint. The orientation histogram has 36 bins covering the 360° range of orientations. Each sample added to the histogram is weighted by its gradient magnitude and by a Gaussian-weighted circular window with a σ 1.5 times that of the scale of the keypoint.

2.4.5 Local Image Descriptor

The SIFT descriptor implementation is inspired by Edelman idea [79]. Edelman proposed a representation based upon a model of biological vision, in particular of complex neurons in primary visual cortex. Complex neurons respond to a gradient at a particular orientation and spatial frequency. In fig.20 keypoint descriptor is depicted.

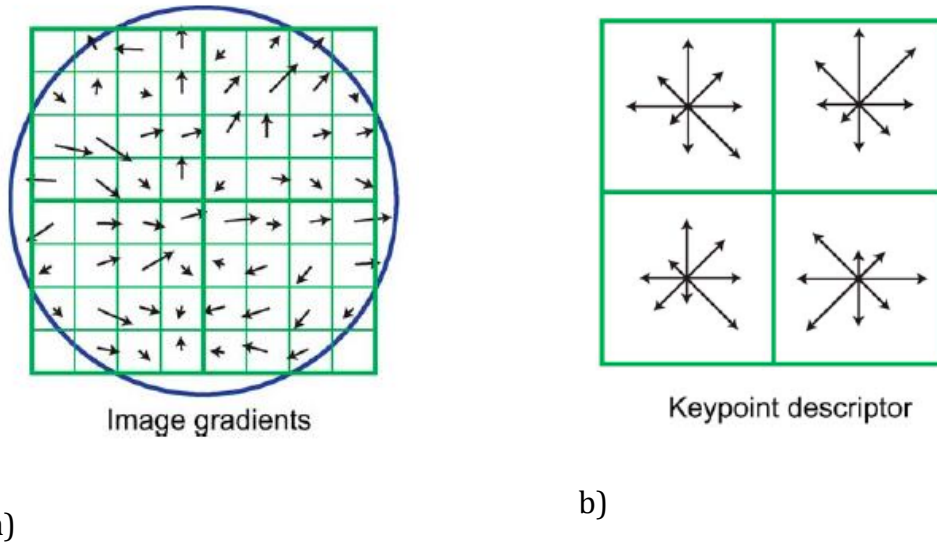


Fig. 20 The image gradient orientations for each subimage (a), the keypoint descriptor (b)

The keypoint descriptor allows for significant shift in gradient positions by creating orientation histograms over 4×4 sample regions. Eight directions for each orientation histogram are obtained. The descriptor is formed from a vector containing the values of all orientation histogram entries. The best results are obtained with a 4×4 array of histograms with 8 orientation bins. Each keypoint is represented from a 128 components vector ($4 \times 4 \times 8 = 128$). For more details about SIFT see [60].

2.5. PCA-SIFT detector

Lowe presented SIFT for extracting distinctive invariant features from images that can be invariant to image scale and rotation. Ke and Sukthankar [71] used PCA to normalize gradient patch instead of histograms. PCA-based local descriptors are also distinctive and robust to image transformations. PCA is used for dimensionality reduction [80], this is suited to represent the keypoint patches. The feature vector is smaller than the standard SIFT feature vector but it can be used with the same

matching algorithms. The input vector is created by concatenation of the horizontal and vertical gradient maps for the 41 x 41 patch centered to the keypoint, so the vector has $2 \times 39 \times 39 = 3042$ elements. The advantage of using PCA-SIFT is that fewer components requires less storage, these componenst furthermore result to a faster matching.

2.6. SURF (Speeded Up Robust Feature)

First of all, a briefly discussion about the concept of Integral Images is needed. Integral images allow for fast computation of box type convolution filters. Integral image $I_{\Sigma}(\mathbf{x})$ at location $\mathbf{x}=(x,y)^T$ represents the sum of all pixels in the input image I within a rectangular region formed by the origin and \mathbf{x} (fig. 21). The method for SURF detection is based on Hessian Matrix approximation. Integral images are used for a reduction of computational time.

$$I_{\Sigma}(\mathbf{x}) = \sum_{i=0}^{i \leq x} \sum_{j=0}^{j \leq y} I(i, j) \quad (43)$$

The sum of the intensities over any upright rectangular area, it takes only three additions. The calculation time is independent of its size.

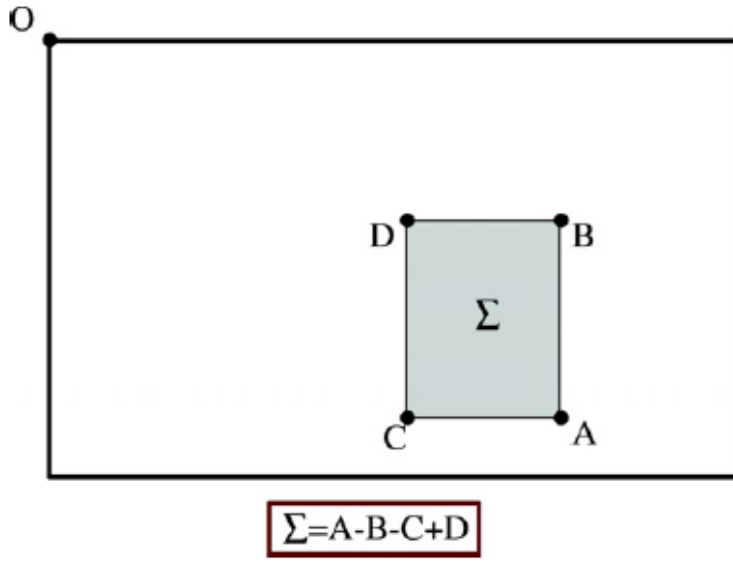


Fig. 21 Integral Image includes three additions to calculate the sum of intensities inside of a rectangular region of any size.

After this brief introduction about Integral Image a synthetic summary of SURF is given. SURF detector is based on Hessian Matrix, blob-like structures are detected at locations where Hessian determinant is maximum.

Given a point $\mathbf{x} = (x, y)$ in an image I , the Hessian matrix $H(\mathbf{x}, \sigma)$ in \mathbf{x} at scale σ is defined as follows:

$$H(\mathbf{x}, \sigma) = \begin{bmatrix} L_{xx}(\mathbf{x}, \sigma) & L_{xy}(\mathbf{x}, \sigma) \\ L_{xy}(\mathbf{x}, \sigma) & L_{yy}(\mathbf{x}, \sigma) \end{bmatrix} \quad (44)$$

$L_{xx}(\mathbf{x}, \sigma)$ is the convolution the Gaussian second order derivative with the image I in point \mathbf{x} , similarly for $L_{xy}(\mathbf{x}, \sigma)$ and $L_{yy}(\mathbf{x}, \sigma)$. Even if Gaussians are optimal for scale-space analysis, in practice they have to be discretized and cropped. Gaussian derivatives can be evaluated at very low computational cost using integral images. Box filters with size are approximations of a Gaussian with $\sigma = 1.2$ and represent the

lowest scale for computing blob-like structures. D_{xx} , D_{yy} , D_{xy} are the approximations with box filters.

$$\det(\mathcal{H}_{\text{approx}}) = D_{xx}D_{yy} - (wD_{xy})^2. \quad (45)$$

The weight w of the filter responses is used to balance the expression for Hessian's determinant. A constraint must be respected for the energy conservation between the Gaussian kernels and the approximated Gaussian kernels:

$$w = \frac{|L_{xy}(1.2)|_F |D_{yy}(9)|_F}{|L_{yy}(1.2)|_F |D_{xy}(9)|_F} = 0.912... \simeq 0.9 \quad (46)$$

where $|x|_F$ is the Frobenius norm. Weighting changes depending on the scale, experimentally keeping constant this factor has not impact on the results. The approximated determinant of the Hessian represents the blob response in the image at location x . Local maxima are detected among Hessian determinants at different scales.

2.6.1 Scale space representation

Scale spaces are usually represented as an image pyramid. The images are smoothed with a Gaussian and then subsampled in order to achieve a higher level of the pyramid. SURF approach, differently from SIFT, applies box filters of any size directly on the original image and even in parallel. The scale space is analyzed by up-scaling the filter size rather than iteratively reducing the image size. The main motivation for this kind of sampling is computational efficiency. Box filters preserve

high frequency components that, on the contrary, can get lost with downsampling in the other approaches. The scale space is divided into octaves. An octave represents a series of filter response maps obtained by convolving the same input image with a filter of increasing size. Each octave is subdivided into a constant number of scale levels. The minimum scale difference between two subsequent scales depends on the length l_0 of the positive or negative lobes of the partial second derivative in the direction of derivation (x or y). For a 9×9 filter the length l_0 is 3.

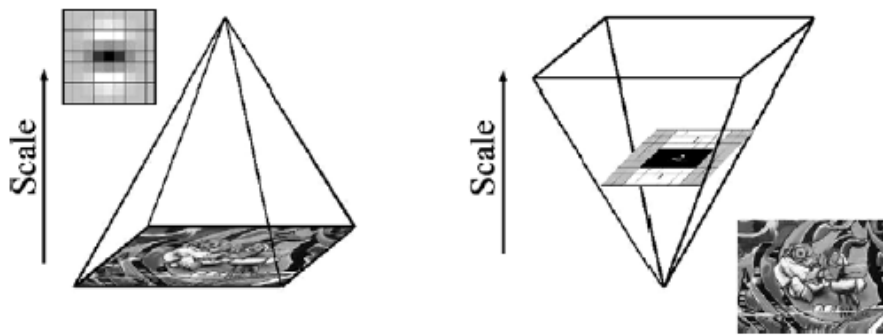


Fig.22 On the left, pyramidal downsampling. On the right, the use of integral images for up-scaling of the filter.

The construction of the scale space starts with the 9×9 filter, which calculates the blob response of the image for the smallest scale. Then filters with size 15×15 , 21×21 , 27×27 are applied. A graphical representation of the filter side lengths for three different octaves is shown in fig.23.

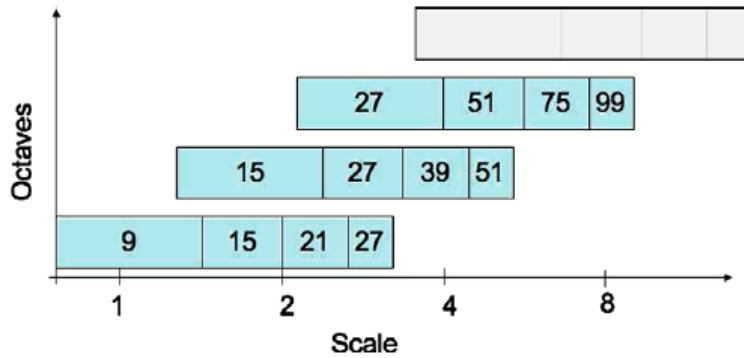


Fig.23 The logarithmic horizontal axis represents the scales, vertical axis shows the octaves. Filter side lengths for three different octaves are shown.

2.6.2 Interest Point Localization

For interest points localization in the image, Fast Invariant method [81] is used. Maxima of determinant of Hessian matrix are interpolated in scale and image space. Scale space interpolation is important because the difference in scale between the first layer of every octave is relatively large.



Fig. 24 SURF keypoints

2.6.3 Interest Point Description

SURF descriptors describe the distribution of intensity in interest point neighborhood, similar to SIFT approach [60]. More precisely, in SURF method first order Haar wavelet responses in x and y directions is computed (this is different from SIFT approach in which gradient is used). The integral images are exploited for speed in calculation, the descriptor consists in a 64 features vector. In the following paragraphs, the steps of orientation assignment and Haar Wavelet responses are approached.

2.6.4 Orientation

The orientation information is extracted from a circular region around the interest point. First, Haar Wavelet responses calculation in x and y directions within a circular neighborhood of radius $6s$ around the interest point (s is scale at which the interest point is detected). Sampling step and the size of wavelet are scale dependent: Sampling step is chosen to be s , wavelet size is set to a side length of $4s$.

Wavelet responses are calculated with a Gaussian ($\sigma=2s$) centered at the interest point. The dominant orientation is estimated by calculating the sum of all responses within a sliding orientation window of size $\pi/3$. Orientation window size is very important parameter which had to be chosen with care. Small sizes fire on single dominating gradients, large sizes tend to yield maxima in vector length that are not outspoken, these results give a misorientation of the interest point.

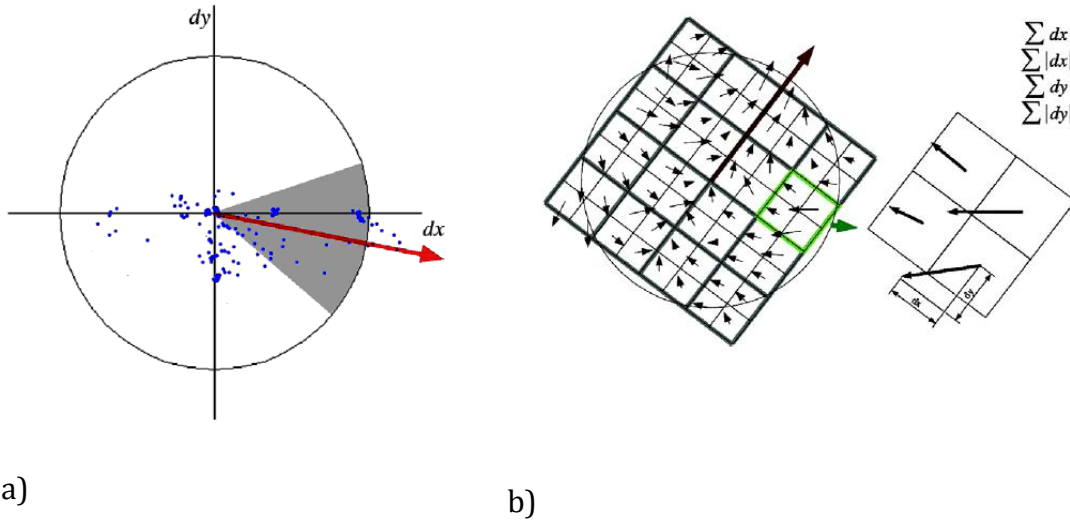


Fig.25 (a) A sliding orientation window of size $\pi/3$ detects the dominant orientation of the Gaussian weighted Haar wavelet responses. (b) An oriented quadratic grid 4 x 4 square sub regions is over the interest points, for each square, the wavelet responses

2.6.5 Haar Wavelet Responses

A square region is centered around the interest point and orientation (selected in previous step). The size of the region is $20s$ and it is splitted into smaller 4 x 4 square sub-regions. Haar wavelet responses are calculated for each sub-regions at 5 x 5 regularly spaced sample points. dx is **Haar wavelet** response in horizontal direction, dy is **Haar wavelet** response in vertical direction (horizontal and vertical with respect to the interest point orientation). dx and dy are weighted with a Gaussian ($\sigma=3.3$) centered at the interest point, as consequence of this operation the robustness towards geometric deformations and localization increased.

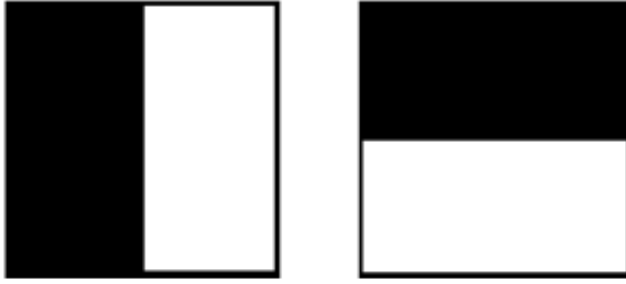


Fig.26 Haar Wavelet Filters (x (left) direction and y (right) direction)

The wavelet responses dx and dy are summed up over each sub-region and form a first set of entries in the feature vector. The sum of absolute values of the responses $|dx|$ and $|dy|$ is extracted to give information about the polarity of intensity changes. Each of 4×4 subregions has a 4D descriptor vector v for intensity structure:

$$\mathbf{v} = (\Sigma dx, \Sigma dy, \Sigma |dx|, \Sigma |dy|)$$

This results in a descriptor vector with 64 components. Wavelet responses are invariant to bias in illumination. Invariance to contrast is achieved by turning the descriptor into a unit vector.

2.7. Local Keypoint Comparisons

In this paragraph comparisons among the most used local keypoints are done. More particularly, SURF detector is compared to the Difference of Gaussians (DoG) detector by Lowe [60] called SIFT, and a version of Harris detector based on Hessian-Laplace detectors proposed by Mikolajczyk [74]. As criterion of evaluation Repeatability is chosen. The Repeatability expresses the reliability of a detector to find the same interest point under different viewing conditions. The number of interest points found is on average very similar for all detectors. The thresholds were adapted according to the number of interest points found with the DoG detector. The experimental results reported in figures 27, 28. are from [61].

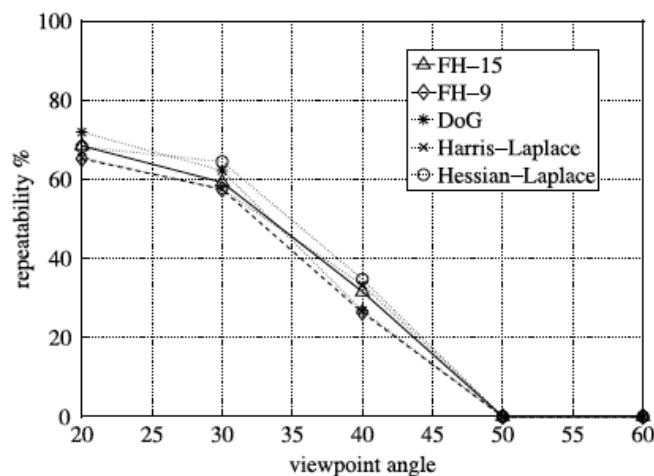


Fig.27 Repeatability in function of viewpoint angle

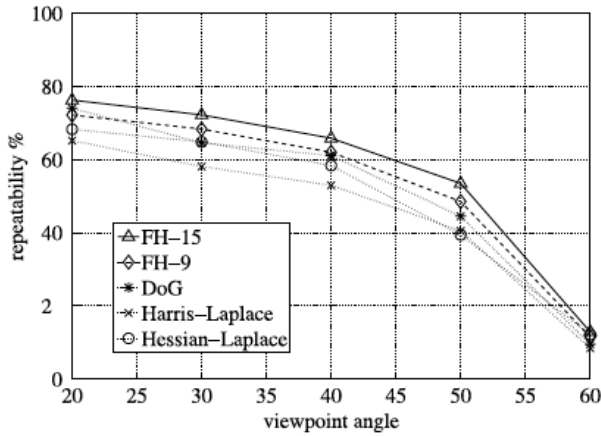


Fig.28 Repeatability in function of viewpoint angle

SURF detector is tested with two versions: FH-9, the Fast Hessian Detector with the initial filter size 9 x 9 and FH-15, the 15 x 15 filter on the double input image size version. The FH-9 detector is more than five times faster than DoG (SIFT detector) and ten times faster than Hessian-Laplace (Hessian-Laplace version of Harris Detector). The FH-15 detector is more than three times faster than DoG (SIFT) and more than four times faster than Hessian-Laplace. From the results showed in [61] the repeatability scores for SURF detectors (FH-15 and FH-9) are comparable or even better than for the competitors. The results reported in tab.1 are from [61].

Detector	Threshold	Number of points	Extraction time
FH-15 (SURF)	60000	1813	160
FH-9 (SURF)	50000	1411	70
Hessian-Laplace	1000	1979	700
Harris-Laplace	2500	1664	2100
DoG (SIFT)	Default	1520	400

Table 1 Comparisons Details: Detectors, Thresholds, Number of local keypoints and calculation time are reported.

Juan et al. [82] performed a further comparisons study of SIFT, PCA-SIFT and SURF. They uses KNN (K-Nearest Neighbor) and Random Sample Consensus (RANSAC) to the three methods in order to analyze the resultsof the methods application in recognition. KNN finds the matches and RANSAC [83] rejects inconsistent matches from which the inliers can take as correct matches. SIFT shows its stability in most situations although it is slow. SURF is the fastest with good performance. PCA-SIFT has good performance, especially in rotation and illumination changes. Juan in [82] focused their attention on keypoint matching for the evaluation of local keypoint detectors and descriptors. In eq. 47 the evaluation measurement: the ratio between the number of point-to-point correspondences that can be established for detected points and the mean number of points detected in two images.

$$r_{1,2} = \frac{C(I_1, I_2)}{\text{mean}(m_1, m_2)} \quad (47)$$

$C(I_1, I_2)$ is the number of corresponding pair, m_1 and m_2 are the numbers of the detector. This measure evaluates the performance of finding matches. Test are done to analyze the robustness against scale changes, view changes, illumination changes and rotation.

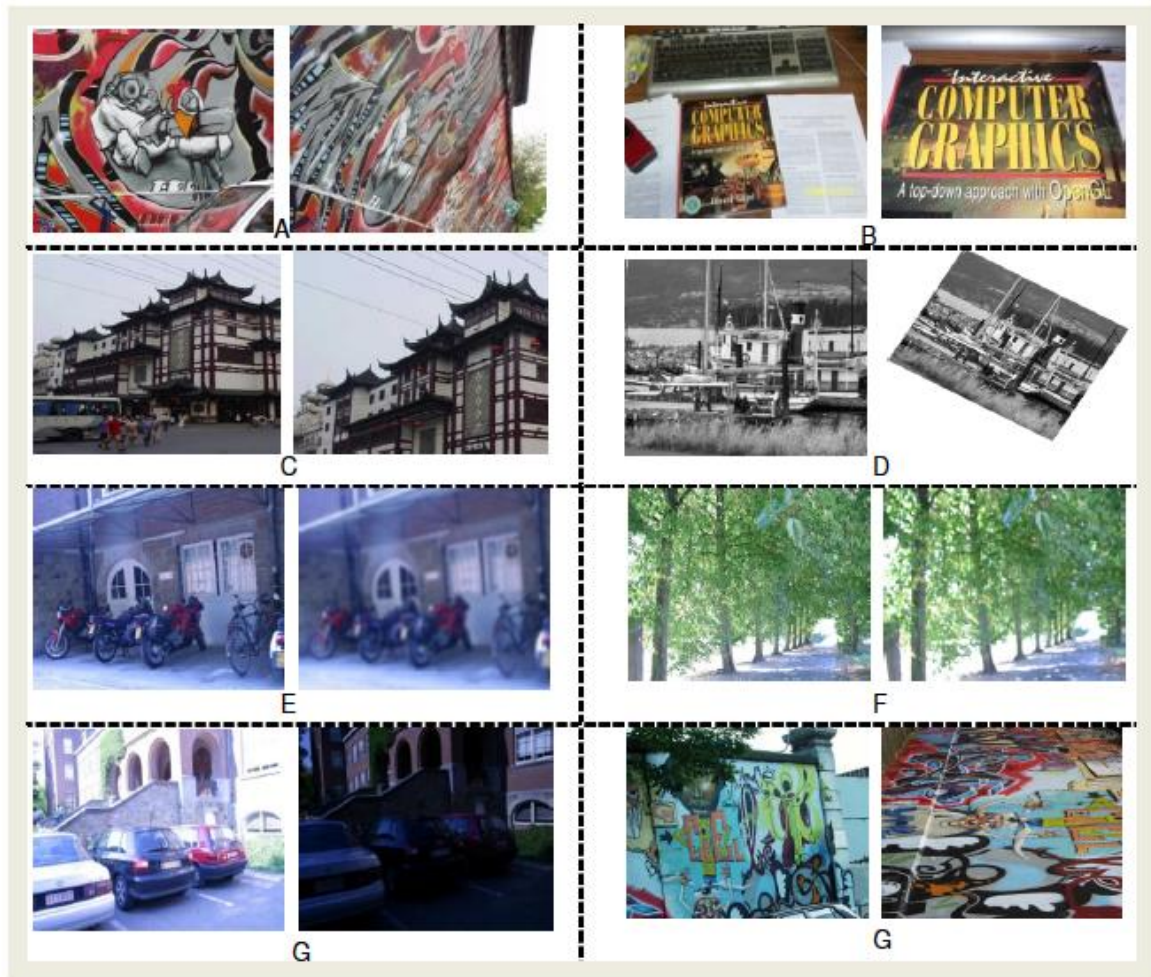


Fig.29 - A and H are the affine transformed images. B and C - scale changed images. D - rotation images. E and F are the blurred images. G - illumination changed images. The figure is from [82].

- The first experiment in [82] analyzes the **processing time** as measure of evaluation. The time counted includes feature detection and matching phase. SURF is the fastest one, SIFT is the slowest but it finds most matches. Graffiti dataset as shown in group A of fig.29 is used for processing time analysis.
- **Scale changes** comparisons are done in second experiment (images from group B and C of figure 29. The evaluation is done by considering the

matching number of the three methods. The result is : when the scale get higher values, SIFT and SURF are much better than PCA-SIFT because PCA-SIFT is not stable as SIFT and SURF.

- The influence of **rotation** on SIFT, SURF and PCA-SIFT, is analyzed in third experiment. It shows that SURF doesn't work well, it finds the least matches and gets the least repeatability. PCA-SIFT found only one correct match of the first ten matches.
- The fourth experiment gives a measure of robustness of the three methods against **blur**. Images from groups E and F of fig.29 are blurred with Gaussian blur. SURF and PCA-SIFT shows good performance. When the radius get larger, SURF detects few matches and PCA-SIFT finds more matches but smaller correct number. SIFT outperforms SURF and PCA-SIFT.

The illumination effect is analyzed in fifth experiment. Images from group G of fig.29 get illumination changes. Under these conditions SURF shows the best performance (31% of repeatability)

In last experiment the stability againts affine transformations is studied. Images feom groups A and H are used, the viewpoint change is 50 degrees. The experiment shows that SURF and SIFT have good repeatability when the viewpoint transformation is small. When the viewpoint transformation get higher values SURF repeatability decreases up to 0 matches detected. On the other side PCA-SIFT shows better performance when viewpoint transformation get higher values.

SIFT is invariant to rotation, scale changes and affine transformations but it is slow and not good at illumination changes. SURF is not robust against rotation and illumination attacks but it has good performance as the same as the SIFT. This section can be concluded saying that there is not a best descriptor methods. It depends on the application. For more details in local descriptors comparisons see [82].

It is very important to emphasize that local keypoints are often used to finding point correspondences between two images of the same scene or object (matching points). The local keypoint descriptors are used for object matching analysis inside of an image, image registration, image retrieval. In the next chapters the most important local keypoint descriptors will be compared in texture features, visual saliency and image forensics applications.

3. TEXTURE SCALE DETECTION

3.1. Introduction

Images generally represent different surfaces in the scene, each surfaces are made of distinct parts. Texture indicates visual patterns in real and synthetic scenes. More precisely, surfaces with discontinuities in depth, material, illumination give rise to texture subimages, in contrast, the surfaces made with homogeneous material, under smoothly varying illumination and smooth variation of image brightness, give rise to non-texture subimage. As seen in paragraph 1.3, the discrimination between texture and non-texture subregions of the image is done also by discovering distinct texels (or textons), in the image. Each group of textons defines the corresponding texture. Due to complexity and diversity of natural images, textures analysis, description and synthesis are very challenging tasks. There are several approaches for texture inspection, including statistical approaches of autocorrelation function, optical transforms, digital transforms, gray tone co-occurrence, run lengths, and autoregressive models [6].

Structural texture analysis focuses primarily on identifying periodicity in textures or identifying the position of texels (in this paper we use the words "texel" and "texton" as synonyms). In literature the Texton has not a precise mathematical definition, although in 1981 Julesz [13] defined it as the "putative unit of pre-attentive human texture perception". In this chapter a texel is defined as "a basic repetitive element of a texture pattern". Texture "scale" is intended to be defined as the size of texture elements (texels or textons) that occur most frequently in the

image.

Textures may be classified, in terms of periodicity and regularity of their structure, in five classes: regular, near regular, irregular, near stochastic and stochastic.

- Regular textures are simply periodic patterns where the intensity of color and the shape of all texture elements are repeated in equal intervals;
- Near Regular textures are a statistical distortion of a regular pattern [84];
- Irregular textures present deformation fields from regular patterns;
- Near Stochastic and Stochastic textures show typically dots and shapes randomly scattered over all the image.

A tile is the fundamental region that can be used to describe a regular texture, by modeling a 2-D plane. Near Regular Textures can be also viewed as statistical departures from regular textures along different dimensions. Texture can be arranged by regularity variations in geometric space and it can be also arranged in function of texture spectrum (see fig. 30). Many scenes in real world do not have a regular structure but a near regular one (buildings, wallpapers, floors, tiles, windows, fabric, pottery and decorative arts, animal fur, gait patterns, feathers, leaves, waves of the ocean, and patterns of sand dunes), so for natural scenes the problem is to detect the scale, and to extract texels, also in case of a near regular structure. To simulate the real world on computers, near-regular textures deserve special attention. Texture scale value can be used in many image processing applications such as image segmentation, texture description, Content Based Image Retrieval and texture synthesis. Texture image segmentation, for example, is done by discovering distinct cohesive groups of spatially repating texels in the image. Different kinds of texture patterns are defined by corresponding texel size. Texture scale estimation is still a challenging problem in scientific community. In this chapter a novel approach to detect texture scale is presented, in case of regular and near regular structures, by means of a new instrument: the Keypoint Density Maps. Furthermore a method to extract the texel from an image is proposed, once the scale

has been detected.

The chapter is organized as follows: section 3.2 discusses some state-of-the-art techniques; section 3.3 introduces the concept of Keypoint Density Maps; section 3.4 presents scale detection method; in section 3.5 texel extraction technique is presented; in section 3.6 experimental results are discussed and compared to a state of the art method; a conclusive section ends the paper.

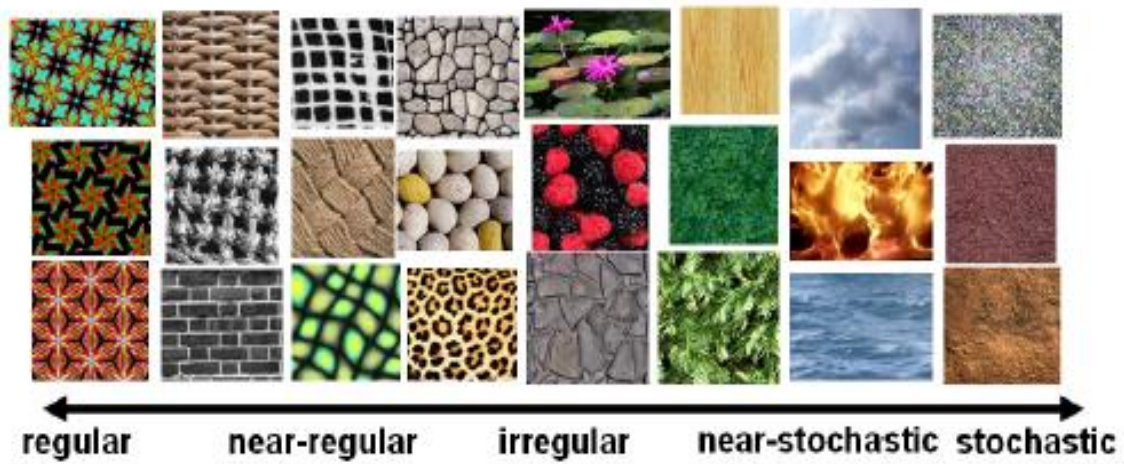


Fig. 30 Texture spectrum - texture patterns

3.2. State of the art

Texture scale detection is not a young research field in image processing, but today it remains one of the most challenging issue in Computer Vision. The first attempts to use a structural approach to extract periodicity from a texture were based on autocorrelation functions [85,86]. Both methods have been widely used but they achieved poor results in case of natural textures. In [87] Malik et al. treated textons as frequently co-occurring combinations of oriented linear filter outputs.

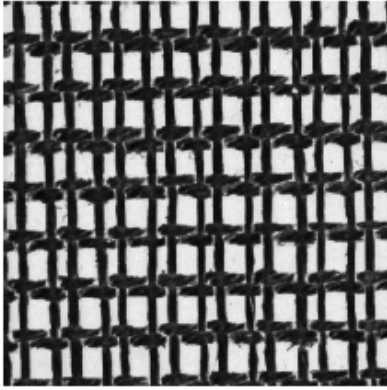
They built a universal texton vocabulary by processing a large number of natural images. Each pixel of the image is mapped to its nearest texton by using a K-means clustering approach. A similar approach was presented by Zhu et al. [88] who adopted a generative image model, in which an image is a superposition of bases from an over-complete dictionary of "textons". The last two methods are optimized for texture segmentation but do not focus on texture scale detection. Hong et al [89] proposed a scale descriptor and an energy minimization model to find the scale of a given texture.

They look for the smallest local patch whose probability density function is similar to the one computed on its neighboring local patches. Grigorescu et al in [90] identified texels in regular textures by searching for the smallest window through which the minimum number of different visual patterns is observed- This corresponds to find the smallest window that minimizes Rényi's generalized entropy (see fig.32). Grigorescu's method performs fair results with periodic textures, but it does not work as well with near regular ones.

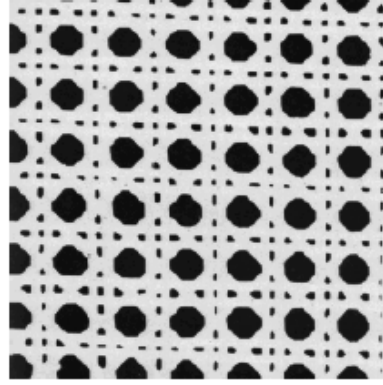
In [91] image texture is represented by a segmentation tree whose structure captures the recursive embedding of regions obtained from a multiscale image analysis. This method is oriented for extracting texels and not for direct scale computing.

In [92] the authors presented an approach for discovering texture regularity, by

formulating lattice finding as a higher-order correspondence problem, with an iterative approach and post-warping.

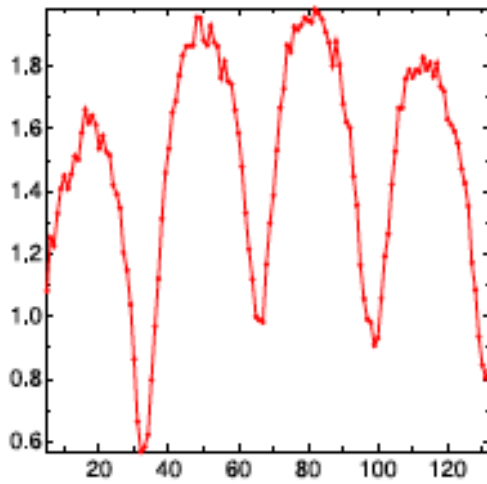


a)

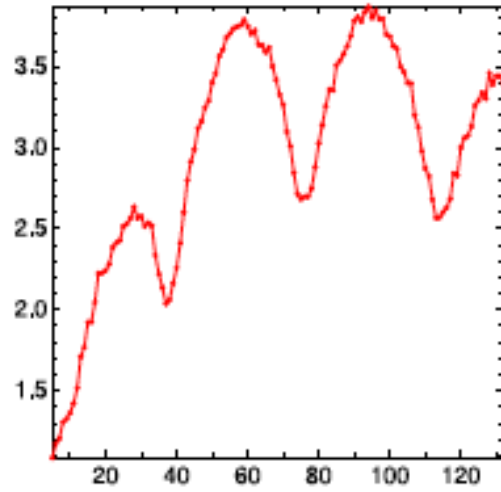


b)

Fig. 31 Two regular texture images from Brodatz Database (a) D20 and (b) D101



a)



b)

Fig.32 Rényi's quadratic entropies of the textures in Fig. 31. The x-axis represents the window size w , the y-axis represents the quadratic Rényi Entropy. The texture scale value is the smallest local minimum of the function.

3.3. Keypoint Density Map

The method presented in this chapter of this thesis uses three different type of algorithms to extract the interest points of an image: SIFT, SURF and Harris corner. The detectors and the descriptors of these local keypoints have been also discussed in the second chapter of this thesis.

In this section the new concept of Keypoint Density Map (KDM) is presented, KDM is the evolution of the SIFT Density Map, a model that we presented in my previous work [93]. A KDM is a representation of the density of a distribution of the keypoints in an image and can give essential information about the regularity of its structure.

Suppose to have an image I , $M \times N$, with an extremely regular distribution of keypoints (fig.33), the average number of pixels per keypoint N_p is:

$$N_p = \frac{M \cdot N}{n}. \quad (48)$$

where n is the number of keypoints in the image. Considering

$$s_1 = \sqrt{N_p} \quad (49)$$

a squared area of size $s_1 \times s_1$ will include only one keypoint, regardless of its position in the image (fig.33). In general, a squared area of size $s_k \times s_k$ where

$$s_k = \sqrt{k \cdot N_p} \quad (50)$$

will include, on average, k keypoints. . Due to construction, it can asserted that the patch will include always exactly k keypoints only when k^2 is integer.

A Keypoint Density Map KDM^k (fig. 34.b) is built by counting the number of keypoints into a sliding window of size s_k . Each point in the KDM^k indicates the number of keypnts into a squared area of size s_k , centered in corresponding point of the image. The KDM mode vector is defined as the vector MV of the dominant modes m^k $k=1...k_{\max}$, where m^k is the most frequent value in the map KDM^k (fig.34.c). k_{\max} is limited by the image size:

$$k_{\max} = \frac{1}{N_p} \cdot s_{k_{\max}}^2$$

$$s_{k_{\max}} = \frac{1}{2} \min(M, N)$$
(51)

Due to truncation in eq.50, (k values must be integers) there is no 1-1 relationship between k and KDM^k , because the same value of s_k (and then the same KDM) could derive from different values of k. This reduces, in practice, the number of maps to be computed. This number is typically less than a half of the image size. In case of very regular textures the KDM mode vector is linear, or quasi linear, with k (for construction, linearity is assured only when k^2 is integer). The progress of the mode vector will be used into the next step to estimate the scale of the image texture.

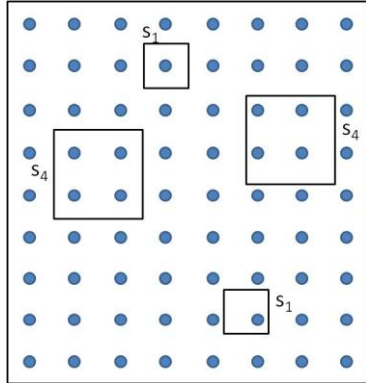


Fig.33 An image with a very regular distribution of keypoints (blue dots).

3.4. Texture Scale Detection

In the previous section very regular distributions of keypoints (fig.33) have been taken into account. In this case the KDM mode vector is quasi linear with the parameter k . For real images the linear model assumption is not verified for all the possible values of k . To estimate the scale the method looks for the subset of values of k which better fit the linear model M_l of the mode vector. In particular the mode vector MV is subsampled in the following way:

$$\begin{aligned} M(k_i) &= MV(n \cdot k_i) \\ n &= 1, 2, 3 \dots n_{\max} \\ k_i \cdot n_{\max} &\leq k_{\max} \quad . \\ k_i &= 1, 2 \dots \frac{k_{\max}}{p_{\min}} \end{aligned} \quad (52)$$

where k_i is the scale to be investigated, k_{\max} is defined in eq.51 and p_{\min} is discussed above. $M(k_i)$ is a vector containing a subset of the values of MV starting from k_i , in arithmetic progression with common difference k_i , until the end of the vector. To ensure that $M(k_i)$ will contain at least p_{\min} points, the starting k_i is limited to the ratio of k_{\max} and p_{\min} . This limits also the maximum scale observable with this method:

$$s_{\max} = \sqrt{\frac{k_{\max} \cdot N_P}{p_{\min}}} \quad (53)$$

For a more accurate analysis, terms should be taken in quadratic progression:

$$n = 1, 4, 9, 16 \dots n_{\max} \quad (54)$$

but this would limit the maximum observable scale, if it has been supposed to work with at least p_{\min} points.

$$s_{\max} = \sqrt{\frac{k_{\max} \cdot N_p}{p_{\min}^2}}. \quad (55)$$

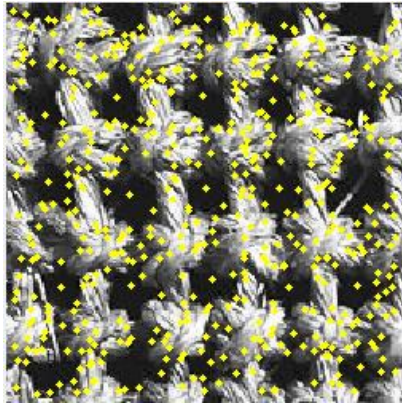
Then, it has been decided to subsample MV with terms in arithmetic progression, at the expense of some inaccuracy. To estimate the scale of the input image it has been selected the subsampled vector M which minimize an error function with respect to the linear model M_l . The corresponding starting value k_s indicate the estimated scale s for the input image:

$$\begin{aligned} s_{k_s} &= \sqrt{k \cdot N_p} \\ k_s &= \arg \min e(M(k_i), M_l(k_i)) \end{aligned} \quad (56)$$

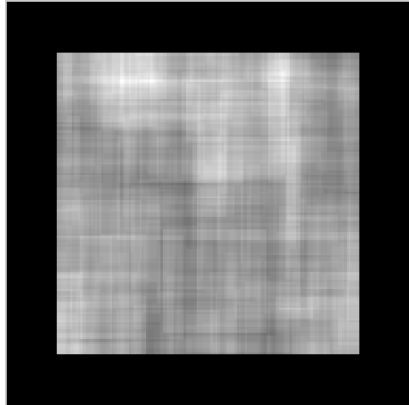
Two different error functions e have been chosen: the mean square error (MSE) (fig.34.d) and the maximum absolute displacement (MAD) (fig.34.e)

$$\begin{aligned} MSE(k_i) &= \sum_n (M_n(k_i) - n \cdot k_i)^2 \\ MAD(k_i) &= \max_n |M_n(k_i) - n \cdot k_i| \end{aligned} \quad (57)$$

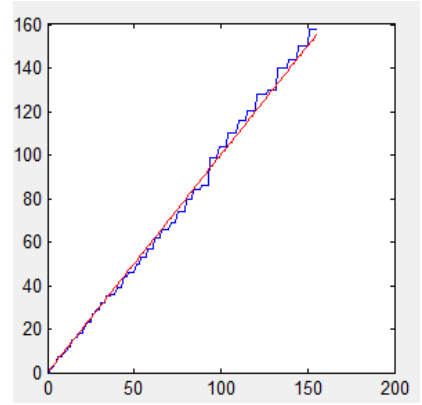
with the same constraints of eq. 51. Experimental results with the two error functions will be discussed in section 3.6



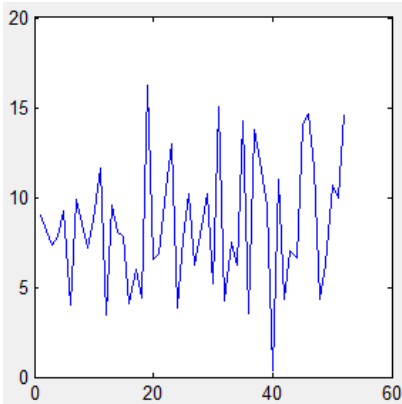
a)



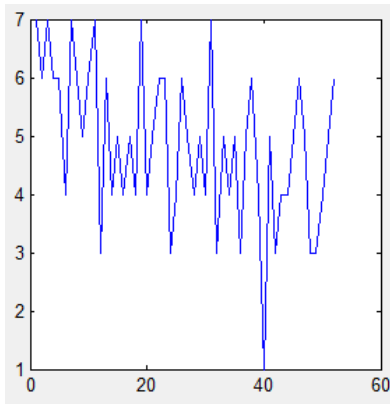
b)



c)



d)



e)



f)

Fig. 34 An example of regular texture (a), the corresponding Keypoint Density Map (SIFT) with $k=39$ (related to the estimated scale $s_k=38$) (b), the Mode Vector (c), the MSE function error (d), the MAD function error (e) and the detected texel(f). Note that the black area on (b) depends on the size of the sliding window, as it cannot overstep the image borders.

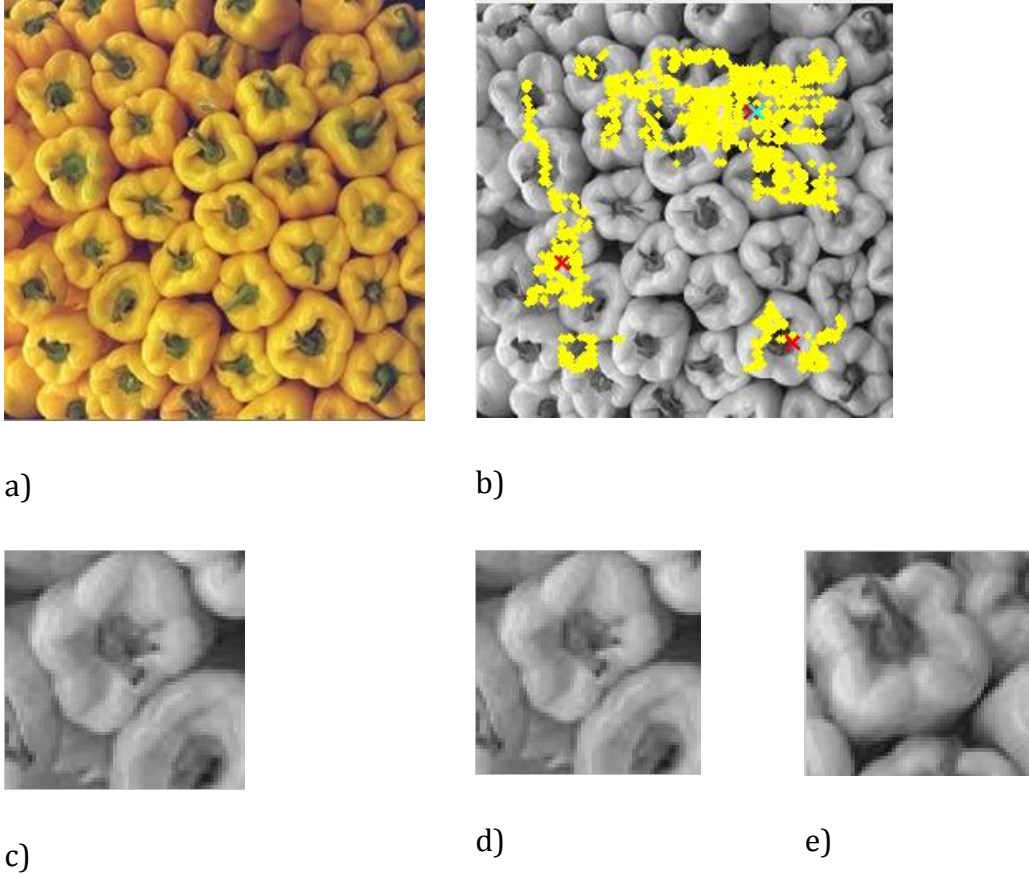


Fig. 35 Input texture (a), clustered blocks (b), and centroids (c,d,e). Extracted texel is (e)

3.5. Texel Extraction

Once the scale has been detected, the texture element (texel) can be extracted from the input image. If s_k is the estimated scale, and k the related index, the corresponding Keypoint Density Map $KDM(k)$ is selected, and all the values in the map which are equal to the most frequent value $MV(k)$ (note that in general $MV(k)$ and k may be slightly different). Each point in the map $KDM(k)$ correspond to a block of size s_k . Only those blocks which contains exactly $MV(k)$ keypoints are selected. Block coordinates are then grouped using mean shift clustering and only centroids are analyzed, to reduce computational effort. In order to find the most representative block, keypoints of the selected blocks are matched, comparing their descriptor vectors (only in case of SIFT and SURF). If B_i and B_j are the two blocks to

be matched, for each keypoint $i \in B_i$ its vector of descriptors d_i is compared with the vectors d_j of all the points $j \in B$. For efficiency, rather than using Euclidean distance, the angle between the two vectors is computed:

$$\alpha_{ij} = \arccos(\bar{d}_i \cdot \bar{d}_j) \quad (58)$$

and the two minimum angles:

$$\begin{aligned} \alpha_1 &= \min_{j \in B_j} \alpha_{ij} \\ \alpha_2 &= \min_{j \in B_j - j_1} \alpha_{ij} \end{aligned} \quad (59)$$

where j_1 is the point in B_j which vector j has the minimum angle with vector i . To increase robustness, matches are accepted only if the ratio of the two minimum angles, α_1 and α_2 , is less than a threshold (0.6, as suggested by Lowe [60]). Note that, for small angles, this is a close approximation to the ratio of Euclidean distances. We further apply RANSAC (RANDOM SAMPLE CONSENSUS)[83] to the matching points to filter outliers. The block selected is the one which maximizes the average number of matches with all the other blocks, after RANSAC, as representative of the texture. Two blocks will match also in case of rotation or scaling, as SIFT and SURF descriptors are invariant to geometrical transformations. Therefore texel extraction method works also in case of near regular textures, in which texels may have slightly differences in orientation and scale (fig.35).

3.6. Experimental Results

In this section the results of the method are compared with those of the Grigorescu and Petkov [90] method. Tests were made onto two different datasets: the PSU Near Regular Texture Dataset (PSU Dataset) which is composed of 45 images with regular and near regular structure, and 80 images from the 2.1D Textures Dataset (UIUC Dataset) which is composed of natural homogeneous textures. I created a ground-truth by manually selecting the scale for all the images in the datasets. Tests were executed on an Intel Core i7 PC (4 CPU, 1.6 GHz per processor, 4 GB RAM), exploiting the Matlab parallel library to make 4 workers run simultaneously. Tests are made with the two error functions described in eq. 57, with two different values of the minimum number of points, and using the three types of interest points described in chapter 2. The PSU dataset is split into “small” (scale ≤ 55) and “large” (scale > 55) textures, to show the influence of the input parameters in the two cases (i arbitrarily selected 55 as threshold to split the dataset into two subsets with more or less the same number of images). To evaluate results, the average relative absolute error is measured:

$$E = \frac{|s_k - s_{GT}|}{s_{GT}} \quad (60)$$

Error of our method and of the reference method, with respect to the ground-truth s_{GT} .

Figures 36, 37, 38, show the results within the PSU database. In case of large textures, in almost all the tests the method presented performs better than Grigorescu’s method, but in some cases (i.e. using the SIFT keypoints, MSE error

function and $p_{\min}=2$) the method presented outperforms it (21% vs 31%). In case of small textures, the SURF and the SIFT based approaches achieve more or less similar results (38%) than Grigorescu, but better results are observed when using SURF, MSE and $p_{\min}=3$ (33%). Harris based has the worst results. As expected, results with large textures improve if we use $p_{\min}=2$, because larger scales can be observed. Otherwise results improve for smaller textures in case of $p_{\min}=3$, as the accuracy increases (a larger number of points to compute the error function), while the maximum detectable scale is reduced.

Within the 2.1D Texture dataset i achieved very satisfactory results, similar to those obtained within the PSU Dataset for larger scales (fig.39), and remarkable if compared to those obtained with Grigorescu's method. In this case i do not split the dataset, as almost all the images show regularity at large scales. Results show also that the method presented works as well with regular and near regular textures (fig. 36,37,38, 39), while Grigorescu's is optimized only for regular ones.

In term of efficiency, execution time strongly depends on the texture scale, as in case of lower scales the novel method needs to build more KDMs. Nevertheless the method presented is faster than the reference method. In fact while Grigorescu's method investigates all the possible scales from a minimum value to the maximum value (a quarter of the image size), the new method analyzes only a subset of the possible scales, according to the rule in eq.50 (in our tests, 50% of the whole range of scales), therefore saving a lot of computational effort. Finally, execution time with this method is halved (180s vs 90s in average for medium sized images), with respect to the reference method.

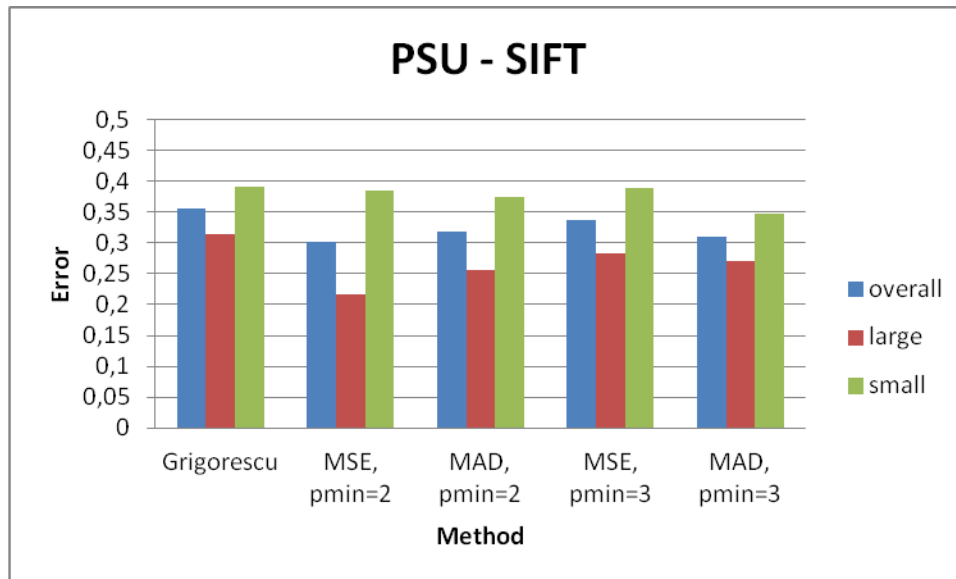


Fig. 36 The relative absolute error, with respect of the ground truth, for the PSU database. Values are measured for reference (Grigorescu) and my method, with different parameters and different keypoint extraction algorithms. (SIFT)

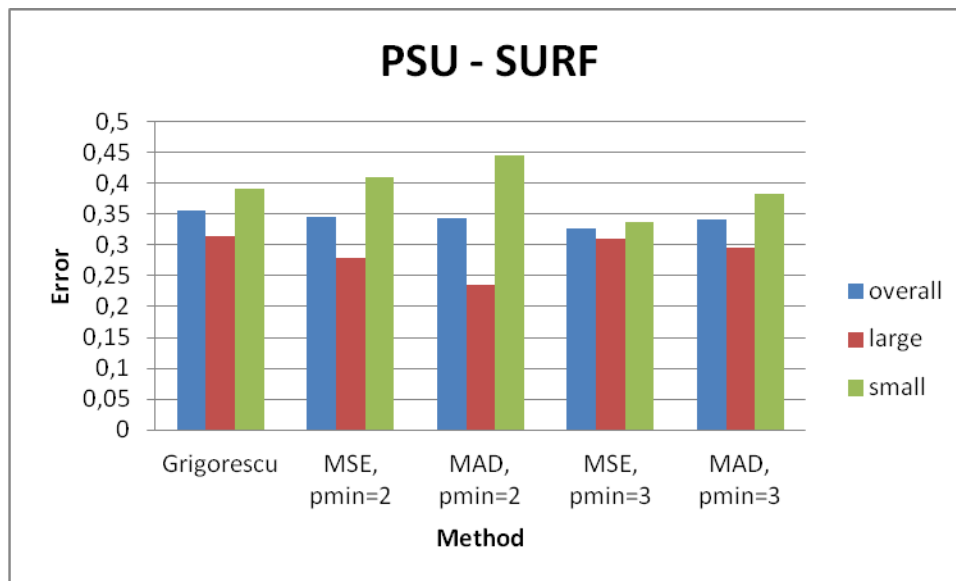


Fig. 37 The relative absolute error, SURF-keypoints - database PSU

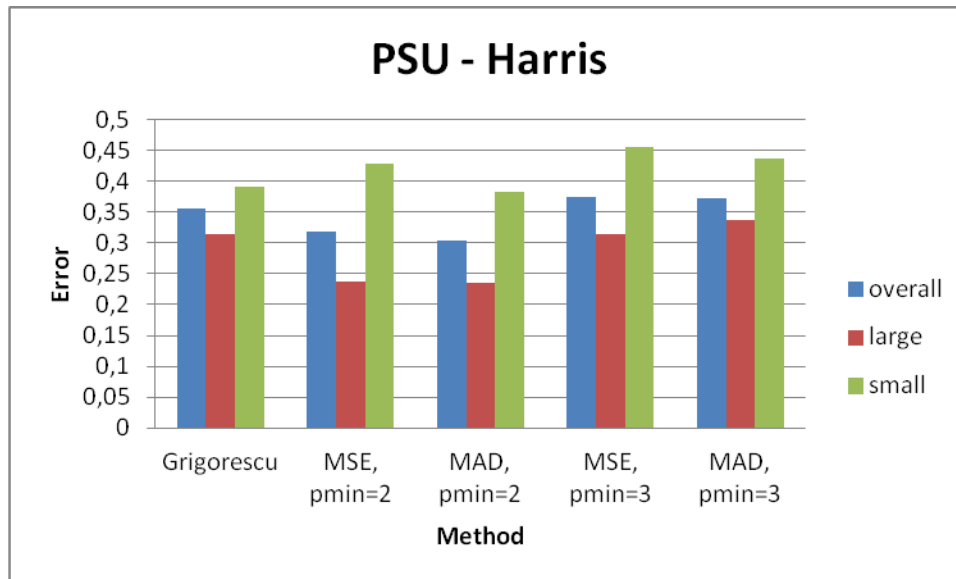


Fig. 38 The relative absolute error, HARRIS-keypoints - database PSU

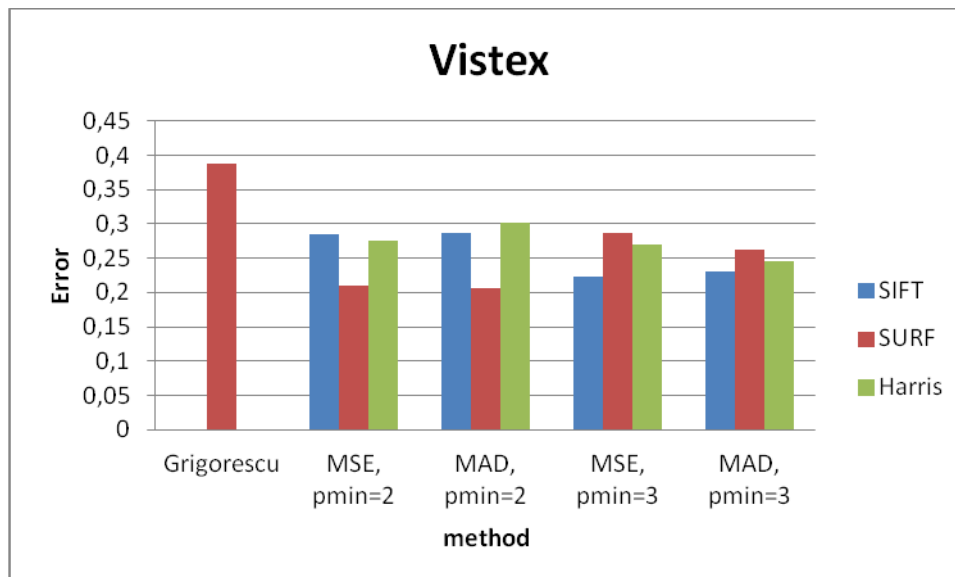


Fig. 39 The relative absolute error, with respect of the ground truth, for the Vistex database. Values are measured for reference (Grigorescu) and my method, with different parameters and different keypoint extraction algorithms

3.7. Conclusions

Detecting scale in textured images is a very hard task, and a relatively unexplored problem. In the work presented in this chapter the distribution of the interest points in the image is analyzed, by means of the Keypoint Density Maps, a novel instrument for Image Analysis applications. To our knowledge, there are no other approaches in literature that use interest points distribution to analyze the regularity of a texture. The results are very encouraging: with respect to the reference method, experiments showed that the novel method achieves similar precision in case of small textures, but outperforms it in cases of large textures, and in case of near-regular textures (figures 40, 41). Furthermore, Keypoint Density Maps (with SIFT) have been yet successfully used in visual saliency detection, and may be a versatile instrument for several image analysis application: texture discrimination or description, image segmentation, etc.

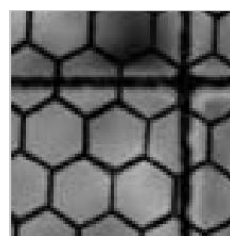
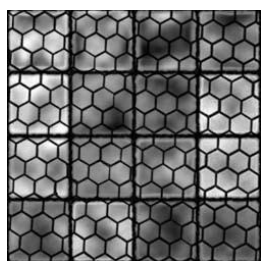
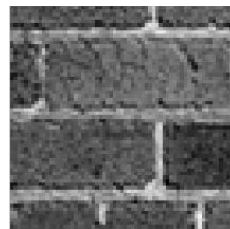
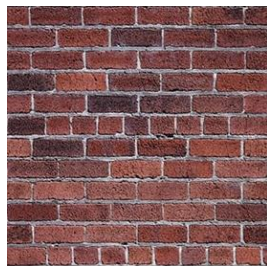
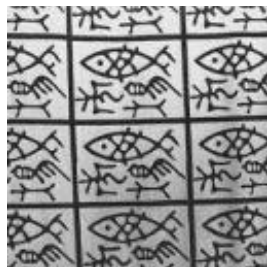
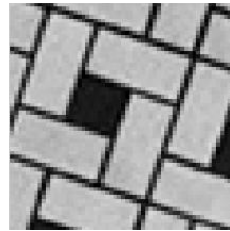
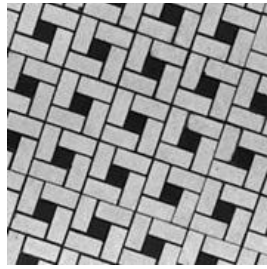


Fig. 10 Some visual examples of texture (left) and texels extracted with my method (right).

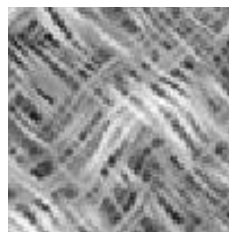
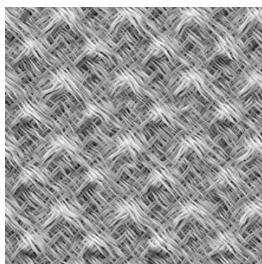
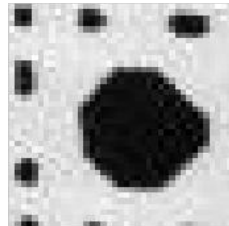
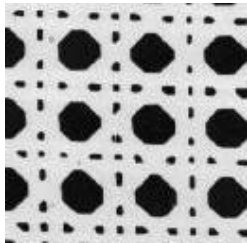
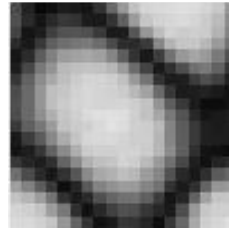
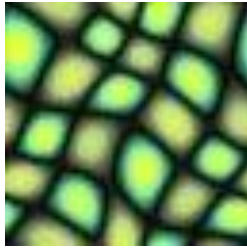


Fig. 41 Some visual examples of texture (left) and texels extracted with my method (right).

4. VISUAL SALIENCY

4.1. Introduction

One of the most challenging issues in Computer Vision field is the detection of salient regions in an image. Psychovisual experiments [94] suggest that, in absence of any external guidance, attention is directed to visually salient locations in the image. Gestalt psychology refers to theories of visual perception which attempt to describe how people tend to organize visual elements into groups or unified entities when certain principles are applied [95]. Main Gestalt grouping principles include similarity, continuation, closure and proximity. These principles describe visual coherencies from different aspects. Some work developed descriptors for each principle [96]. Visual Saliency or Saliency mainly deal with identifying fixation points that a human viewer would focus on at the first glance. Visual saliency usually refers to a property of a "point" in an image (scene), which makes it likely to be fixated. Most models for visual saliency detection are inspired by human visual system and tend to reproduce the dynamic modifications of cortical connectivity for scene perception.

- “Visual saliency is a property of objects or regions in which they stand out in a scene when viewed by the human visual system.” [97]
- “The salient map of an image highlights the pertinent objects while deleting the insignificant ones.” [98]

- “Visual saliency is the perceptual quality that makes an object, person, or pixel stand out relative to its neighbors and thus capture our attention” [99]

“Studies in psychology and cognition fields have found that, when looking at an image, our visual system would first quickly focus on one or several “interesting” regions of the image before further exploring the contents.” [100]

The goal of visual saliency is to match real human fixations of the scene (Fixation Points). A good saliency map should match, as more as possible, with human fixation points map of the scene. In the following paragraphs State of the art of visual saliency approaches is given. My method [93] for saliency maps is presented and compared with the best visual saliency methods of the state of the art.

4.2. State of the art

With the growing of digital multimedia data, salient maps have been increasing in interest in last few years, especially for CBIR (content based image retrieval). The reason why is that salient maps detect predominant objects in the scene.

In scientific literature Saliency approaches can be subdivided in three main groups:

Bottom-up, Top-down, Hybrid.

1. In **Bottom-up** approaches (stimulus driven) human attention is considered a cognitive process that selects most unusual aspects of an environment while ignoring more common aspects. In [101] the method is based on parallel extraction of various feature maps using center-surround differences. In [102] multiscale image features are combined into a single topographical saliency map. A dynamical neural network then selects attended locations in

order of decreasing saliency. Harel et al. in [103] proposed graph based activation maps.

2. In **Top-down** approaches [104,105] the visual attention process is considered task dependent, and the observer's goal in scene analysis is the reason why a point is fixed rather than others. Object and face detection are examples of high level tasks that guide the human visual system in top-down view.
3. Generally **Hybrid systems** for saliency use the combination of the two levels, bottom-up and top-down. In hybrid approaches [106,107] Top-down layer usually cleans the noisy map extracted from Bottom-up layer. In [106] top-down component is face detection. Chen et al. [107] used a combination of face and text detection and they found the optimal solutions through branch and bound technique.

A common problem for many of these models is that they often don't match real fixation maps of a scene. A newer kind of approach was proposed by Judd et al. [108] who built a database [109] of eye tracking data from 15 viewers. Low, middle and high-level features of this data have been used to learn a model of saliency. In our work we aimed to further study this problem. We decided to investigate about the relationship between real fixation points and computer generated distinctive points. The method, presented in this section, performs a new measure of visual saliency based on image low level features, particularly through the distribution of keypoints extracted by SIFT algorithm, as descriptor of texture variations into the image. In this work color feature is not considered. The method is totally unsupervised and it belongs to bottom-up saliency methods. Method effectiveness is measured by comparing resulting maps with real fixation maps of the reference database [109] and with two of the most important bottom-up approaches [102][103] and a hybrid method[108].

4.3. Proposed Saliency Measure

New measure of Visual Saliency is given in this method by focusing on low level image features such as texture. What's the matter for which texture information is used for detecting visual saliency? The answer is that texture gives important informations about image "behavior". The base for extracting salient regions, according to the method presented here, is to emphasize texture rare event. It is studied the spatial distribution of keypoints inside of an image to describe texture variations all over the image. The levels of roughness of both fine and coarse regions can be very different (in a fine region we will find a larger number of keypoints than in coarse regions), so keypoints density is used to find various texture events and to identify the most salient regions. In this work SIFT [60] algorithm is used to extract keypoints from an image. Then the concept of SIFT density maps (SDM) is presented, SDM is used to compute the final saliency map.

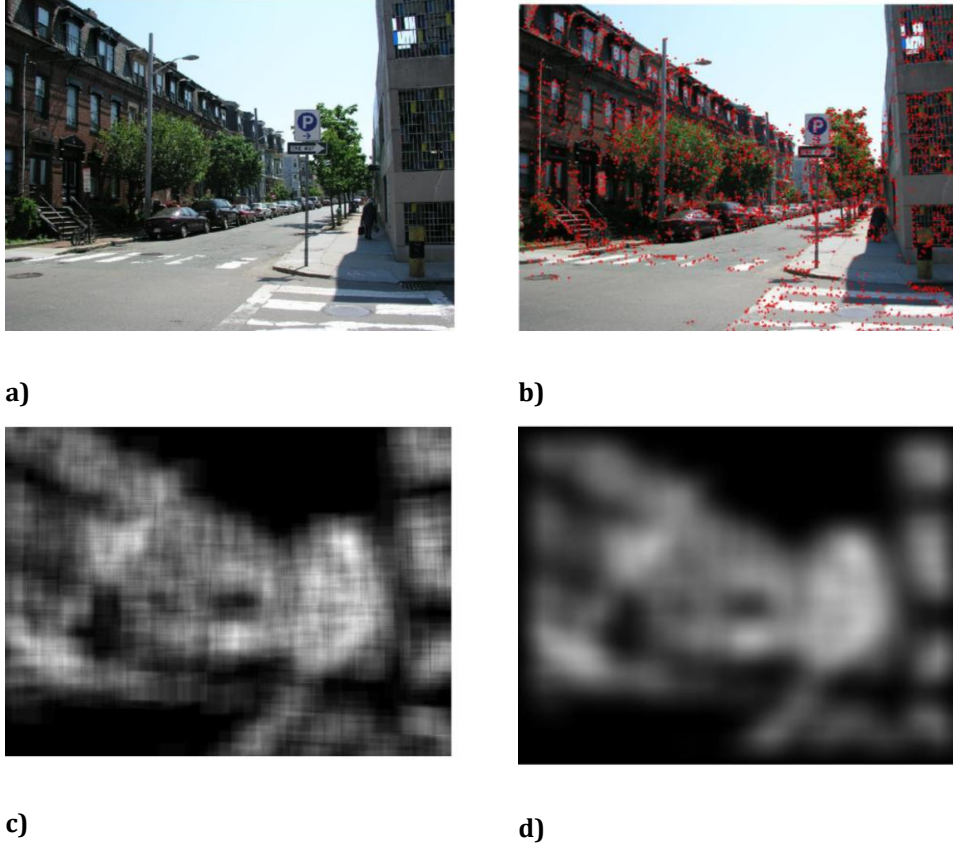


Fig. 42. Original Image (a), SIFT keypoints (b), SIFT Density Map ($k=64$) (c), final Saliency Map (d).

4.4. SIFT Density Maps

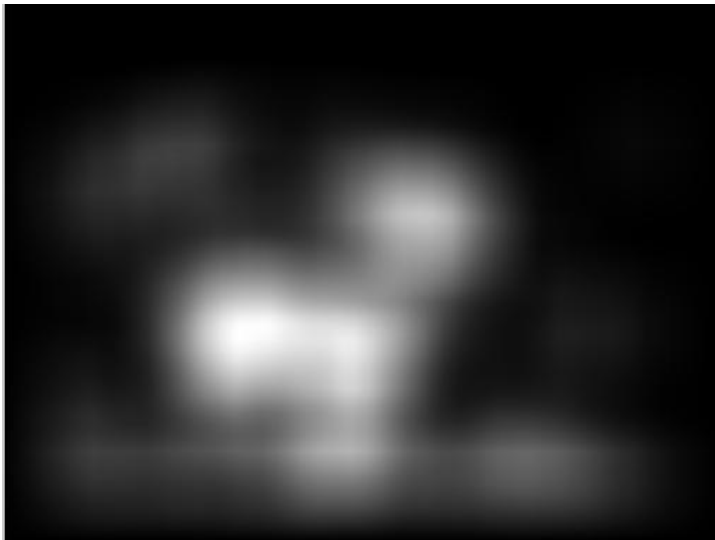
A SIFT Density Map (SDM) is a representation of the density of keypoints in an image, and can give essential information about the regularity of its texture. A SIFT Density Map $SDM(k)$ is built by counting the number of keypoints into a sliding window of size $k \times k$, which represent the scale of observation. Each point in the $SDM(k)$ indicates the number of keypoints into a squared area of size $k \times k$, centered in corresponding point of the image. It is evident that density values are strictly related to the value of k , and are limited by the window size. In fact smaller windows should be sensible to texture variations at a finer level, while larger

windows will emphasize coarser deviations. In section 4.5 the sensibility of the results with k will be discussed.

In real scenes, the simultaneous presence of many elements (the sky, the urban habitations, the urban green spaces) will show many kinds of texture. From a SIFT distribution point of view, the homogeneous surface of the sky has almost null values, the urban green spaces has mean density while urban habitations have high concentration of keypoints. (fig. 42)



a)

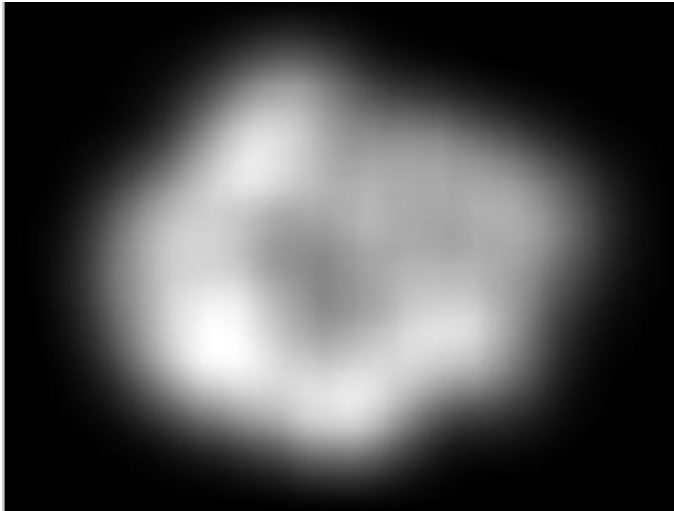


b)

Fig. 43. Two image examples: a homogeneous subject in a textured scene (a) and the corresponding Saliency Map (b);



a)



b)

Fig. 44.; Two images example :a textured object in a homogeneous background (a) and the corresponding Saliency Map (b)

4.5. Saliency Map

The saliency map SM, performed in my method [93], for a given k , is built as the absolute difference between the SDM values and the most frequent value MV of the map:

$$SM(k) = |SDM(k) - MV(SDM(k))| \quad (61)$$

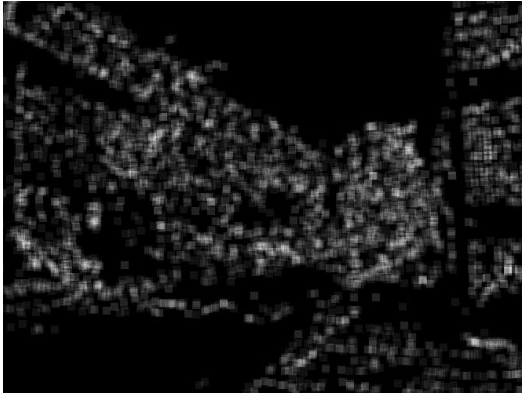
which is further normalized with respect to the maximum value to restrict SM values to $[0,1]$.

The most salient areas into the image are those related to the SDM values with the maximum deviation from the most frequent value, typically the most rare texture events in the image. This measure emphasizes both the case in which a textured object is the salient region, as it is surrounded by homogeneous areas (the most frequent value near to 0), and the case in which a homogeneous area is surrounded by textured parts (a higher most frequent value). (figures 43, 44)

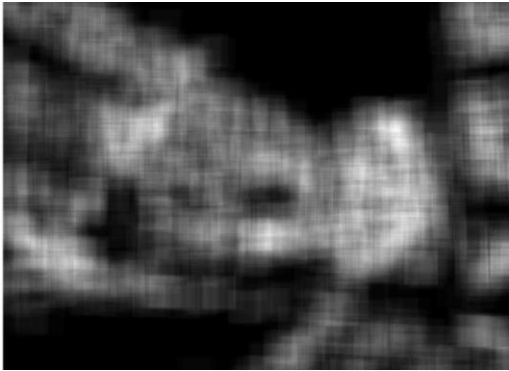
In addition, for a smoother representation of the saliency map, we apply to the SM an average filter which has a window size that is a half of that used to build the map (k).



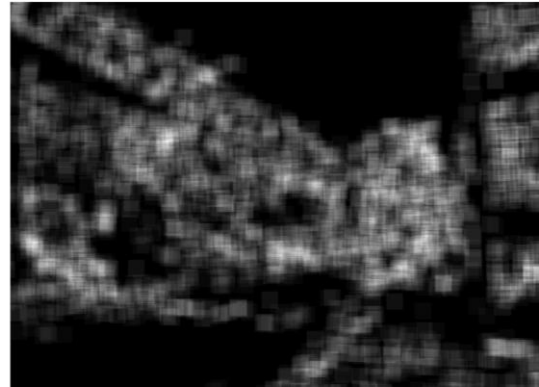
a)



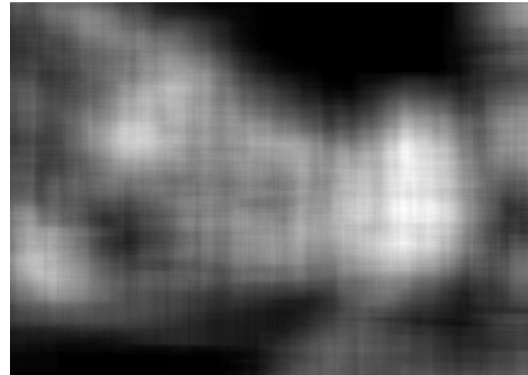
b)



d)

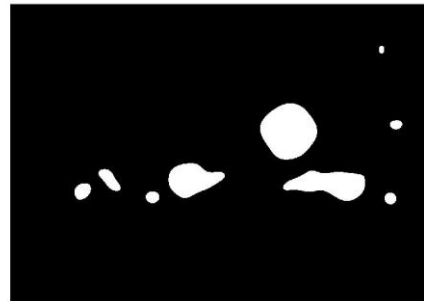
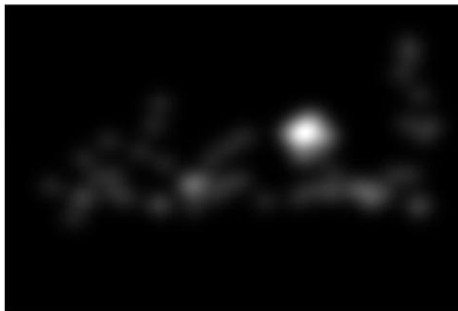


c)



e)

Fig. 45 Original Image (a), SIFT Density Maps with different values of k (16,32,64,128).



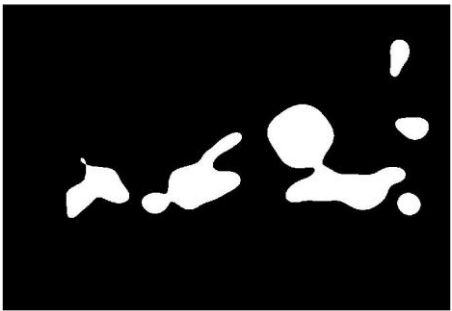

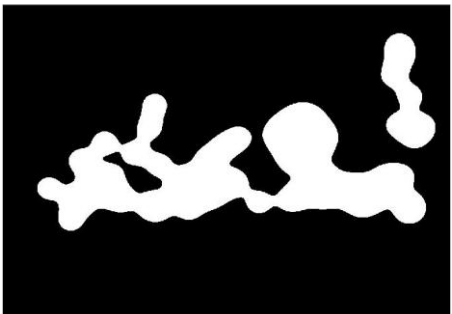
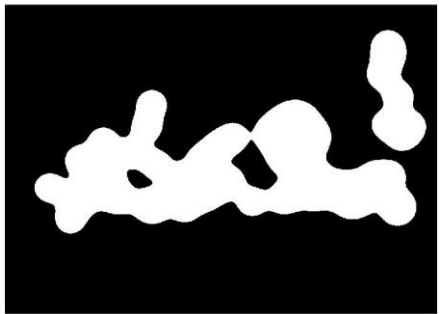
a)	b)
	
c)	d)
	
e)	f)

Fig. 46 Fixation Map (a) of the image in fig. 45.(a), Binary maps with different thresholds (0.95, 0.9, 0.85, 0.8, 0.75)

4.6. Experimental results

In this section results of the method I performed, are compared with those of Itti-Koch[102], Harel's Graph Based Visual Saliency (GBVS) [103] and Judd [108] methods. Tests were made on [109] dataset which consists of 1003 images and the corresponding maps of fixation points, which are taken as reference groundtruth (in the tests all the images have been resized down by a factor of two). Tests were executed on an Intel Core i7 PC (4 CPU, 1.6 GHz per processor, 4 GB RAM), and the method has been implemented in Matlab. Koch's Saliency Toolbox[110] is used to compute saliency maps for the methods [102] and [103], and the maps given in [109] for the Judd's method. Tests were repeated for different values of window size (16, 32, 64, 128 - fig.45), with the aim to study the sensibility of the results to this parameter. To compare the results with the other methods, and to the groundtruth, the less N% salient pixels (with N= 95, 90, 85, 80, 75 - fig.46) are discarded from the saliency, to create a set of binary maps. Then performances are measured by using as metric (SP), a combination of precision and recall parameters:

$$R = \frac{n(M_D \cap M_R)}{n(M_R)}; P = \frac{n(M_D \cap M_R)}{n(M_D)} \quad (62)$$

$$SP = R \cdot P$$

where M_D is the binary version of the detected saliency map (with the method presented, or the others), while M_R is the binary version of the reference fixation map.

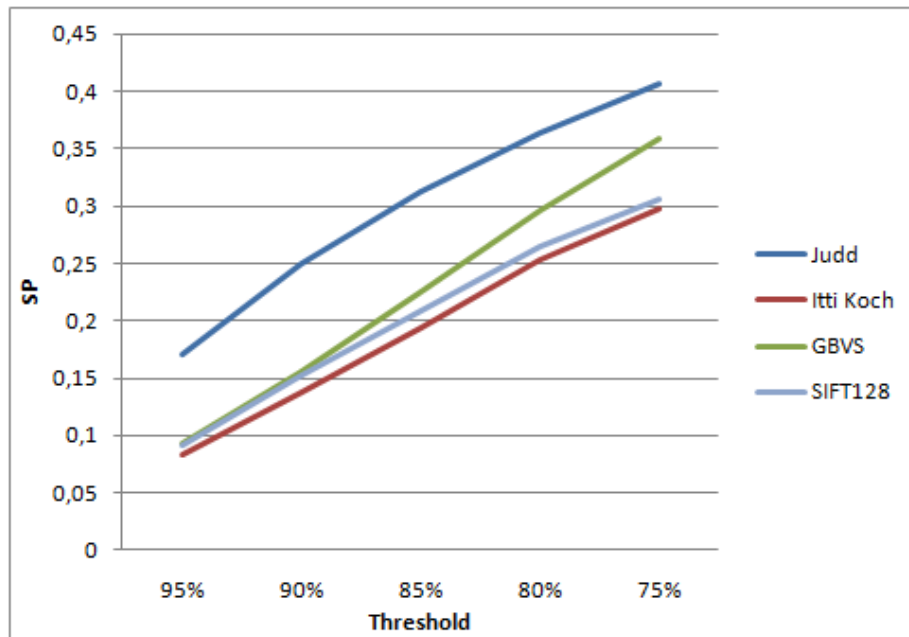
R is the recall, i.e. the ratio between the number of pixels in the intersection between the detected map M_D and the reference map M_R , and the number of pixels in M_R . When it tends to 1, M_D covers the whole M_R , but no information is given about pixels outside M_R (a map made of only salient pixels gives $R=1$ if compared with any other map). If it tends to 0 detected and reference map have smaller intersection.

P is the precision, i.e the ratio of the number of pixels in the intersection between M_D and M_R , and the number of pixels in M_D . When P tends to 0, the whole M_D has no intersection with M_R . If it tends to 1, fewer pixels of M_D are labeled outside M_R . Nevertheless this parameter will not assure that the whole reference area has been covered.

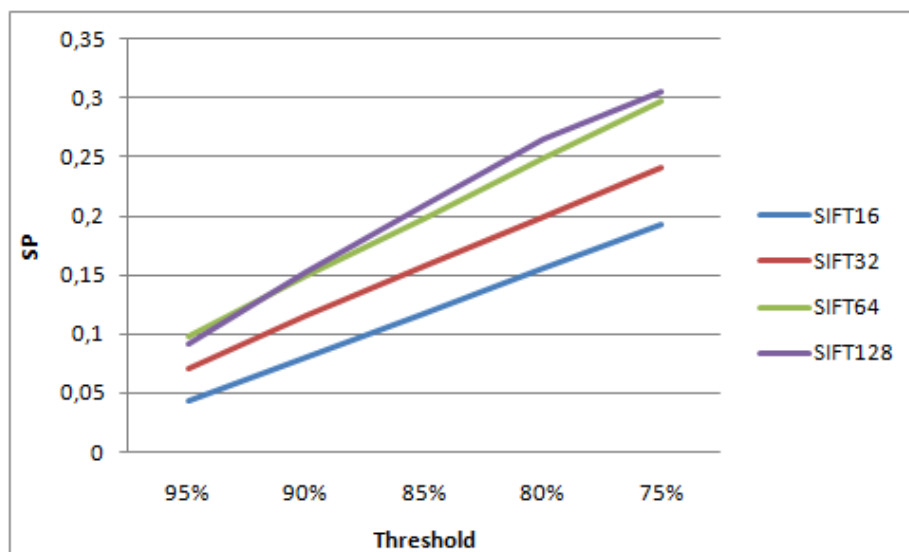
Fig. 47 shows average precision results versus different values of thresholds. Note that my method gives its best results for $k=128$. As noted in section 4.5, smaller windows can capture finer details, while larger windows emphasize coarse variation of texture. In terms of saliency, human attention is more attracted by areas in which there are large texture variations, rather than by small deviation. Then a larger window size is preferable. Note also that results with 64 and 128 are similar. In fact it is observed that while the recall value increases with the window size, precision, in case of very large window, does not increase as well.

In the comparison with the other methods, first some fundamental issues must be underlined:

- Judd method is supervised, and uses 9/10 of the whole dataset for training and 1/10 for testing. Judd results are averaged only on the 100 testing images. It uses both color and texture information.



a)



b)

Fig. 47. Average Saliency Precision (SP) vs. Threshold for my method with different window sizes (a), and for my method (k=128) compared to the other methods (b).

- Itti-Koch and GBVS method are unsupervised method and use both color and texture information.

- My method is unsupervised and use only texture information to build the saliency map.

My saliency map gives better results than Itti-Koch, for all the threshold values, even if we use only texture information. Results are similar to GBVS for higher threshold values (0.95 and 0.9), which give information about the most salient pixels, while my precision does not increase as well for lower values of threshold (0.85, 0.8). As expected Judd method achieves best results, as it is a supervised method, while all the other methods are unsupervised. Furthermore Judd tests refers only within a small selected subset of images (100 testing images), while other methods have been tested within the whole dataset. Judd results are reported only as asymptotic values to be compared with. Figures. 48, 49, 50 show some examples of saliency maps with all the discussed methods. Regarding temporal efficiency, my method takes less than 10s to build a saliency map, and it is comparable with Itti-Koch and GBVS method for medium images (300 x 600). Most of the time (70% ca) is spent to extract keypoints, but it depends on image complexity, i.e. the number of keypoints extracted.

4.7. Conclusions

Visual saliency has been investigated for many years but it is still an open problem, especially if the aim is to investigate the relationship between synthetic maps and points, in a real scene, that attract a viewer attention.

The purpose of the method presented in this chapter was to study how computer generated keypoints are related to real fixation points. No color information has been used to build my saliency maps, as keypoints are typically related only to image texture property.

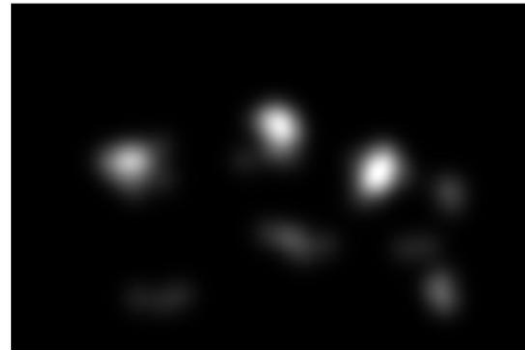
Even if only texture information is used, experimental results show that my method is very competitive with respect of two of the most cited low-level approaches.

Judd's method achieves better results as it is a supervised method which has been trained with the fixation maps within the selected dataset.

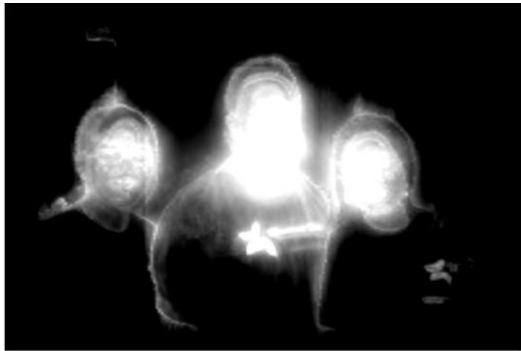
In my future works i want to study new color based saliency techniques to be integrated with my proposed approach, to improve experimental results.



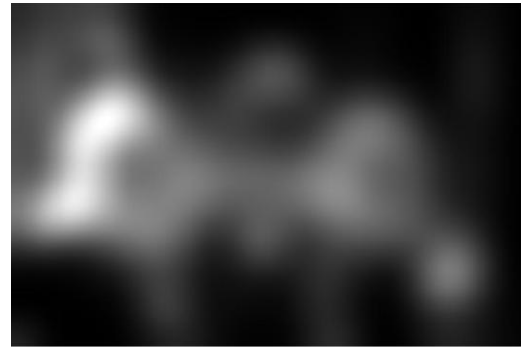
a)



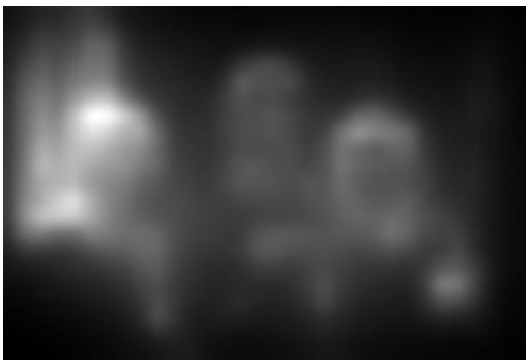
b)



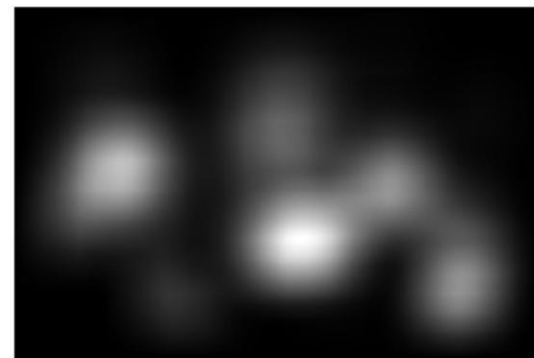
c)



d)



e)



f)

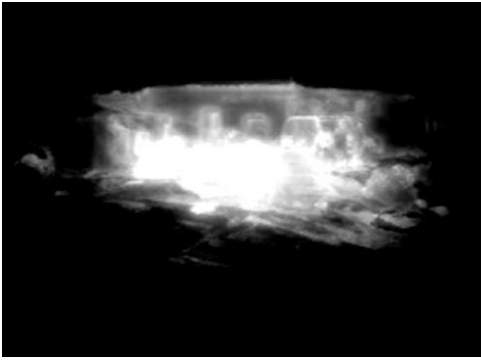
Fig. 48. Some visual results. Original images (a), fixation maps (b), Judd maps (c), Itti-Koch maps (d), GBVS maps (e), my method (f) window size 128.



g)



h)



i)



j)



k)

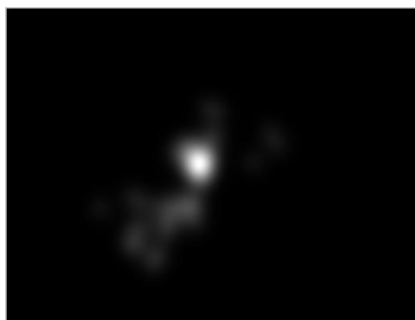


l)

Fig. 49. Some visual results. Original images (g), fixation maps (h), Judd maps (i), Itti-Koch maps (j), GBVS maps (k), my method (l) window size 128.



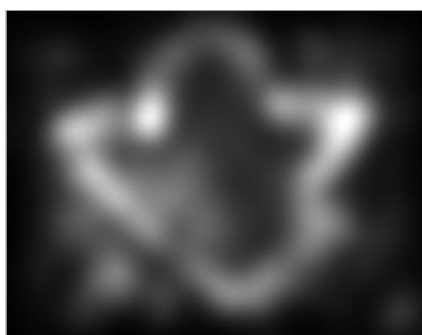
m)



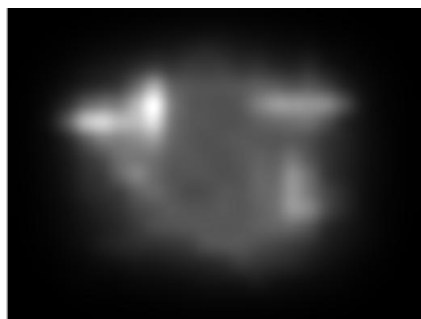
n)



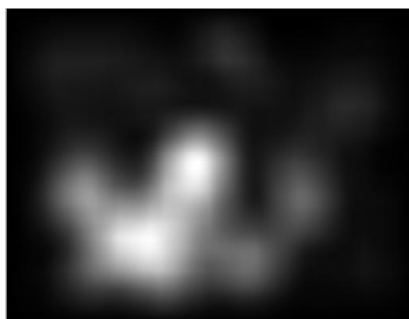
o)



p)



q)



r)

Fig. 50. Some visual results. Original images (m), fixation maps (n), Judd maps (o), Itti-Koch maps (p), GBVS maps (q), my method (r) window size 128.

5. IMAGE FORENSICS

5.1. Introduction

The wide availability of digital cameras and imaging tools increased digital image manipulation phenomenon. The manipulation of digital images is made simple by low-cost hardware and software tools that are easily and widely available. For this reason, today, it is difficult to establish if a picture is totally real, "almost" real, or far away from the real description of a scene. We can find examples of tampered images in several fields: tabloid magazines, political campaigns, medical imaging, digital forensics. In the analogical era, a photo was always considered as a proof of occurrence of a real scene. Nowadays image manipulations mine the credibility of a digital document (for example, a body of evidence can be hidden in an image presented in a court of law). Although digital watermarking techniques have been proposed to provide authenticity to images, the majority of images captured do not contain a digital watermark. Digital Image Forensics[111] deals with the problem of certifying the authenticity of a picture, or its origin, without explicit a priori information, e.g. using watermarks. This research field, particularly, deals with:

1. image source identification – it determines the source of digital image (such as digital-camera or scanner) through data acquisition.
2. discrimination of synthetic images from real images; it identifies computer generated images

3. image forgery detection. It determines when a digital image has been tampered.

This chapter is focused on third branch of image forensics: forgery detection. This is a primary goal for image forensics, an image can be tampered in many ways at different degrees. In next paragraph an overview of the most important kind of image forgeris.

5.2. Image Tampering

Many of image tampering techniques [112] generate image with no visual traces or artifacts. The most used techniques for image tampering are:

- **Copy-move:** It is typically used to add or hide some objects in a scene. Parts of the image are duplicated elsewhere into the same image, often after being modified by geometrical transformations.
- **Image-splicing:** A common form of photographic manipulation is the digital splicing of two or more images into a single composite. When performed carefully, the border between the spliced regions can be visually imperceptible.
- **Resampling:** To create a convincing composite, it is often necessary to resize, rotate, or stretch portions of an image. For example, when creating a composite of two people, one person may have to be resized to match the relative heights. This process requires resampling the original image onto a new sampling lattice, introducing specific periodic correlations between neighboring pixels.
- **Re-touching:** It is a kind of image manipulation for photo restoration or enhancement. Techniques as adjusting colors, contrast equalization, white balance, sharpness, removing elements or visible flaws on skin or materials.

An instance of re-touching is the manipulation for fashion, it is the removal of visible flaws on top-model skin. This is used for commercial use.

A good overview of the most used image manipulation is given Farid Image forensics survey [2].

5.3. Copy-Move forgeries

One of the most studied image manipulations in literature is the copy-move process of parts of an image, to hide or add details in a real scene. Parts of the image are duplicated elsewhere into the same image, often after being modified by geometrical transformations (such as translation, rotation, change in scales). In fig. 51 it is shown a famous example of copy-move forgery, in which a portion of an image is copied and pasted to hide information. In fig. 52 an object of the scene is copied and pasted in another part of the image. As showed, a copy move tampering done with an high grade of accuracy is not often visible to a first look.

Different approaches have been performed in state of the art of copy move forgery detection techniques. I focused my attention on detecting copy-move image tampering, to estimate the reliability of an image. In next paragraphs of this chapters the following sections are included: state of the art of copy-move detection approaches; my methods for copy-move detection; experimental results.



Fig.51 A famous example of a copy move tampering, George Bush was removed by simple copying and pasting of several soldiers



Fig. 52 An example of copy move tampering: a new soldier is copied and pasted in the first line of chinese terracotta soldiers

5.3.1 State of the art

Copy-move tampering is often retouched with geometrical and illumination adjustments, so the portion of pixel copied has the same, or similar, features of the rest of the image. For this reasons it is very difficult for a human observer to detect the forgery. We can divide the most part of copy move detection methods in literature into two main groups:

- **block-matching based methods;**
- **local features matching methods.**

In the first family of techniques an image is divided into overlapping blocks, then some features are extracted from each block, and compared each other to find the most similar ones. Results are analyzed and decision is made only if there are several pairs of similar image blocks within the same distance. Several authors proposed different features to represent image blocks: Discrete Cosine Transform (DCT) [113], Principal Component Analysis (PCA) [114], Discrete Wavelet Transform (DWT) and Singular Value Decomposition (SVD) [115], color information [116]. In [113], the advantage of using DCT is that the signal energy is concentrated on the first few coefficients, and operations such as noise addition, compression, and retouching should not affect these coefficients. Nevertheless, it does not work if duplicated regions are processed by geometrical transformation. In [114] eigenvectors of the covariance matrix between blocks are used to represent them. This representation proved to be robust to compression and noise addition, however re-sampling (scaling, rotation) affects eigenvalues. The results of [115] and [113] are very similar because SVD is on parallel with PCA technique. Luo et al. in [116] used color information to represent blocks: average value of red, blue and green color components of the whole block and of 4 sub-blocks, divided according to different directions. Experimental results showed that this method was very robust

to JPEG compression, Gaussian blurring and additive noise. These approaches are, on the whole, robust to noise addition, compression, retouching, but lack of robustness against geometrical attacks (rotation, scaling, distortion).

The second family of techniques is based on local features matching: several methods used image interest points matching to identify duplicated regions. Huang et al. presented a method which uses SIFT descriptors [117] to detect copy move forgeries in an image, by matching keypoints. In [118] I performed a method that matches clusters of SIFT keypoints and that proved to be robust to scaling and rotation. In [119] Amerini et al. used SIFT keypoints matching to estimate the parameters of the affine transform and to recover matched areas. These last two methods are the first works in literature which tried to study the structure of the copies, instead of analyzing only low-level features. For a complete overview about region duplication detection see [120].

I performed three image forensics techniques for copy-move detection. In the following paragraphs, for each of methods a further description is given.

5.4. Block matching approach

In this paragraph my block matching approach for copy-move detection is presented. In this method texture features are exploited to be extracted from blocks to be matched. The goal is study if texture is suited for the specific application, and to compare performance of several texture descriptors.

In fig.53 the scheme of overall method is shown.

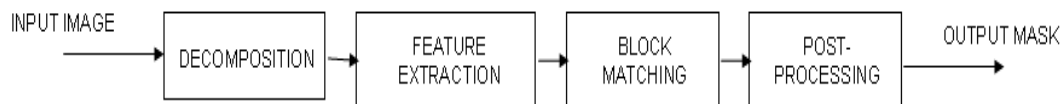


Fig. 53 scheme of overall block matching method

Input image (size $M \times N$) is first grayscaled, as in this method I am interested only in studying the influence of texture properties. Image is then decomposed into

overlapping square blocks (size B x B). The number of blocks to be analyzed is $nB = (M-B+1) \times (N-B+1)$. Texture features are extracted from each block: 5 different texture descriptors are tested (see paragraph 5.4.1). Each feature is represented as a vector.

Blocks are sorted according to vector's component which has the maximum variance along all the blocks. The sorted list of the blocks is then scanned in order to find similar blocks. Features from each block are compared to those of the next blocks in the sorted list, according two possible approaches:

- Fixed Window: $WS = nB \cdot P$ where P is a fixed parameter, that indicates the percentage of following blocks in the list to be considered.
- Adaptive window: blocks are compared up to those in the list which distance (for the key component) is below a threshold.

Even if the second approach is adaptive, the first one is faster, since matching time for matching is constant for all the blocks. Furthermore no significant differences in accuracy has been measured. Therefore it is preferred the fixed window solution setting the value of WS as 1/1000 of the number of blocks.

As similarity criterion the relative error is used:

$$\left| \frac{V_i^j - V_i^k}{V_i^j} \right| < TH \quad \forall i = 1..n \quad (63)$$

between the corresponding components V_i^j and V_i^k of the feature vectors (size n) extracted from two blocks B_j and B_k . To make it symmetrical, instead of using (63), the relative error is computed as the ratio of the absolute error and the average value of the two components:

$$\left| \frac{V_i^j - V_i^k}{\frac{1}{2}(V_i^j + V_i^k)} \right| < TH \quad \forall i = 1..n \quad (64)$$

If all the relative errors are below a percentage TH , the two blocks are considered as candidate forgeries. The relative error of each component is chosen, rather than Euclidean distance of the vectors, as in some cases, components have different order of magnitudes.

Then the distance d_{jk} between spatial coordinates of matching blocks is computed. Matching blocks that overlaps are considered false positives and discarded. To further reduce false positives, the method selected only pairs of blocks which are at the most frequent distance (MFD processing). In fact in a tampered image a copy is the translated version of a set of blocks from elsewhere into the image, and the same translation function is applied to all the copied blocks. Furthermore, in case of forgeries, duplicated areas are composed by several matching blocks. Then, among candidates, isolated blocks, that are not connected to any other blocks, are deleted from the output mask (IB processing). This general approach has been applied extracting 5 different texture descriptors.

5.4.1 Texture Descriptors

Texture is one of the most studied image features in Computer Vision, Image Processing and Computer Graphics applications. For our purpose we test 5 different standard texture descriptors (see chapter 2):

- Statistical: mean, standard deviation, skewness and kurtosis of the pixels grey values. Output is a 4-dimensional vector.

- Edge Histogram: in our simplified version, we filter blocks with 4 directional (vertical, horizontal, 45, 135) and a non-directional Roberts-like operators. Mean of the filtered blocks are considered as descriptors. Descriptors is a 5-dimensional feature vector.
- Tamura descriptors: Contrast, Coarseness and Directionality properties from the Tamura set of features. Output a 3-dimensional feature vector.
- Gabor descriptors: a bank of Gabor filters (2 scale and 4 orientations) is applied to blocks. Mean and standard deviation of the coefficients from each sub-band are calculated to form a 16-dimensional texture feature.
- Haralick Descriptors: The Haralick descriptors are based on statistical moments and obtained from co-occurrence matrix. We use as descriptors correlation, energy, contrast, homogeneity, resulting in 4-dimensional vector.

5.4.2 Evaluation Metrics

To evaluate the precision of the method, the source and the destination areas of every copy moves are saved as a binary mask: this area is called A_R (reference area, fig. 54.c). The result of the detection method is an output mask A_D (detected area, fig. 54.d-i)

Better results as much A_D is similar to A_R . In particular we measured detection precision (DP) as follows:

$$R = \frac{n(A_D \cap A_R)}{n(A_R)} \quad (65)$$

$$P = 1 - \frac{n(A_D \cap \bar{A}_R)}{n(A_D)} \quad (66)$$

$$DP = R \cdot P \quad (67)$$

where:

- R is the recall, i.e. the ratio between the number of pixels in the intersection of the detected area A_D and the reference area A_R , and the number of pixels in A_R . When it tends to 1, A_D covers the whole A_R , but nothing can be said about pixels outside A_R ; if it tends to 0 A_D and A_R have smaller intersection; P is the precision, i.e. one minus the ratio between the number of pixels in A_D , which are not in A_R , and the number of pixels in A_D . When P tends to 0, the whole detected area has no intersection with the reference. If it tends to 1, fewer pixels of A_D are labeled outside A_R . Nevertheless this parameter will not assure that the whole reference area has been covered.

DP combines these two parameters: DP is high if A_D both covers A_R and has few outliers, and it is low if A_D and A_R are only partially overlapped or, when A_R is well covered, if A_D encloses many pixels which are not in A_R .

Varying block size two overlapping effects are observed:

-If the block size is larger than the tampered areas, $R \rightarrow 0$. In fact, whatever block we consider it will contain pixels which do not belong to the tampered area. Therefore no matches can be found.

-If block size is small $P \rightarrow 0$, because the probability to have natural similarities in an image increases, so that the number of false positives.

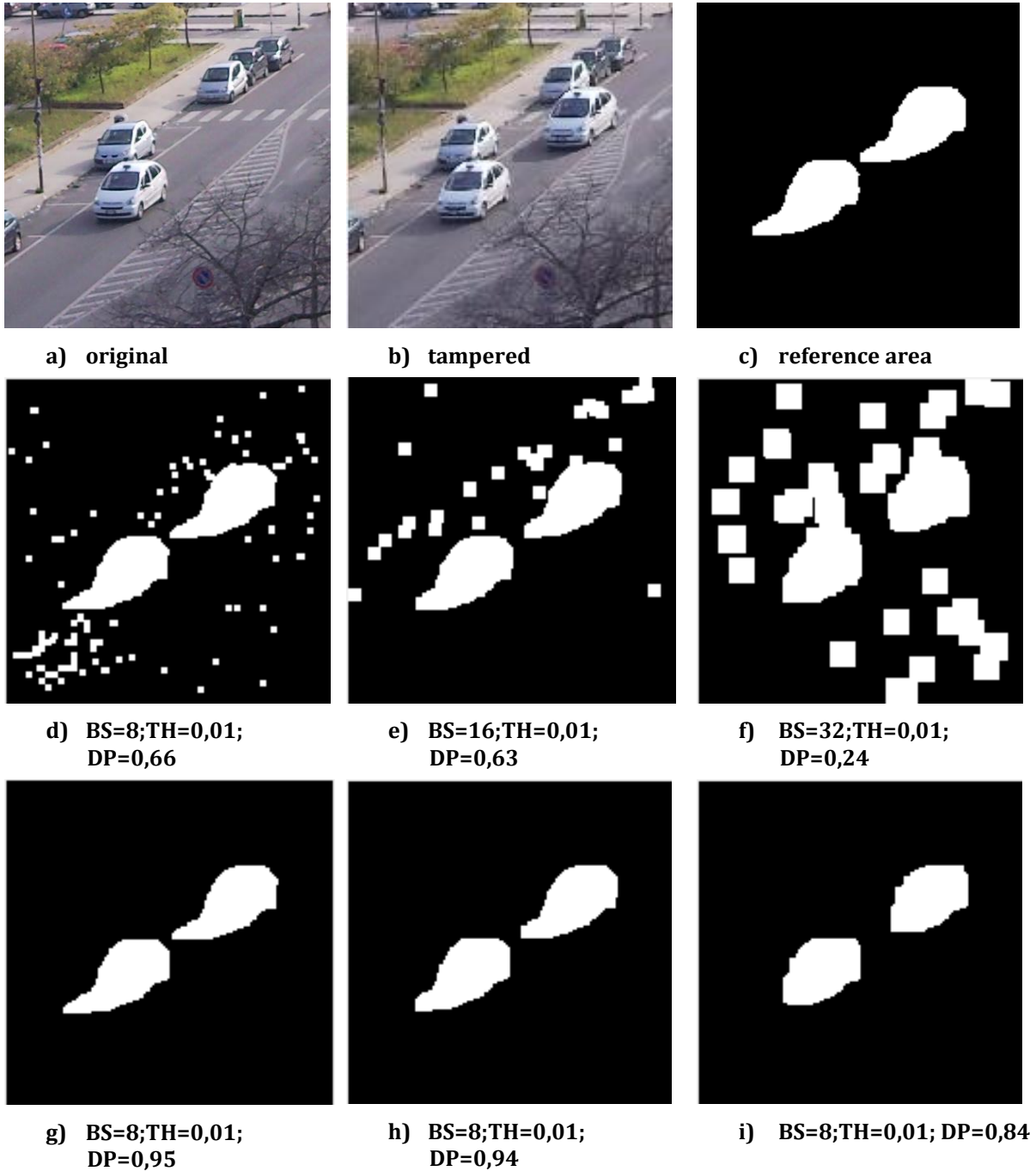


Fig. 54 Example results: Statistical Descriptor. After matching (d-f) and after post-processing (MFD and IB processing) (g-i). BS=block size, TH=threshold, DP=Detection Precision

5.4.3 Experimental Results

The overall method has been implemented in Matlab and executed on an Intel Core i7 PC (4 CPU, 1.6 GHz per processor, 4 GB RAM). We exploit the Matlab parallel library to make 4 workers run simultaneously. The dataset is made of 20 400x400 tampered uncompressed images, into which areas are copied and pasted onto other parts of the same image. For each test both accuracy and execution time are measured. Tests are also repeated for JPEG-compressed versions of the images (70% quality) to study how compression influences the detection ability of the analyzed descriptors.

Fig.54 shows some results obtained with the Statistical descriptor, varying block size and with a fixed threshold value. Note that after matching (54.d-f), DP decreases when block size increases, since larger false positives are detected, and the method does not correctly detect contours of the copies, in correspondence of fine details of the reference area. After post-processing (54.g-i), results with block size 8 and 16 are very similar, while 32 is a too larger value for this input image, and object contour is roughly detected.

- Figure 55 shows average execution time for two of the steps of the method: feature extraction and matching (post-processing time is negligible with respect of matching). with block size.

- Edge Histogram and Tamura descriptors are quite fast for small blocks while time strongly increases with block size.

- Gabor and Haralick feature extraction time does not depend on block size.

Regarding matching time, results (fig.55) versus block size (results averaged on thresholds) and versus threshold value are shown (averaged on block sizes):

- It loosely depends on block size (it slightly decrease as the number of blocks decrease)
- Processing time increases when threshold value is higher,as the number of matches increases, particularly for Tamura and Gabor.

Figures 56,57,58,59,60 report results about average detection precision results achieved with the 5 tested descriptors, for uncompressed (left) and JPEG-compressed(right) images.

Results after matching (purple, light blue lines and red in figures 56-60), before post processing, show that:

- Statistical, Edge Histogram and Gabor descriptors give similar results: precision is very high (90%) for lower values of the threshold Th while it decreases when for higher values of Th , as some false positives are detected.
- Tamura and Haralick give bad results (acceptable results only for Tamura with small block size)

After post-processing (dark blue, yellow and violet):

- For small blocks all the methods give similar results, very high precision (even about 95%). Only Haralick gives a lower average precision (85%).
- Statistical, Haralick and Edge Histogram precisions are practically independent of threshold value, within the tested range of values.
- Gabor and Tamura precisions decrease for higher threshold values, as some false positives are still detected.

For JPEG-compressed images (quality 70%):

- After matching, before post-processing, no good results with any of the descriptors, within the selected range of thresholds.

- After post-processing, bad results are shown with lower values of the threshold, and some results are achieved only with 0,1. Acceptable results with Statistical (80%) and Edge Histogram (70% with block size 16 and 32).
- Higher values of the threshold do not improve precision, as the number of false positives strongly increases.
- For higher compression ratio, (e.g. 50%) the method does not give acceptable results, not even using best descriptors.

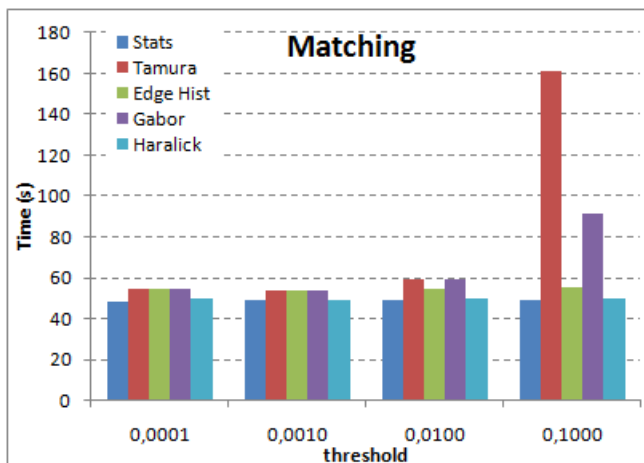
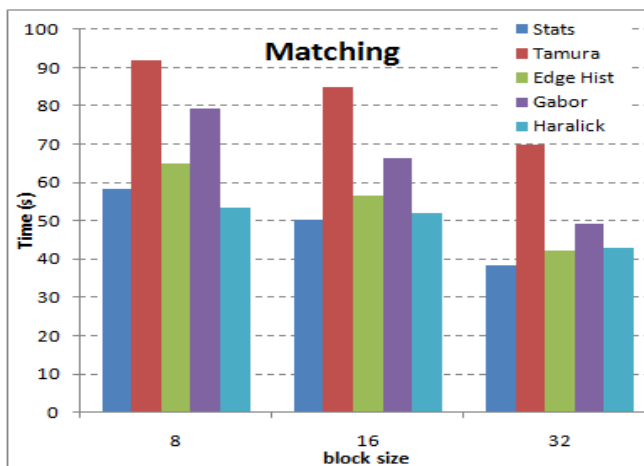
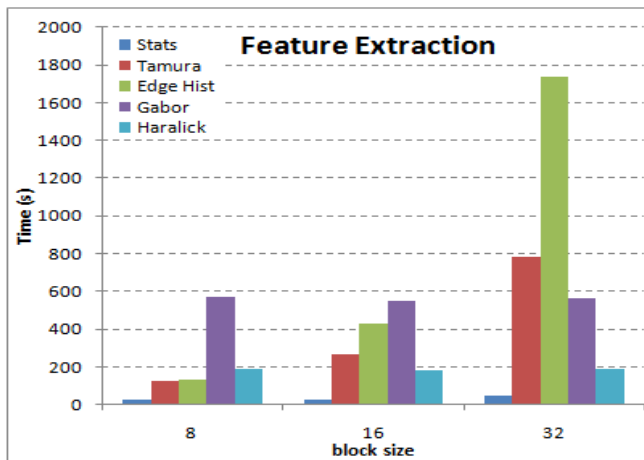


Fig.55 Average Execution Time: Feature Extraction vs block size (left); Matching vs block size (center) and vs threshold (right).

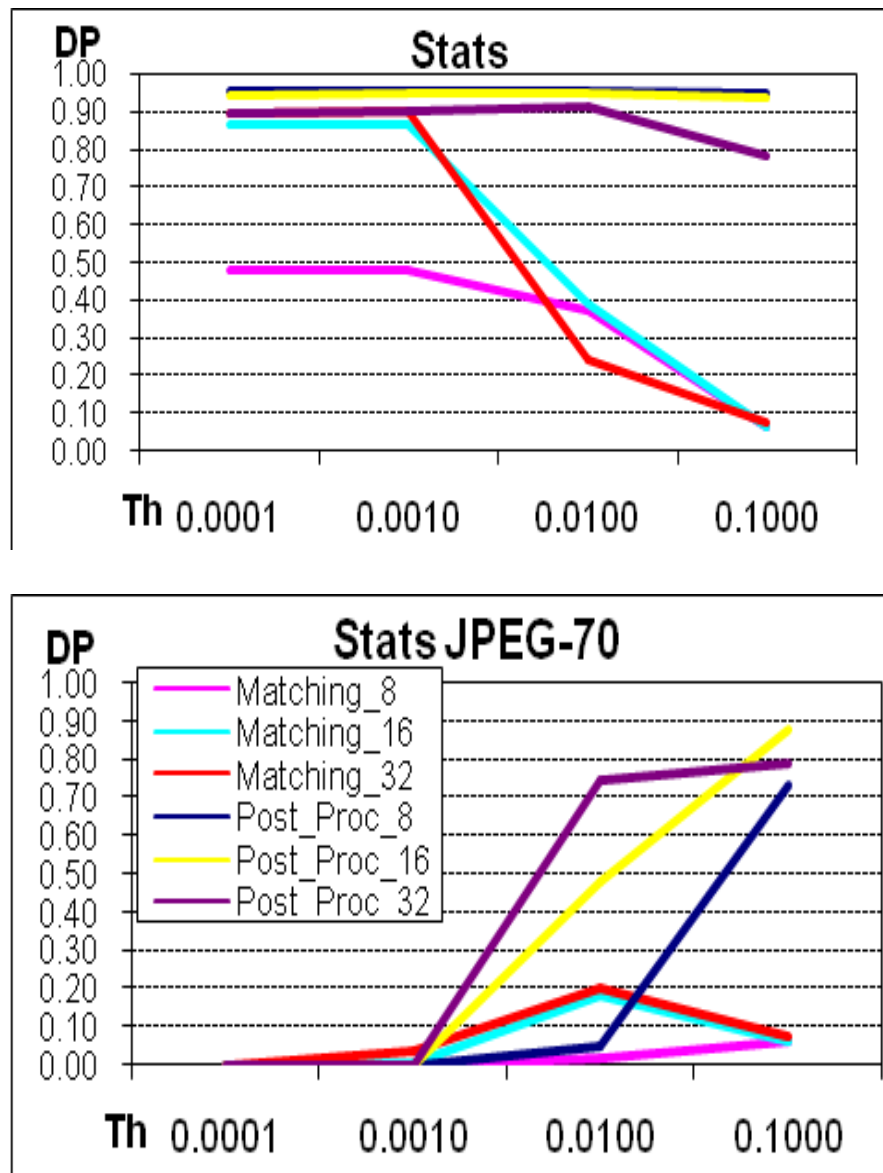


Fig. 56 Average Detection Precision (DP) vs Threshold (Th). Lines represents results measured after matching and after post-processing with different values of block sizes (8,16,32). (for clarity, table callouts are shown only on the right).

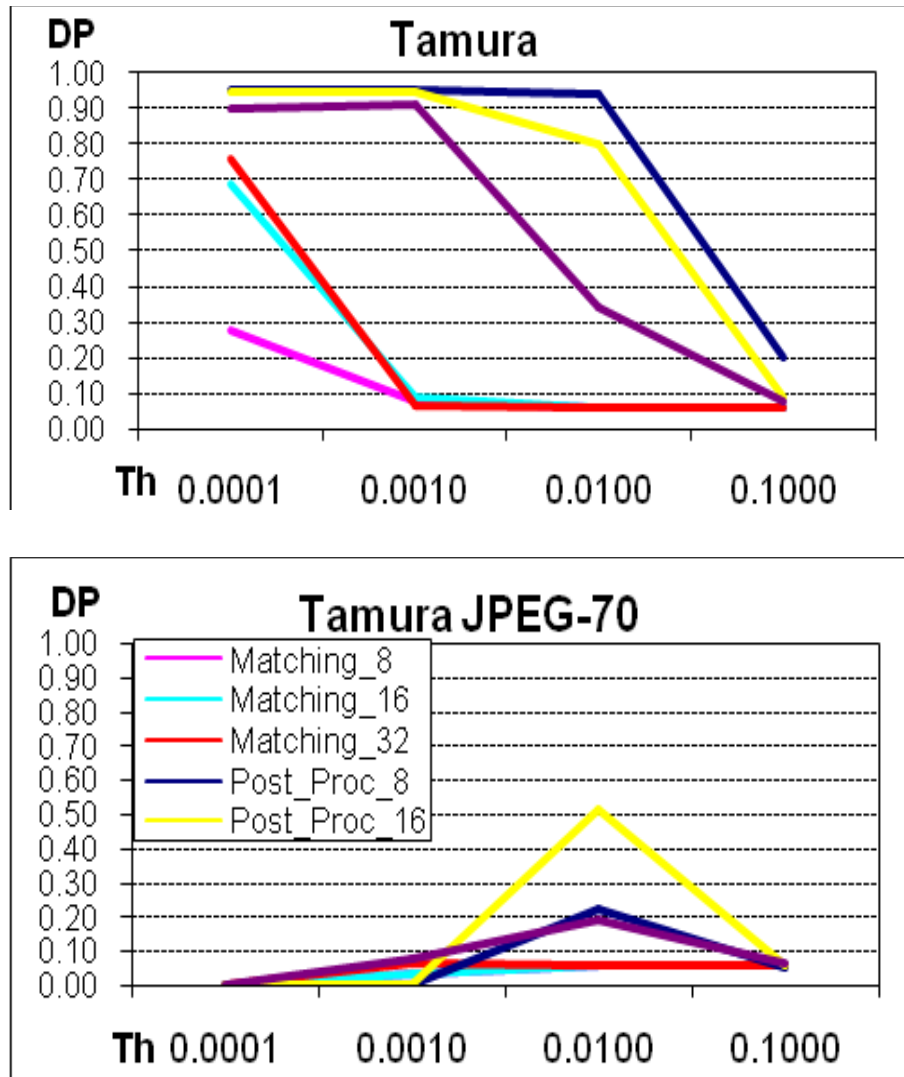


Fig. 57 Average Detection Precision (DP) vs Threshold (Th). Lines represents results measured after matching and after post-processing with different values of block sizes (8,16,32). (for clarity, table callouts are shown only on the right).

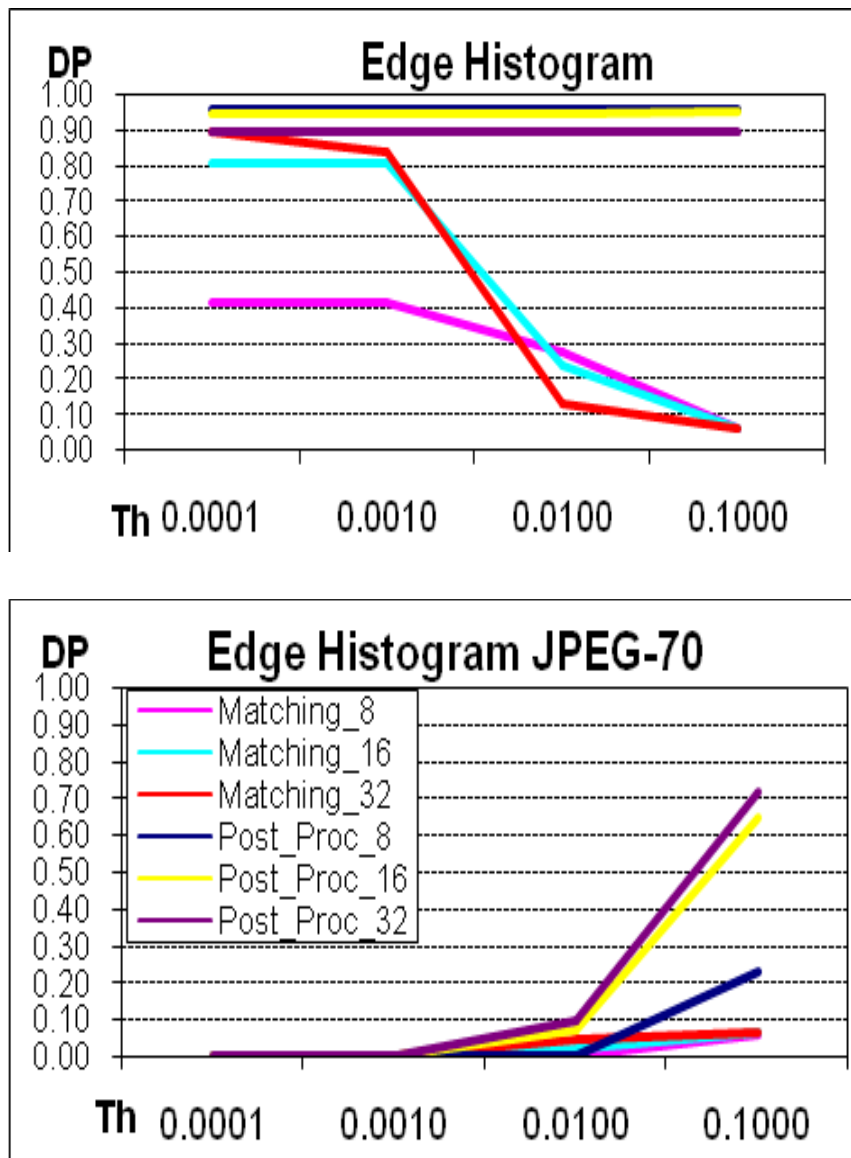


Fig. 58 Average Detection Precision (DP) vs Threshold (Th). Lines represents results measured after matching and after post-processing with different values of block sizes (8,16,32). (for clarity, table callouts are shown only on the right).

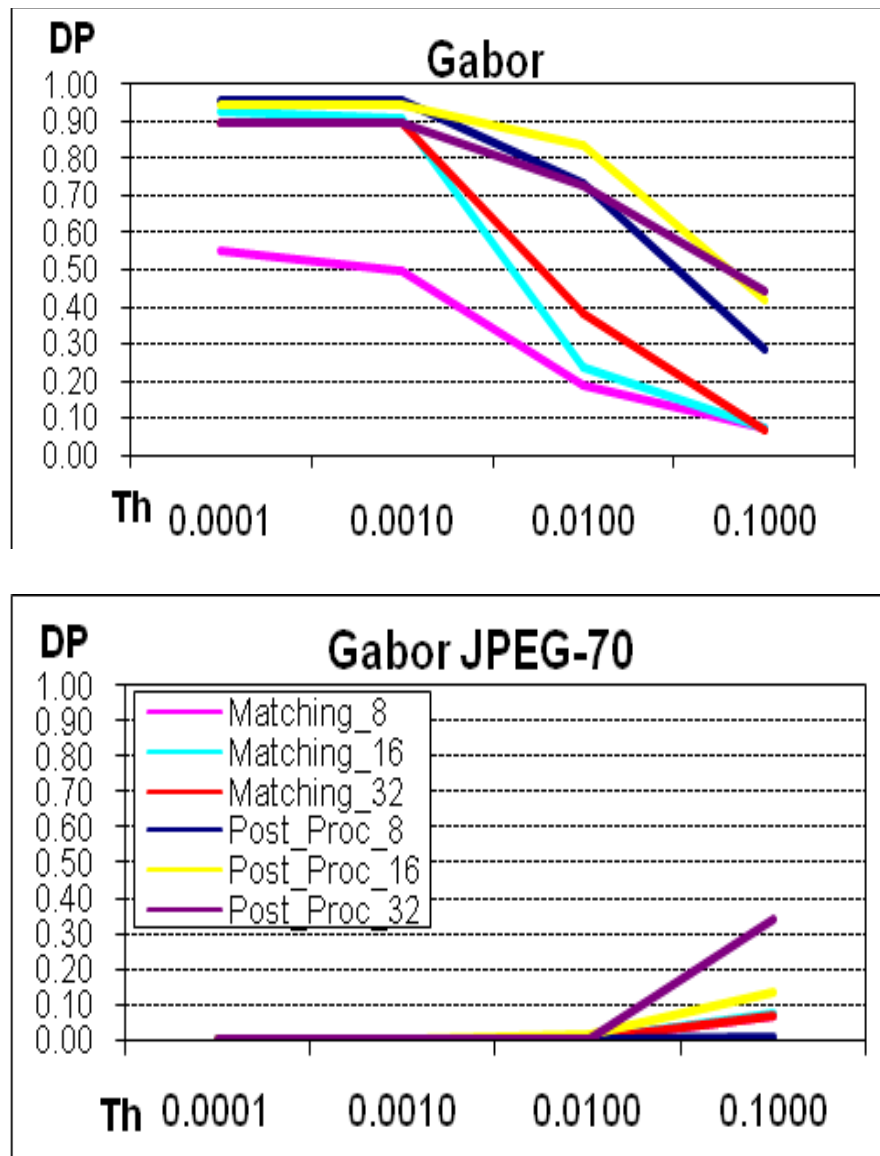


Fig. 59 Average Detection Precision (DP) vs Threshold (Th). Lines represents results measured after matching and after post-processing with different values of block sizes (8,16,32). (for clarity, table callouts are shown only on theright).

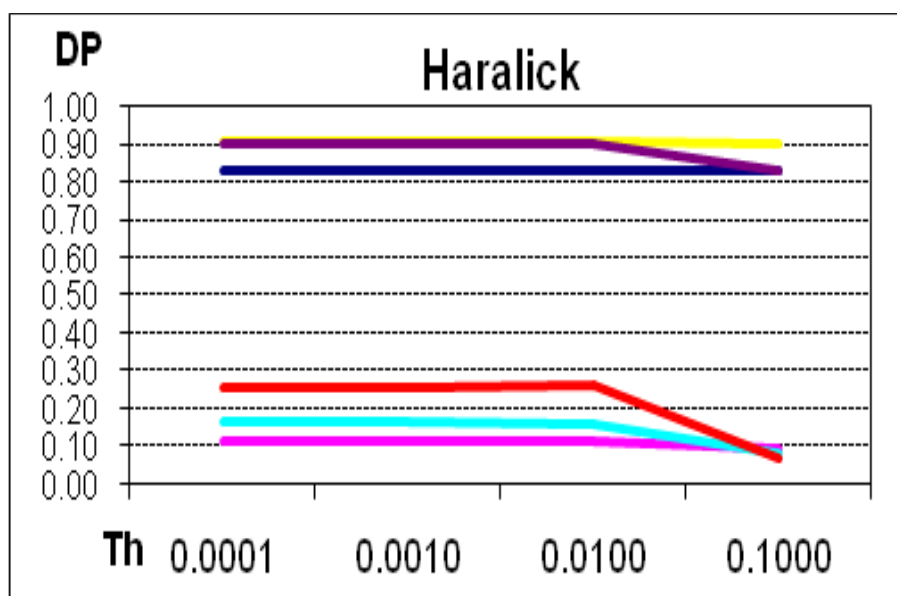
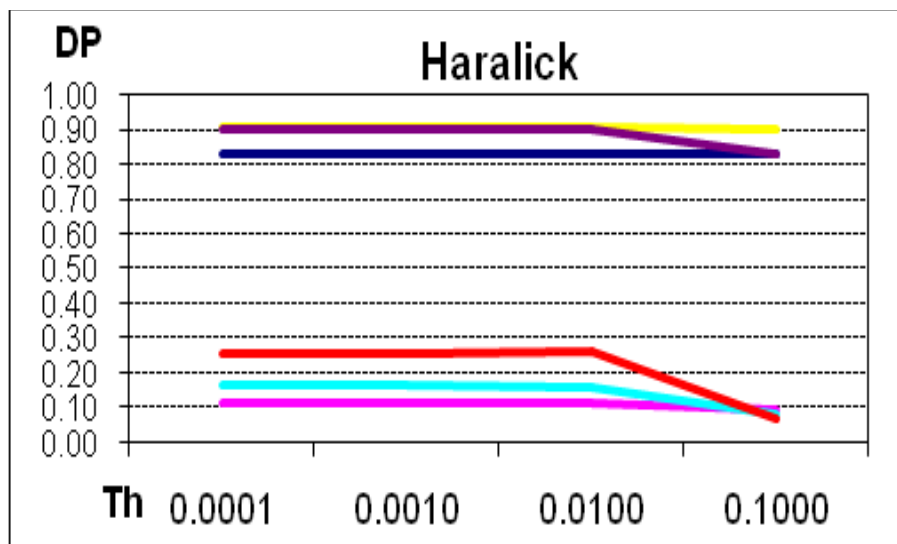


Fig. 60 Average Detection Precision (DP) vs Threshold (Th). Lines represents results measured after matching and after post-processing with different values of block sizes (8,16,32). (for clarity, table callouts are shown only on theright).

5.4.4 Conclusions

The goal of the method described is to study the ability of some standard texture descriptors to detect copies in tampered images. A common framework to test descriptors is used: a block matching approach and a post-processing step, to filter out false positives.

Experiments showed that the simplest descriptor (the Statistical) is that giving the best results in terms of precision versus execution time. Edge Histogram gives good results too, in case of small block size.

The system is tested also on JPEG compressed images and it is observed that Statistical descriptor and Edge Histogram give still the best results, but setting higher values for the threshold parameter in the matching process.

In experiments color properties are intentionally ignored, because, as said before, the goal was to test only texture as relevant feature for the application. Furthermore, block matching methods are not applicable when copies are processed by geometrical transformations. It could be interesting, in future works, to compare results achieved with texture descriptors, with those obtained using other image features (color, shape), and to combine them within a single framework.

5.5. Keypoint cluster matching approach

The method presented in this section is one of my works on copy-move detection. The method aims to use local features and keypoints to find objects in a scene that are suspiciously similar. It can be divided into different sub-steps: feature extraction, keypoint clustering, cluster matching (which is supported by a parameter optimization step) and post-processing. In this paper two different types of local descriptors (SIFT and SURF) are compared.

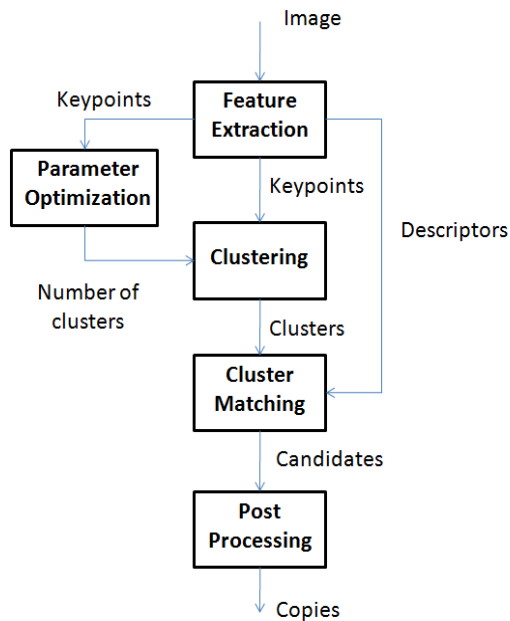


Fig. 61 Overall scheme of the proposed method

5.5.1 Features Extraction

Scientific literature of image processing offers many techniques to extract features that can be invariant to geometrical transformation. This method is focused on two

of the most used descriptors in literature: SIFT (Scale Invariant Feature Transform) [60] and SURF (Speed up robust feature) [61], which are widely used in many image processing applications.

5.5.2 Keypoint clustering

After keypoints and descriptors are computed (SURF or SIFT) for the whole image, points are grouped using an agglomerative hierarchical-tree cluster method. Starting from a matrix of Euclidean distances between each point coordinates, the weighted center of mass distance (WPGMC) is used as linkage method. For each hierarchy of the tree, if r and s are two clusters to be combined, the Euclidean distance between their weighted centroids is computed:

$$d(r, s) = \|\tilde{x}_r - \tilde{x}_s\|_2 \quad (68)$$

where \tilde{x}_r (and as well \tilde{x}_s) is defined recursively as:

$$\tilde{x}_r = \frac{1}{2}(\tilde{x}_p + \tilde{x}_q) \quad (69)$$

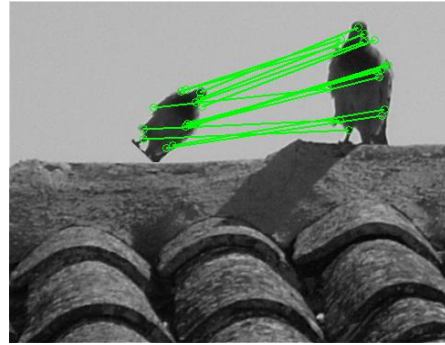
and \tilde{x}_p, \tilde{x}_q are the weighted centroids of the clusters p and q from the lower step. The “best” number of clusters is automatically and dynamically detected with a posteriori analysis. The cluster tree is formed, once for all, and then it is tested a set of candidate numbers of clusters (n_i $i=2,..n_{\max}$). For each candidate n_i , all the possible pairs of clusters within the corresponding clustering are compared, then it is counted the number of matching points (see section 5.5.3) for each pair of clusters $N(j,k)$. Only (non degenerate) matches where the number of matching points is higher than a threshold are considered:

$$M(j,k)=\begin{cases} 0 & N(j,k)<th \\ N(j,k) & otherwise \end{cases} \quad (70)$$

and the score for the candidate n_i is computed as:

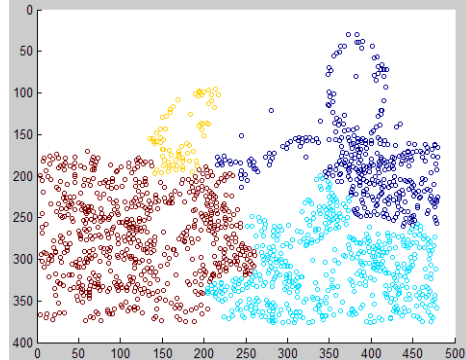
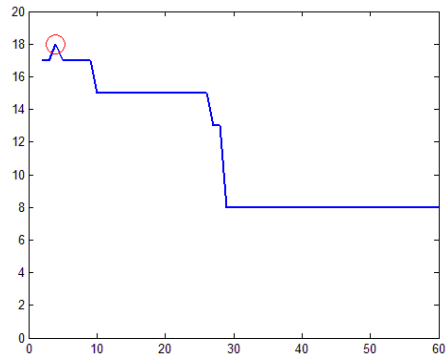
$$S_i = \sum_{j=1}^{n_i} \sum_{k=j+1}^{n_i} M(j,k). \quad (71)$$

It is selected the “best” number of clusters that maximizes this score, i.e. that gives the highest number of significant matching points (see fig. 62.c,d). In case of multiple maxima, the lowest number of clusters I selected, for efficiency.



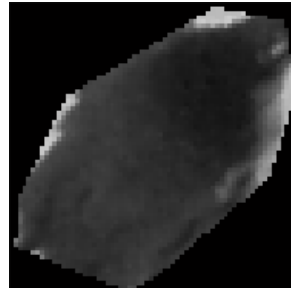
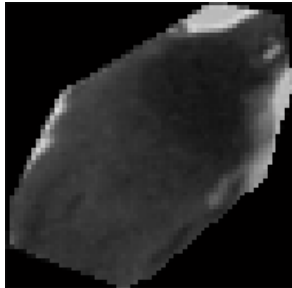
a)

b)



d)

c)



e)

f)

Fig. 62 In this photo a pigeon is copied, rotated and scaled(a). In (b) the link of centroids of cluster that match, in (c) the cluster distribution of keypoints, in (d) the relationship between the number of clusters and matching points, in (e) and (f) registration of convex hull of two clusters that match each other.

5.5.3 Custers matching

In this step it is described how two clusters of keypoints are matched. If C_i and C_j are the two clusters to be matched, for each point $i \in C_i$ its vector of descriptors \bar{d}_i is compared with the vectors \bar{d}_j of all the points $j \in C_j$. For efficiency, rather than using Euclidean distance, it is computed the angle between the two vectors:

$$\alpha_{ij} = \arccos(\bar{d}_i \cdot \bar{d}_j) \quad (72)$$

$$\text{and} \quad \alpha_1 = \min_{j \in C_j} \alpha_{ij} \quad \alpha_2 = \min_{j \in C_j - j_1} \alpha_{ij} \quad (73)$$

where j_1 is the point in C_j which vector \bar{d}_{j_1} has the minimum angle with vector \bar{d}_i . To increase robustness, matches are accepted only if the ratio of the two minimum angles, α_1 and α_2 , is less than a threshold (0.6 as in Lowe [60]). Note that, for small angles, this is a close approximation to the ratio of Euclidean distances. This matching criterion is also that used to select best the number of clusters. To improve the matching process, after the number of clusters is chosen clustering, RANSAC [83] (RANdom SAmple Consensus) is applied to select a set of inliers that are compatible with a homography transform between the two clusters. After RANSAC, outliers are discarded and the corresponding homography is estimated.

If less than 4 matches are found, RANSAC cannot be applied, and the match between the two clusters cannot be considered reliable. In case of a small number of matching points, which may occur comparing small clusters, estimated homography does not give good results. Therefore, after discarding outliers, the Moore-Penrose pseudoinverse of matrix is used to estimate homography by solving a system of linear equations:

$$X_1 = H \cdot X_2 \quad (74)$$

where X_1 and X_2 are two sets of matching points, in homogenous coordinates, and H is a 3x3 transformation matrix that projects X_1 and X_2 . Coefficients of H are

estimated:

$$\hat{H} = X_1 \cdot X_2^* \quad (75)$$

where X_2^* is the pseudoinverse matrix of X_2 . Considering X_2 as the reference set of points, the inverse of the estimated transformation is applied to X_1

$$\hat{X}_1 = \hat{H}^{-1} X_1 = \hat{H}^{-1} \cdot H \cdot X_2 \cong X_2 \quad (76)$$

thus \hat{X}_1 is an approximation of X_2 . Which cluster should be considered as reference is not important to the next steps. Then the inverse transformation is applied in order to register the two areas, and the two smallest convex hulls, A_1 and A_2 , that enclose matching points (see figg. 62.e,f) are extracted.

5.5.4 Post Processing

The content of the two areas is compared, with a texture inspection process, in order to discard matches between objects that are similar, but not copies. Prewitt and Laplacian masks are used to extract vertical, horizontal, diagonal (North-East and North West) and not-oriented edges from the two selected masks. Descriptors are then computed as the average value of pixels in the area after filtering. If the distance between the descriptor vectors of the two areas is lower than a threshold (set 10 by experiments), they are considered identical, and their match correct. I decided to use this very simple texture descriptor as it is fast and effective and does not need padding to be applied to not rectangular areas (e.g. Gabor filters). Texture analysis step is introduced to validate results coming from the previous matching process. Analyzing texture, rather than color, makes the method able to find matches even in case of color alteration of the copies (see fig. 63), if this not alter the edges of the object.

The two areas to be compared can be slightly different almost for three reasons:

- Interpolation applied by inverse transformation;
- JPEG compression after tampering;
- Approximation errors (keypoint extraction, homography estimation, inverse transformation)

With respect to the first problem, experimental tests showed that using bicubic interpolation reduces the difference between reference and approximated area. With regard to the second one, the selected descriptors extract information from image edges, which are less affected by noise introduced by JPEG compression. The third problem needs a specific solution: after inverse transformation, the two areas are not perfectly aligned (see fig. 64.b,c). For larger areas this is not a relevant problem, because edge information is averaged. In case of smaller ones, due to this misalignment, some edges in the reference area could be missing in the approximated one, the two areas may look consistently different, and descriptor values may not give correct match. Therefore a local alignment is needed. A further rigid translation is applied to the points of the approximated area in a neighborhood of 5x5 pixels centered into original coordinates. Whenever the difference between descriptors is lower than the above threshold, local alignment stops. This means that, for each match evaluation, at most 25 different translated versions of the approximated area are compared. Consistent improvement in the results justifies the increased execution time.

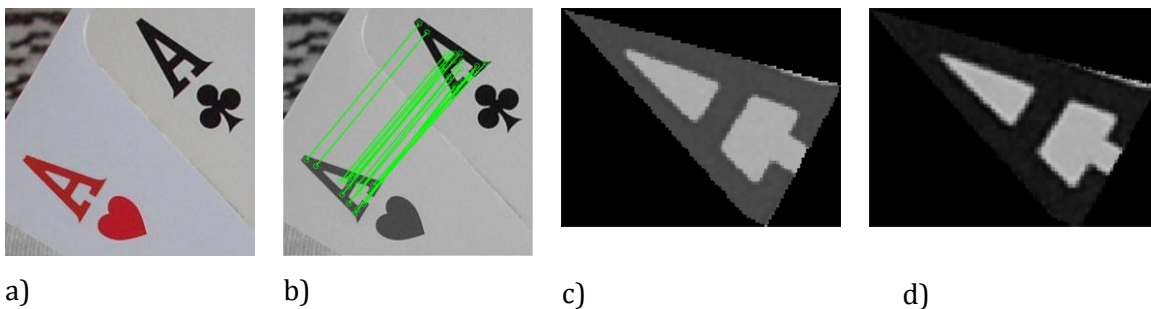
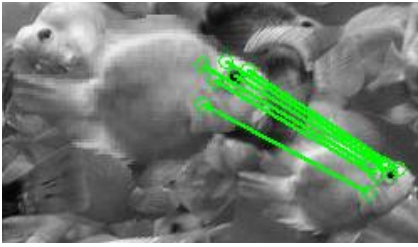
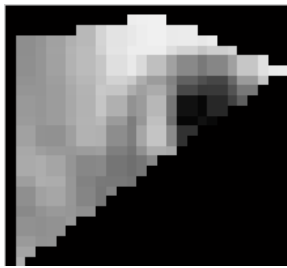


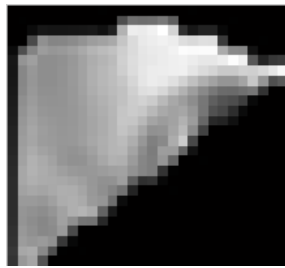
Fig. 63 (a) Input image. Ace of Hearts is the copy of ace of Clubs. (b) matches between points. (c, d) the two extracted areas: reference (c) and approximation (d). Note that match is found nevertheless the two areas have different colors. (n°clusters=3)



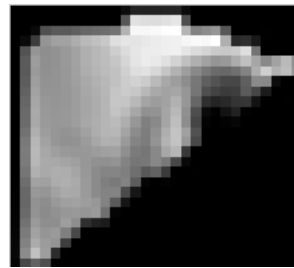
a)



b)

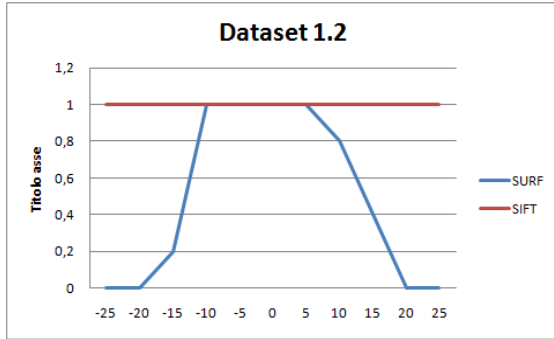


c)

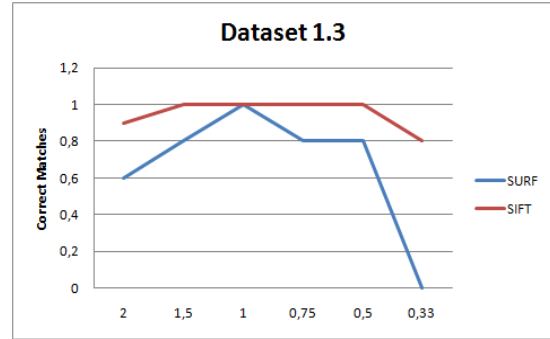


d)

Fig. 64 (a) matching points between two objects in the scene (particular). (b) reference area (part of the fish eye). Approximation without (c) and with (d) local alignment.

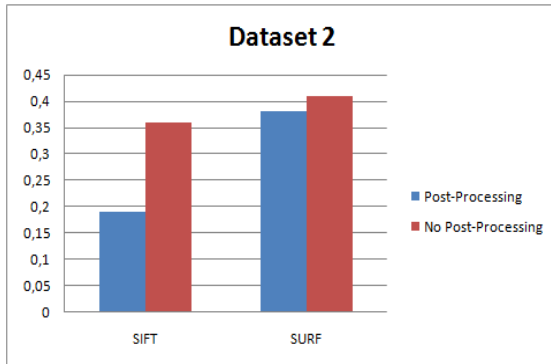


a)

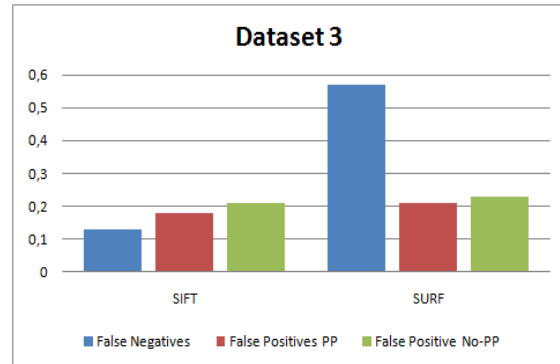


b)

Fig. 65. Percentage of Correct Matches vs rotation angle (a) and vs scale factor (b)



a)



b)

Fig. 66. (a) Percentage of detected tampered images in the not-tampered dataset, before and after the post processing step. (b) Percentage of False Negatives, and False Positives (before and after post processing) in the third dataset.

5.5.5 Experimental Results

The method presented in this section has been implemented in Matlab and executed on an Intel Core i7 PC (4 CPU, 1.6 GHz per processor, 4 GB RAM), exploiting the

Matlab parallel library to make 4 workers run simultaneously. It's been exploited the Matlab code in [121] for SIFT extraction, in [122] for SURF extraction and that in [123] for RANSAC. This approach is tested on several types of dataset.

5.5.5.1 Controlled geometrical transformations

The first group of images (**D1**) has been created by copy moving objects applying controlled geometrical transformations, to test the robustness of this method with respect to some specific attacks. 10 images are selected with very simple scene (single object, simple background) from the Torralba [109] dataset and then three sets of transformations are applied. The first subset D1.1 has been created by rotating the copy by 12 different angles in $[0,360[$ with a step of 30° . The second subset D1.2 is obtained by rotating the copy by 11 different angles in $[-25, 25]$ with a step of 5° . The third subset D1.3 is created by scaling the copy by 6 different scale factors $[2,1.5,1,0.75,0.5,0.33]$. D1 is then formed by 270 images (removing repetitions). Within dataset D1.1 we observed that SIFT points are robust to any rotation angle. In practice no false negatives have been measured in tests. On the contrary SURF points (for the used implementation), as shown also in [82], are not robust to large rotations, then no matches are found.

With respect to D1.2, as expected, SIFT points are completely invariant to the selected rotation angles. SURF points showed to be robust to small angle $[-10,10]$ rotations. Results are plotted in fig. 65.a.

For the D1.3 dataset, results are shown in fig. 65.b. Also in this case, SIFT are more robust than SURF to the specific transformation (scaling) within the selected range.

5.5.5.2 *Not tampered images*

The second dataset (**D2**) is made of 100 complex scene from the Torralba [109] dataset without applying any alterations, to test the system about the detection of false positives. Each image contains an average value of 3200 SIFT points and 1700 SURF points, and a average number of point-to-point self-matches (comparing an image with itself) of 100 (SIFT) and 150 (SURF). Fig.66 shows the number of images in the dataset into which alterations have been wrongly detected, as the dataset does not contain tampered images. Also in this case SIFT outperforms SURF, especially after the post-processing step. Furthermore, it is counted the number of objects detected before and after post-processing (note that more than one false positive can be detected in a single image), the result is that the 44% of the false matches is removed in the SIFT case, while 30% ca in the SURF case. These experiments justify the use of the post-processing step in this method, even at the cost of losing efficiency.

5.5.5.3 *Tampered images*

The third dataset (**D3**) is made of 100 images with several types of combined transformations (rotation, scaling, translation), used to create copies hard to be discovered by visual inspection.

This dataset is used to test the detection ability of this method in possible “real cases” of alterations.

Results are shown in fig. 66.b. As in the other dataset, SIFT achieves better results than SURF, above all in terms of False Negatives (tampered images that have not

been detected). In terms of False Positives (object that have been wrongly indicated as copies), the two methods achieve similar results. Experiments are repeated on the third dataset with three different values of JPEG quality compression (100, 75, 50). Tests showed that the method is very robust to JPEG compression (results are quite similar to those in fig. 66.b). This can be explained as both SIFT and SURF extract information from image edges, which are less affected by noise introduced by JPEG compression, as well the texture descriptor used in the post-processing step.

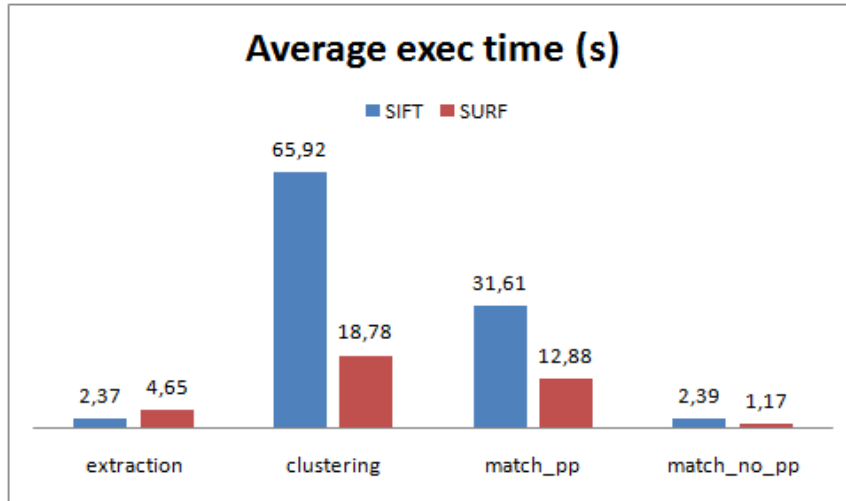


Fig.67. Average execution time (in second) for the steps of the proposed method

5.5.5.4 Temporal Efficiency

In terms of efficiency, in theory SURF descriptors should be faster than SIFT ones. This is the most important advantage of the SURF on SIFT. In practice due to the two specific implementations we use, time spent to extract SURF is higher than that for SIFT (fig.67)

Time for clustering is higher in the SIFT case as typically SIFT extracts more keypoints, then more time is needed in the parameter optimization step.

Fig. 67 shows also the average time spent for matching, with and without the post processing step. The improved experimental results, in terms of reduced false positives (see section 5.5.5.2) justifies the increased execution time.

5.5.6 Conclusions

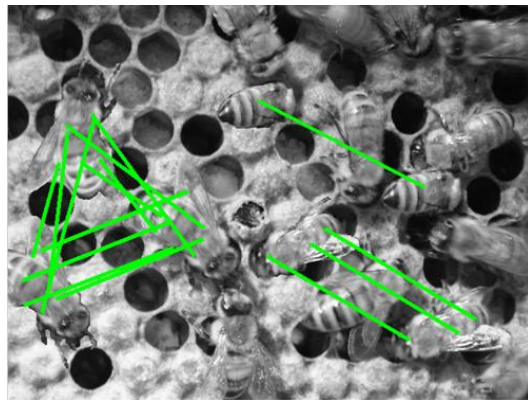
Local features proved to be extremely suited for a copy move detection method, in case of geometrical attacks. Test showed that SIFT outperforms SURF for this specific goal. In the presented approach clustering helps in reducing the number of false matches between keypoints, as matches are accepted only when clusters match. In particular, the adaptive clustering method was designed to maximize the number of matches between significant clusters. Therefore, it is expected the same to have some false matches. For this reason in the method it has been introduced a post-processing step, based on texture analysis, to reduce the number of false positives at the cost of some more computational effort. Only objects (clusters of keypoints) which have both similar shape and texture can be considered as real copies. Experimental results justify the increased execution time. The proposed approach achieved excellent results, but it strictly depends on keypoint extraction: keypoints are not extracted in homogeneous regions, so if tampering is made by copy-moving a homogeneous portion of the image, cannot be detected by a feature based approach. It would be an interesting challenge to build an image forensics framework that will combine a feature based approach, that is robust to geometrical attacks, with a block-matching technique, that typically works also in case of homogenous areas.



a)



b)



c)

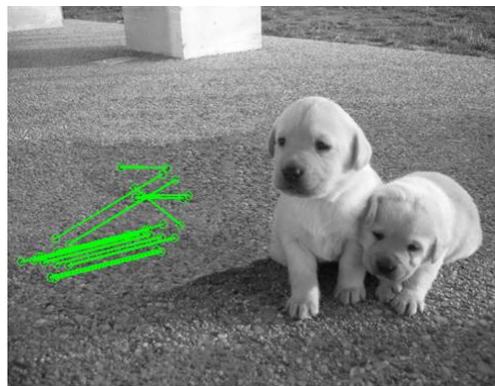
Fig. 68 (a) Original and (b) tampered image. (c) Result with keypoints matching and with cluster matching (n° clusters = 4, matches between centroids are shown).



a)



b)



c)

Fig.69 (a) Original and (b) tampered image. (c) Results with keypoints matching and with cluster matching (n° clusters = 4, matches between centroids are shown).

5.6. Triangles matching approach

This new approach is based on the analysis of triangles of local key points, rather than blocks of pixels, or single key points. The aim is to find the structure of the copies by analyzing the spatial distribution of the key points. Matching between triangles is done both extracting local features inside the triangles (color) and comparing their geometrical properties (angles). Before describing the method, it is given a brief overview on Delaunay Triangulation, that is the “core” of the proposed method.

5.6.1 *Delaunay Triangulation*

A common approach to discretize a domain is to use a triangulation. Given a set T of planar points, the Delaunay triangulation $D(T)$ of T points is made such that no points in T is inside any other triangle of the same of $D(T)$. Delaunay triangulations maximize the minimum angle of all the angles of the triangles in the triangulation.

Delaunay triangulation is a tool used for obtaining good quality meshes both in Computer Graphics for 3D object representation and when using meshes for equations and problem representation [124]. A mesh can generally be described as a number of points connected in some way by lines. Many complex high dimensional problems can be expressed with graphs, and a good quality mesh lets researcher get closer to a numerically stable solution for these problems. The circle circumscribed about a Delaunay triangle has its center at the vertex of a Voronoi polygon[125].

5.6.2 Triangles Matching

The idea behind the method is very simple. All the objects in a scene may be represented as a set of connected triangles. This model is the basis for the representation of 3D objects in Computer Graphics. In this method this representation is applied to 2D images.

First, interest points from an image are extracted. Three of the most common algorithms for extracting keypoints (SIFT, SURF and Harris, see chapter 2) are used. Then a Delaunay triangulation is build onto the extracted points. Image is therefore subdivided into triangles, which include pixels with very similar features. In fact Delaunay triangulation (rather than Voronoi tessellation) is used as its atomic element typically does not include edges of the objects in the scene, then its content can be completely described by its color features.

From each triangle the first n dominant colors are extracted: each color channel is quantized into b bins and a 3D histogram is build with the pixels of the triangle. The n most frequent values of the histogram are taken as the dominant colors of that triangle (n and b will be further discussed in the experimental section). Each triangle is represented by $3*n$ values (n values per channel). For this purposes information about color frequencies is discarded.

Further triangle areas and inner angles are computed. Angles are taken in counterclockwise order starting from the maximum one. This solution helps to make this method robust to affine transformation, as discussed below. Input image is finally segmented into triangles (see fig. 70), which are described by their dominant colors, their areas and their ordered sequence of angles.

To find possible copy-moved regions, we look for similar triangles into the image. Triangles are matched comparing two different features: colors and angles.

First, triangles are sorted according to the L_1 norm of their color vectors. The sorted list of triangles is then scanned and the features of each triangle compared to those of the next triangles in the list, within a fixed window (a percentage of the number of triangles). In adaptive window approach, triangles are compared up to

those in the list which distance is below a threshold, but this solution proved to be slower than the fixed window approach, without improving results.

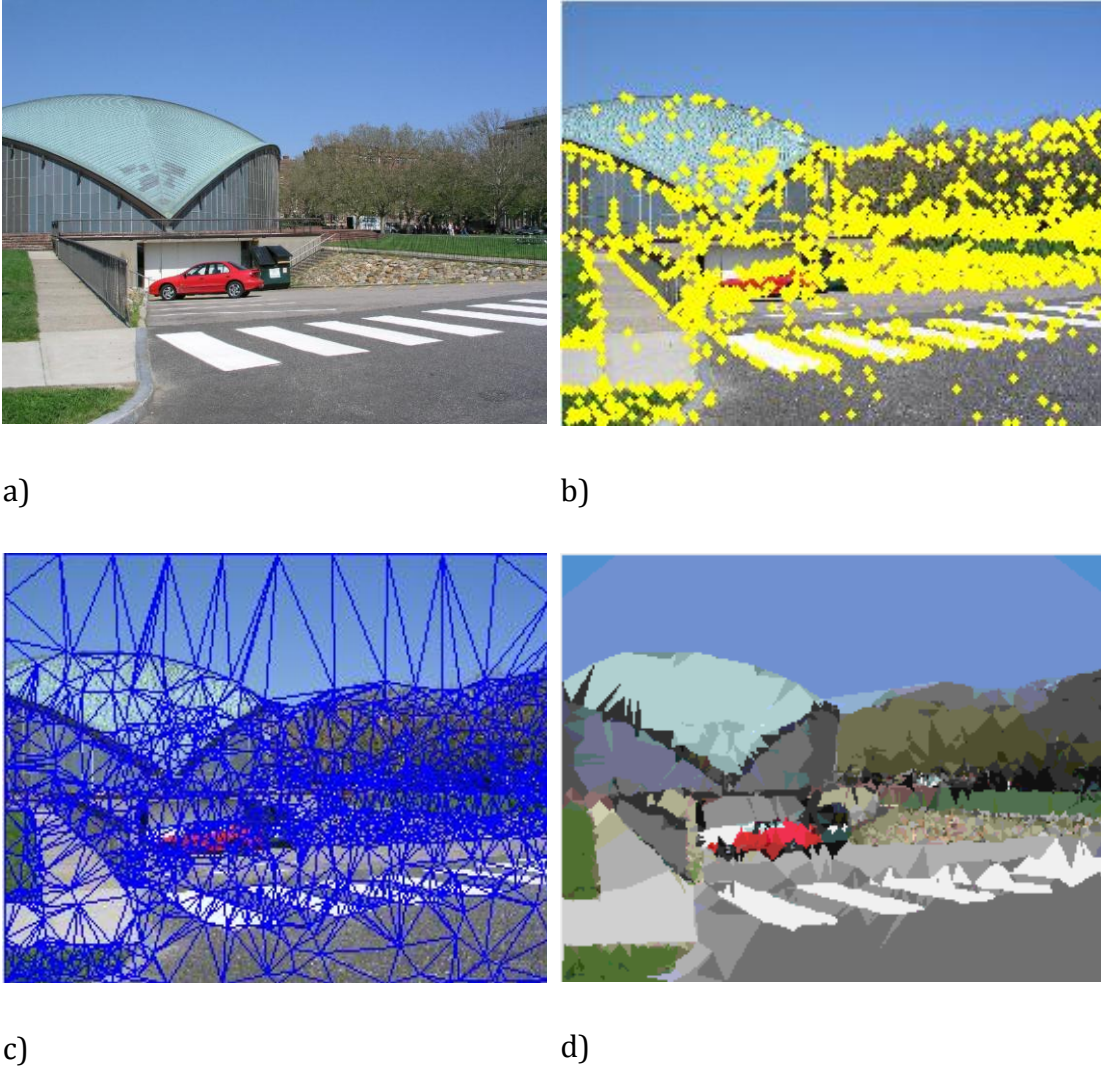


Fig. 70 Original image (a), with superimposed SIFT points (b), with Delaunay triangulation (c) and the segmented image, where each triangle is filled with its first dominant color.

If both the Sum of the Absolute Deviation (SAD) of the color vectors and of the angles of the two triangles are below a threshold, the two triangles are considered similar. If j and k ($k > j$) are the indexes of the two triangles to be compared:

$$\begin{aligned}
& \sum_{i=1}^{3 \cdot n} |C_i^j - C_i^k| < TH^c \\
& \sum_{l=1}^3 |a_l^j - a_l^k| < TH^a \quad (77) \\
& (k - j) < w_s
\end{aligned}$$

where w_s is the fixed window size (computed as a percentage of the number of triangles), C is the color vector (made of $3 \cdot n$ values), a is the angle vector in radians (in which angles are sorted as described above), TH^c and TH^a are two thresholds, that will be discussed in the next section. When comparing angles, sorted as described, two triangles may be match even in case of rotation or scaling. Thus the method is designed to find copied objects also in case of geometric transformation.

To reduce false positives, two triangles j and k are compared only if the ratio between their areas:

$$r_A = \frac{\min(A_j, A_k)}{\max(A_j, A_k)} > 0.25 \quad (78)$$

This solution limits the maximum detectable scale of the copied objects to 4, but thanks to this solution the 15-20% of the wrong matches are deleted.

After the matching process, the result is a list of pairs of triangles. To further delete false matches, the centroids of these triangles are computed, and RANSAC (RANDOM SAMPLE CONSENSUS[83]) is applied to the set of matching centroids, to select a set of inliers that are compatible with a homography transformation. If less than 4 matches are found, RANSAC cannot be applied, and the match is considered not reliable.

5.6.3 Experimental Results

This method has been also implemented in Matlab and executed on an Intel Core i7 PC (4 CPU, 1.6 GHz per processor, 4 GB RAM), exploiting the Matlab parallel library to make 4 workers run simultaneously. The implementation of the method uses the Matlab code in [121] for SIFT extraction, in [122] for SURF extraction and that in [123] for RANSAC. This approach is tested on several types of dataset.

The first group of images (D1) has been created by copy-pasting objects applying controlled geometrical transformations, to test the robustness against some specific attacks. 10 not compressed images with very simple scenes (single object, simple background) are selected from the Torralba [109] dataset and 3 sets of transformations are applied. The first subset D1.1 has been created applying to the copies 11 different types of rotation around the angle zero in $[-25^\circ, 25^\circ]$ with step 5° .

The second subset D1.2 has been created by rotating the copies by 12 different angles in $[0^\circ, 360^\circ[$ with a step of 30° . The third dataset by scaling the copies by 6 different scaling factors $[2, 1.5, 1, 0.75, 0.5, 0.33]$. D1 is then formed by 290 images.

The second dataset (D2) is made of 50 JPEG-compressed images (D2.1) with simple translated copies, and 50 JPEG-compressed images (D2.2) with combined transformations (rotation, scaling, translation), used to create copies hard to be discovered by visual inspection. This dataset is used to test the detection ability of the method in possible true cases of alterations, and with compression (until quality factor 50).

The first dataset (D1) is used to find the optimal parameters of the method. Tests are repeated tuning some of the parameters, to study how results depends on them.

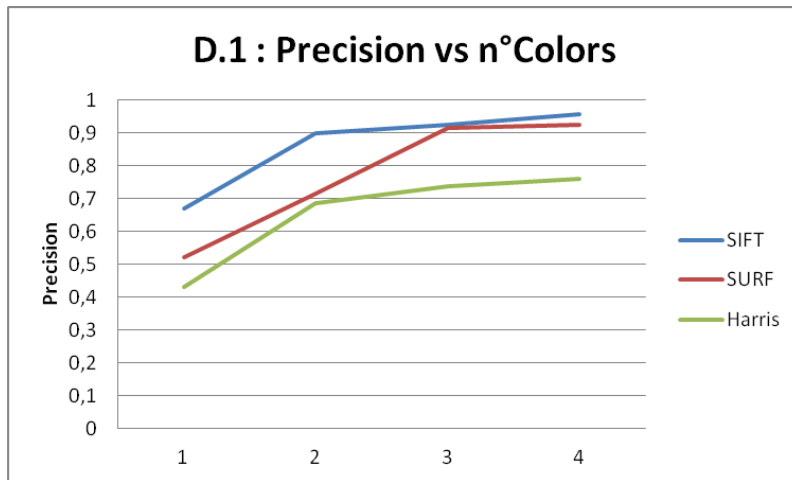


Fig. 71. Precision vs number of colors, for images in the Dataset D.1 which have copies with no rotation or scaling ($TH^a=0,25$)

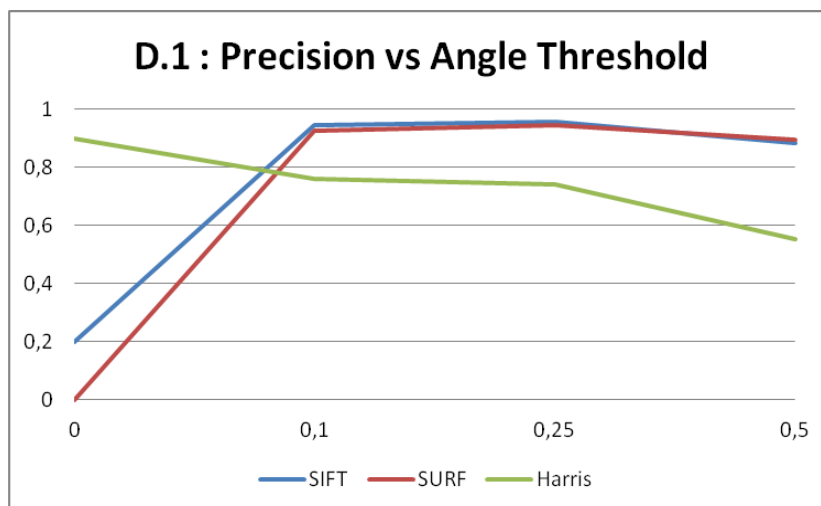


Fig. 72. Precision vs angle threshold TH^a , for images in the Dataset D.1 which have copies with no rotation or scaling ($n^{\circ}colors=4$).

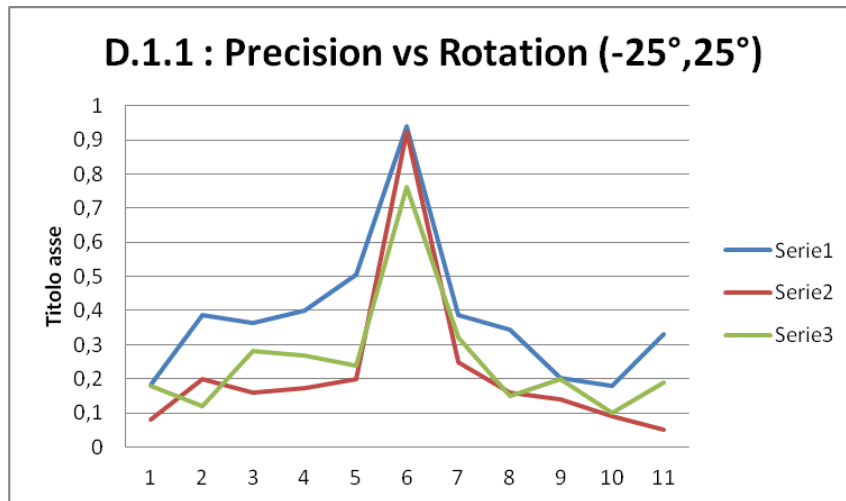


Fig. 73. Precision vs angle of rotation, for images in the Dataset D.1.1 which have “rotated” copies ($TH^a=0,25$ and $n^{\circ}colors=4$).

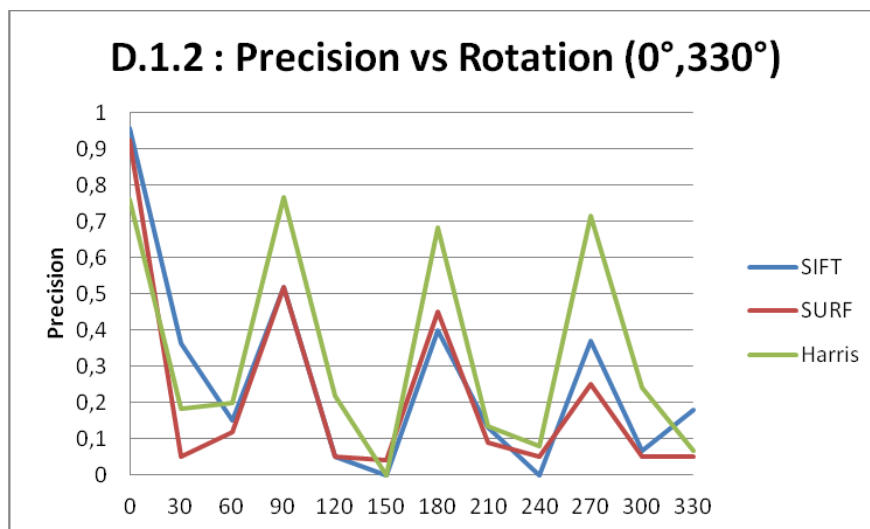


Fig. 74. Precision vs angle of rotation, for images in the Dataset D.1.2 which have “rotated” copies ($TH^a=0,25$ and $n^{\circ}colors=4$).

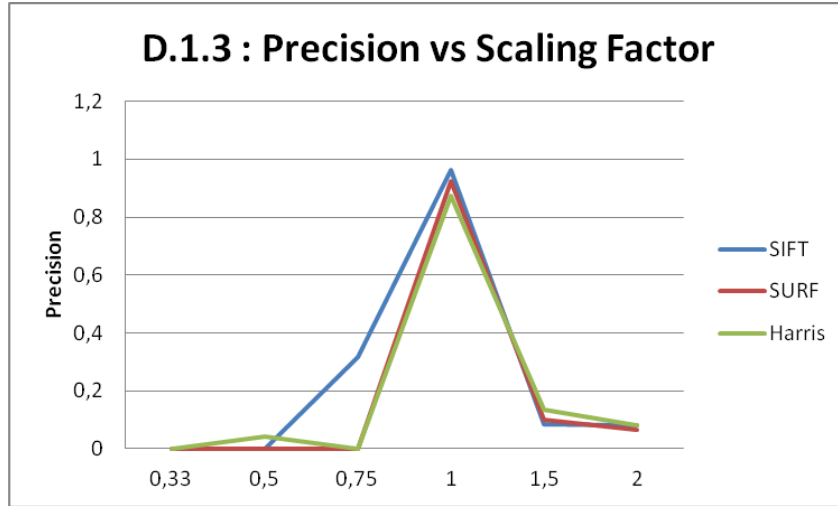


Fig. 75. Precision vs scaling factor, for images in the Dataset D.1.3 which have “scaled” copies ($TH^a=0.25$ and $n^{\circ}colors=4$).

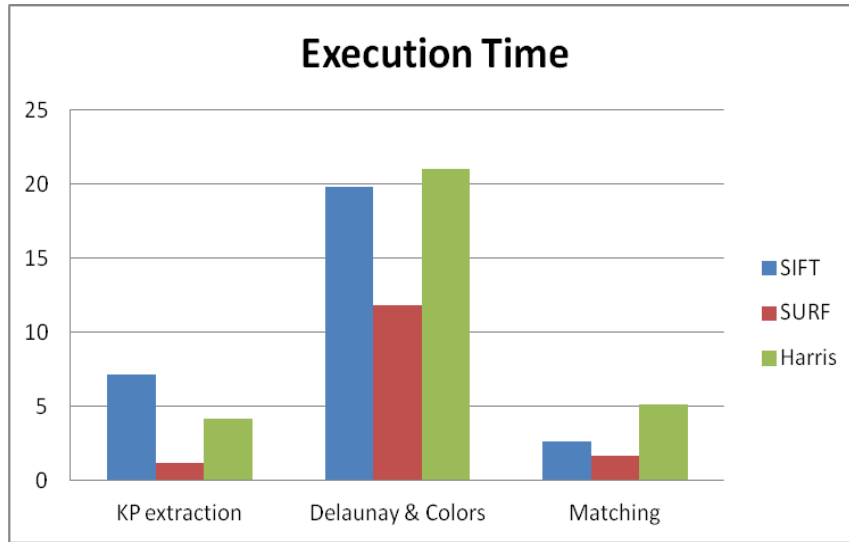


Fig. 76. Execution time, in seconds, of the different steps of the process.

In particular we measured results using:

- the number of bins to quantize color channel $b=8$,
- the number of dominant colors $n=1..4$,
- the color threshold $TH^C=0$,
- the angle threshold $TH^a= \{0;0.1;0.25;0.5\}$,
- the size of the fixed window $w_s=N_T/50$, where N_T is the number of triangles

in an image. The number of bins is a critical parameter and should need a in-depth analysis. In this paper we fixed it to 8. It has been decided to compare only triangles that have exactly the same dominant colors ($TH^c=0$), otherwise too much false positives would be measured. w_s does not influence too much results, then we decided to fix it to a value that is a good trade-off between precision and execution time. TH^a and number of colors has been tuned to show their influences on tests.

To evaluate results, the precision of the method is computed as the ratio of the number of true output matches and the number of total matches.

Fig. 71 shows results of the method when tuning the “number of colors” parameter. Precision is measured only within images in the dataset D.1 in which copies are not modified by geometric tampering. SIFT-based and SURF-based methods outperform Harris-based method within this dataset. SIFT achieves best results among all the methods for all the number of colors.

Fig. 72 shows results of the method when tuning the angle threshold parameter. Precision is measured within the same dataset of the previous test. Also in this case, SIFT and SURF perform better than Harris, except in case of $TH^a=0$, where Harris achieves its best results.

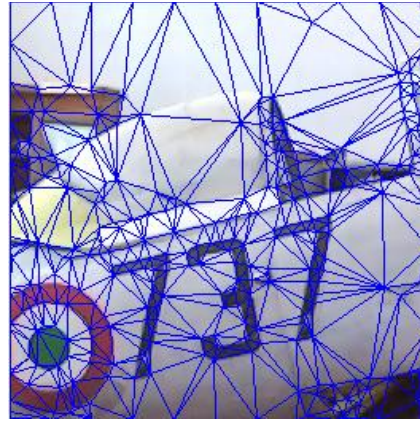
Using the optimal parameters evaluated from these first tests, we made other tests on rotated (fig. 73,74) and scaled (fig. 75) copies. Within the dataset D1.1, SIFT achieves best results, while Harris and SURF have similar results. Within D.1.2 Harris achieves its best results, above all whenever the rotation angle is multiple than 90° , while SIFT and SURF have almost similar results. Note that best results are achieved for angles that are multiples than 90° , as in these cases triangles are less distorted by rotation. In case of scaling (D.1.3) all the three methods get similar results. SIFT performs slightly better than the other two methods only for a scaling factor of 0.75. Results are encouraging, but can be improved if a deep analysis on the point distribution is done.

Within the D2 dataset, for the D2.1 images, respectively a precision of 87.3% (SIFT), 86.2% (SURF), and 69% (Harris), which are similar to the results achieved within the D1 dataset, for images with no rotation or scaling. For the D2.2 images, we measured respectively a precision of 25,1% (SIFT), 24.3% (SURF), and 20,7% (Harris). As discussed above, the lower precision values are due to the distortion of the angles in case of rotation or scaling.

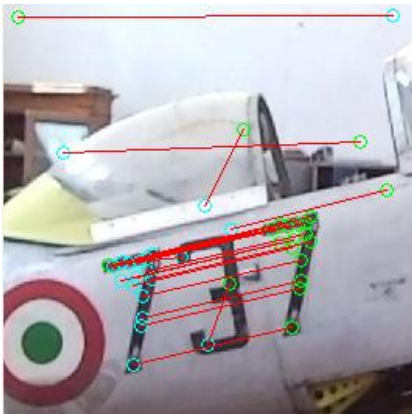
In terms of efficiency, most of the time is spent to extract color features from the triangles (see fig.76), and it depends on the number of triangles, i.e. the number of extracted keypoints. Time spent to extract SURF points is, as expected, much lower than that for SIFT points, while Harris algorithm is the slowest in this step. In every case, time for the whole process is lower than one minute. Fig.76 shows some visual examples of the results of our method.



a)



b)

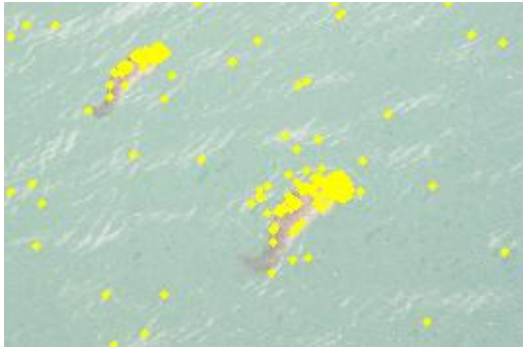


c)

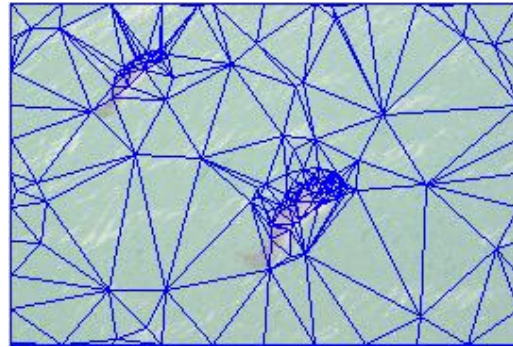


d)

Fig. 77 Some visual results. Original image with superimposed key-points (SIFT a), Delaunay triangulation (b), matching triangles before (c) and after (d) RANSAC. Images are taken from the D2.1.



i)



l)



m)

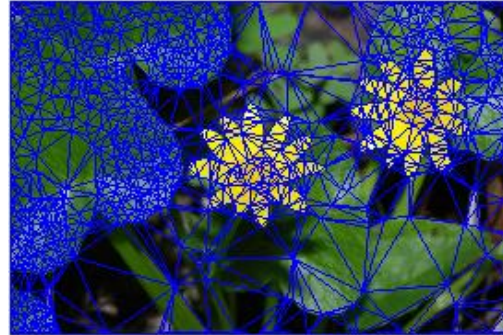


n)

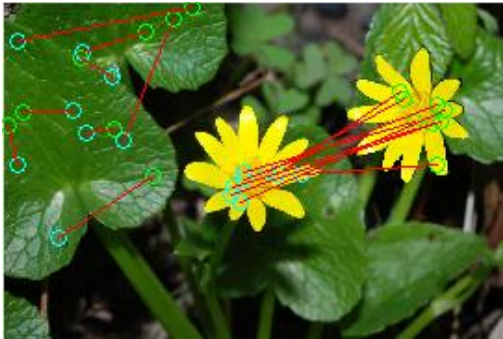
Fig. 78 Some visual results. Original image with superimposed key-points (Harris i), Delaunay triangulation (f), matching triangles before (m) and after (n) RANSAC. Images are taken from the D1.2 (scale factor 0.75).



e)



f)



g)



h)

Fig.79 Some visual results. Original image with superimposed key-points (SIFT e), Delaunay triangulation (b), matching triangles before (g) and after (h) RANSAC. Image taken from the D1.1 (rotation =120°)

5.6.4 Conclusions

Modeling objects with triangles is a very well known solution for Computer Graphics applications, but this is a totally new concept for Image Forensics problems. Many state-of-the-art techniques typically use block matching methods. Those methods that are robust to geometrical transformations extract from the inside of the blocks low level features that are invariant to rotation and scaling.

The method presented above is based on the analysis of the structure of the objects in the scene, that is represented as a mesh of triangles. Each object is divided into triangles, and matches are searched between those that have both similar structures (angles) and content (color of the inner pixels).

The strongest point of this approach is that comparing the structure (angles) of two triangles, similarities can be found also in case of rotation and scaling, without extracting any particular features. Therefore this method is built to be robust to geometrical transformations. Furthermore this method can be adapted in the future to find copies also in case of anisotropic transformation, as matches are searched between the atomic elements of the objects.

Nevertheless results depend on the extraction of the interest points of the image. Triangulation is built onto these points, so if a rotation or a scaling transformation is applied to the pasted objects, triangles may be partially distorted, and the method may fail. Results are encouraging, but a further analysis is needed onto the stability of the extracted interest points used to build the triangle mesh.

Another point that will need further analysis is the color quantization, which may strongly influence triangle color matching. I plan to study this problem in my future works.

CONCLUSIONS

In this PhD thesis advanced image inspection techniques are presented. These techniques are based on texture and local keypoint analysis. These techniques deal with Texture scale detection, Visual Saliency (saliency map) and Image Forensics.

-Texture scale detection: Detecting scale in textured images is a very hard task, and a relatively unexplored problem. The technique presented in this thesis analyzes the distribution of the interest points in the image, by means of the Keypoint Density Maps, a novel instrument for Image Analysis applications. The results, as seen in chapter 3, are very encouraging: with respect to the reference method, experiments showed that the novel method achieves similar precision in case of small textures, but outperforms it in cases of large textures, and in case of near-regular textures. Furthermore, Keypoint Density Maps (with SIFT) have been yet successfully used in visual saliency detection, and may be a versatile instrument for several image analysis application: texture discrimination or description, image segmentation, etc.

-Visual Saliency: Visual saliency has been investigated for many years but it is still an open problem, especially if the aim is to investigate the relationship between synthetic maps and human fixation points. The purpose of method is to study how computer generated keypoints are related to real fixation points. No color information has been used to build my saliency maps, as keypoints are typically related only to image texture property.

Even if only texture information is used, experimental results show that my method is very competitive with respect of two of the most cited low-level approaches. Judd's method achieves better results as it is a supervised method which has been trained with the fixation maps within the selected dataset. In my future works i

want to study new color based saliency techniques to be integrated with my proposed approach, to improve experimental results.

-Image forensics(1): The goal of the first method described [51] is to study the ability of some standard texture descriptors to detect copies in tampered images. A common framework to test descriptors is used: a block matching approach and a post-processing step, to filter out false positives. Experiments showed that the simplest descriptor (the Statistical) is that giving the best results in terms of precision versus execution time. Edge Histogram gives good results too, in case of small block size.

The system is tested also on JPEG compressed images and it is observed that Statistical descriptor and Edge Histogram give still the best results, but setting higher values for the threshold parameter in the matching process. The goal was to test only texture as relevant feature for the application, so color properties are ignored.

Block matching methods are not applicable when copies are processed by geometrical transformations. It could be interesting, in future works, to compare results achieved with texture descriptors, with those obtained using other image features (color, shape), and to combine them within a single framework.

-Image Forensics(2): In the second copy-move detection technique reported, local features proved to be extremely suited for a copy move detection, in case of geometrical attacks. Test showed that SIFT outperforms SURF for this specific goal. Clustering helps in reducing the number of false matches between keypoints, as matches are accepted only when clusters match. In particular, the adaptive clustering method was designed to maximize the number of matches between

significant clusters. In the method it has been introduced a post-processing step, based on texture analysis, to reduce the number of false positives at the cost of some more computational effort. Only objects (clusters of keypoints) which have both similar shape and texture can be considered as real copies. Experimental results justify the increased execution time. The proposed approach achieved excellent results, but, if tampering is made by copy-moving a homogeneous portion of the image, it cannot be detected by a feature based approach because keypoints are not extracted in homogeneous regions. It would be an interesting challenge to build an image forensics framework that will combine a feature based approach, that is robust to geometrical attacks, with a block-matching technique, that typically works also in case of homogenous areas.

-Image Forensics(3): The third copy-move detection technique is based on the analysis of the structure of the objects in the scene, that is represented as a mesh of triangles.

The strongest point of this approach is that comparing the structure (angles) of two triangles, similarities can be found also in case of rotation and scaling, without extracting any particular features. Therefore this method is built to be robust to geometrical transformations. Furthermore this method can be adapted in the future to find copies also in case of anisotropic transformation, as matches are searched between the atomic elements of the objects.

Nevertheless results depend on the extraction of the interest points of the image. Triangulation is built onto these points, so if a rotation or a scaling transformation is applied to the pasted objects, triangles may be partially distorted, and the method may fail. Results are encouraging, but a point that will need further analysis is the color quantization, which may strongly influence triangle color matching. I plan to study this problem in my future works

REFERENCE

- [1] Tamura, H. and Mori, S. and Yamawaki, T.. "Textural features corresponding to visual perception" on IEEE Transactions on Systems, Man and Cybernetics, 8:460—473, 1978
- [2] Coggins, J.M. , "A framework for Texture Analysis Based on Spatial Filtering" Ph.D. Thesis Computer Science Department, Michigan State University, East Lansing, Michigan, 1982
- [3] C. H. Chen, L.F. Pau, P.S.P. Wang. Handbook of Pattern Recognition and Computer Vision, (eds) pp. 207-248, World Scientific Publishing
- [4] Tamura, H., S. Mori, and Y. Yamawaki, "Textural Features Corresponding to Visual Perception," IEEE Transactions on Systems, Man, and Cybernetics, 8:460-473, 1978
- [5] Sklansky, J., "Image Segmentation and Feature Extraction," IEEE Transactions on Systems, Man, and Cybernetics, SMC-8, pp. 237-247, 1978.
- [6] Haralick, R.M., "Statistical and Structural Approaches to Texture," Proceedings of the IEEE, 67:786-804, 1979
- [7] Richards, W. and A. Polit, "Texture matching," Kybernetik, 16:155-162, 1974
- [8] Zucker, S. W. and K. Kant, "Multiple-level Representations for Texture Discrimination," In Proceedings of the IEEE Conference on Pattern Recognition and Image Processing, pp. 609-614, Dallas, TX, 1981
- [9] Hawkins, J. K., "Textural Properties for Pattern Recognition," In Picture Processing and Psychopictorics, B. Lipkin and A. Rosenfeld (editors), Academic Press, New York, 1969
- [10] Julesz, B., E. N. Gilbert, L. A. Shepp, and H. L. Frisch, "Inability of humans to discriminate between visual textures that agree in second-order statistics - revisited," Perception, 2, pp. 391-405, 1973
- [11] Julesz, B., "Visual Pattern Discrimination," IRE Trans. on Information Theory, IT-8, pp. 84-92, 1962
- [12] Julesz, B., "Experiments in the visual perception of texture," Scientific American, 232, pp.34-43, 1975

- [13] B. Julesz. "Textons, the Elements of Texture Perception, and their Interactions." *Nature*, 290:91-97, March 1981.
- [14] J.J. Kulikowski, S. Marcelja, and P.Bishop. "Theory of spatial position and spatial frequency relations in the receptive fields of simple cells in the visual cortex." *Biol. Cybern*, 43:187-198, 1982.
- [15] J. Daugman. "Entropy reduction and decorrelation in visual coding by oriented neural receptive fields." *Transactions on Biomedical Engineering* . 1:107-114, 1989.
- [16] J.R. Bergen and M. S. Landy. "*Computational Modeling of Visual Texture Segregation*", chapter 17, 253--271. MIT Press, 1991.
- [17] Gadelmawla, ES. A vision system for surface roughness characterization using the gray level co-occurrence matrix. *NDT E International* 577--588 2004
- [18] Haralick RM, Shanmugam K, Dinstein I. "Textural features for image classification." *IEEE Trans Syst, Man Cybernet* 3:610 -621. 1973;
- [19] Chen CH. "A study of texture classification using spectral features." In *Proceedings of the Sixth International Conference on Pattern Recognition*. p. 1074-7. 1982
- [20] Konak ES. "A content-based image retrieval system for texture and color queries." Master Thesis, Computer Engineering and the Institute of Engineering and Science, Bilkent University; 2002
- [21] Umarani, C. , Radhakrishnan, S. and Ganesan, L. "Combined statistical and structural approach for unsupervised texture classification." *International Journal on Graphics, Vision and Image Processing*. 7:31--36, 2007
- [22] R.K. Goyal, W.L. Goh, D.P. Mital, K.L. Chan, "Invariant element compactness for texture classification" *Proceedings of the Third International Conference on Automation, Robotics and Computer Vision (ICARCV'94)*, Singapore, 1902-1096. 1994
- [23] D.P. Mital, W.L. Goh, K.L. Chan, R.K. Goyal. "A translation rotation and scale invariant texture analysis technique based on image granularity", *Proceedings of the Fifth International Symposium on Robotics and Manufacturing*, 1994
- [24] R. Goyal, W.L. Goh, D.P. Mital, K.L. Chan, "Scale and rotation invariant texture analysis based on structural property." *Proceedings of IECON*, No. 2, 1995, pp. 1290-1294
- [25] G. Eichmann, T. Kasparis, "Topologically invariant texture descriptors", *Comput. Vision Image Process*. 41:267-281. 1998

- [26] W-K. Lam, C-k. Li, "Rotated texture classification by improved iterative morphological decomposition", IEE Proc—Vision Image Signal Process. 144 (3):171–179. 1997
- [27] Larry S. Davis, "Polarogram: a new tool for image texture analysis", Pattern Recognition (3):219–223. 1981.
- [28] M. Mayorga, L. Ludman, "Shift and rotation invariant texture recognition with neural nets, Proceedings of IEEE International Conference On Neural Networks", 4078–4083. 1994
- [29] Y.L. Sheng, H. Arsenault, "Experiment on pattern recognition using invariant Fourier–Mellin descriptors", Opt. Soc. Am. A 3 (6) (1986) 771–776
- [30] Y. Sheng, J. Duvernoy, "Circular-Fourier-radical-Mellin transform descriptors for pattern recognition", JOSA Commun. 3 (6) (1986) 885–888
- [31] H.H. Arsenault, Yunlong Sheng, "Properties of the circular harmonic expansion for rotation-invariant pattern recognition", Appl. Opt. 25 (18):3225–3229. (1986)
- [32] J. Duvernoy, "Optical digital processing of directional terrain textures invariant under translation, rotation, and change of scale", Appl. Opt. 23 (6): 828–837. (1984)
- [33] M.K. Tsatsanis, G.B. Giannakis, "Object and texture classification using high order statistics", IEEE Trans. PAMI 14 (7):733–750. (1992)
- [34] L. Wang, G. Healey, "Using Zernike moments for the illumination and geometry invariant classification of multispectral texture", IEEE Trans. Image Process. 7 (2):196–203. (1998)
- [35] M. Pietikainen, T. Ojala, Z. Xu, "Rotation-Invariant texture classification using feature distributions," Pattern Recognition 33:43–52. 2000
- [36] Pentland, A. P. Fractal-Based Description of Natural Scenes, Transaction. IEEE transactions on pattern analysis and machine intelligence, vol. PAMI-6, NO. 6, 1984
- [37] Keller, J. M., S. Chen, and R. M. Crownover, "Texture Description and Segmentation through Fractal Geometry," Computer Vision, Graphics, and Image Processing, 45, pp. 150-166, 1989.
- [38] Voss, R., "Random fractals: Characterization and measurement," In Scaling Phenomena in Disordered Systems, edited by R. Pynn and A. Skjeltorp, (Editors), Plenum, New York, 1986.

- [39] Park, D.K. and Jeon, Y.S. and Won, C.S. "Efficient use of local edge histogram descriptor." In journal image, (2):3, 2000
- [40] Ro, Y.M.R. and Kim, M.K. and Kang, H.K.K. and Manjunath, BS and Kim, J.K. "MPEG-7 homogeneous texture descriptor." ETRI journal. 23:41—51. 2001
- [41] Wu, P., Manjunath, B.S., Newsam, S., Shin, H.D. "A texture descriptor for browsing and similarity retrieval." Journal of Signal Processing: Image Communication, Vol 16 Issue 1-2, pp:33-43. 2000
- [42] Miyamoto, E. and Merryman Jr, T. "Fast calculation of Haralick texture features". Human computer Interaction Institute & Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, 2008
- [43] Manjunath, B., Ma, W. "Texture features for browsing and retrieval of image data." IEEE Trans on Pattern Analysis and Machine Intelligence 18:837–842. 1996
- [44] Jain A.K. and Farrokhia F. 1991, Unsupervised Texture Segmentation Using Gabor Filters, Pattern Recognition, Vol. 24, No. 12, pp. 1167-86.
- [45] L. Wang, G. Healey, "Using Zernike moments for the illumination and geometry invariant classification of multispectral texture", IEEE Trans. Image Process. 7 (2) (1998) 196–203
- [46] Sutton, R. and E. L. Hall, "Texture Measures for Automatic Classification of Pulmonary Disease,"IEEE Transactions on Computers, C-21, pp. 667-676, 1972.
- [47] Landeweerd, G. H. and E. S. Gelsema, "The Use of Nuclear Texture Parameters in the Automatic Analysis of Leukocytes," Pattern Recognition, 10, pp. 57-61, 1978.
- [48] Jain, A. K. and S. K. Bhattacharjee, "Text Segmentation Using Gabor Filters for Automatic Document Processing," Machine Vision and Applications. 5:169—184. 1992
- [49] Jain, A. K., S. K. Bhattacharjee, and Y. Chen, "On Texture in Document Images," IEEE Conference on Computer Vision and Pattern Recognition, Champaign. 677—680. 1992
- [50] Haralick, R. M., K. Shanmugam, and I. Dinstein, "Textural features for image classification," IEEE Transactions on Systems, Man, and Cybernetics, SMC-3, pp. 610-621, 1973
- [51] Ardizzone, E. and Bruno, A. and Mazzola, G., "Copy-move forgery detection via texture description", Proceedings of the 2nd ACM workshop on Multimedia in forensics, security and intelligence. 59—64. 2010

- [52] Fauqueur, J. and Kingsbury, N. and Anderson, R. "Multiscale keypoint detection using the dual-tree complex wavelet transform." IEEE International Conference on Image Processing, 1626-1628. 2006
- [53] Wang, D. and Srihari, S.N. "Analysis of form images." in IJPRAI 8:1031--1052 1994
- [54] Schmid, C. and Mohr, R. and Bauckhage, C. "Comparing and evaluating interest points." Sixth International Conference on Computer Vision ICCV, pag (230-235) 1998
- [55] Kordelas, G. and Daras, P., "Robust SIFT-based feature matching using Kendall's rank correlation measure", IEEE International Conference on Image Processing, 325—328. 2009
- [56] Chetverikov, D. and Szabo, Z. "A simple and efficient algorithm for detection of high curvature points in planar curves." 1999
- [57] Rangarajan, K. and Shah, M. and Van Brackle, D., "Optiml Corner Detector" Computer Vision, Graphics, and Image Processing 48:230—245. 1989
- [58] Azarbajejani, A. and Pentland, A., "Real-time self-calibrating stereo person tracking using 3-D shape estimation from blob features" Proceedings of the 13th International Conference on Pattern Recognition 627—632. 1996
- [59] Mikolajczyk, K. and Schmid, C., " A performance evaluation of local descriptors". IEEE Transactions on Pattern Analysis and Machine Intelligence 27:1615—1630. 2005
- [60] D. Lowe, "Distinctive image features from scale-invariant keypoints," Int. Journal on Computer Vision, vol. 60, no. 2, pp. 91–110, 2004
- [61] Bay, H. and Tuytelaars, T. and Van Gool, L., "Surf: Speeded up robust features" Computer Vision—ECCV 404—417. 2006
- [62] J. Zhang, M. Marszatek, S. Lazebnik, and C. Schmid, "Local features and kernels for classification of texture and object categories: A comprehensive study," Int. Journal of Computer Vision, vol. 73, no. 2, pp. 213–238, 2007
- [63] Y. Ke, R. S., and L. H., "Efficient near-duplicate detection and sub-image retrieval," in ACM Multimedia, 2004, pp. 869–876
- [64] S. F. Chang, W. Hsu, L. Kennedy, L. Xie, A. Yanagawa, E. Zavesky, and D.-Q. Zhang, "Columbia university trecvid-2005 video search and high-level feature extraction," in TRECVID, 2005
- [65] J. Sivic and A. Zisserman, "Video google: A text retrieval approach to object matching in videos," in IEEE Intl. Conf. on Computer Vision, 2003, pp. 1470–1477

- [66] C. Harris and M. Stephens. "A combined corner and edge detector." In *Alvey Vision Conference*, 1988
- [67] W. Förstner. "A framework for low level feature extraction." *ECCV*, 283-394, 1994
- [68] F. Heitger, L. Rosenthaler, R. von der Heydt, E. Peterhans, and O. Kuebler. Simulation of neural contour mechanism: from simple to end-stopped cells. In *Vision Research*, 32(5):963-981, 1992
- [69] T. Lindeberg, "Feature Detection with automatic scale selection", *IJCV* 30 (2):79-116. 1998
- [70] K. Mikolajczyk, C. Schmid, "Indexing based on scale invariant interest points." *International Conference on Computer Vision* 1:525-531. 2001
- [71] Y. Ke, R. Sukthankar, "PCA-SIFT: a more distinctive representation for local image descriptors", *Computer Vision and Pattern Recognition* 2:506-513. 2004
- [72] T. Kadir, M. Brady, "Scale, saliency and image description", *IJCV* 45 (2):83-105. 2001
- [73] F. Jurie, C. Schmid, "Scale-invariant shape features for recognition of object categories", in: *CVPR*, 2:90-96. 2004
- [74] K. Mikolajczyk, C. Schmid, "Scale and affine invariant interest point detectors", *IJCV* 60 (1) (2004) 63-86.
- [75] Moravec, H. P.. "Obstacle avoidance and navigation in the real world by a seeing robot rover." tech. report CMURITR8003 Robotics Institute Carnegie Mellon University doctoral dissertation Stanford University, 1980
- [76] Freeman, W. and Adelson, E. 1991. "The design and use of steerable filters." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(9):891-906
- [77] Mikolajczyk, K.. "Interest point detection invariant to affine transformations." PhD thesis, Institut National Polytechnique de Grenoble. 2002
- [78] T. Lindeberg 1994. "Scale space theory: A basic tool for analysing structures, at different scales." *Journal of applied statistics*. 21(2):224-270
- [79] Edelman, S., Intrator, N. and Poggio, T. "Complex cells and object recognition." Unpublished manuscript: <http://kybele.psych.cornell.edu/~edelman/archive.html> 1997.
- [80] Smith, L.I., "A tutorial on principal components analysis", Cornell University, USA, vol.51 pag.52. 2002

- [81] F. Mindru, T. Tuytelaars, L. Van Gool, T. Moons, Moment invariants for recognition under changing viewpoint and illumination, CVIU 94 (1-3) (2004) 3-27.
- [82] Juan, L. and Gwun, O.. A comparison of sift, pca-sift and surf. International Journal of Image Processing (IJIP). vol (3) number (4) pp. 143-152. 2009
- [83] M.A. Fishler and R.C. Boles. "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography." Comm. Assoc. Comp,Mach., 24(6): 381-395, 1981
- [84] Liu, Y and Lin, W.C. and Hays. "Near-regular texture analysis and manipulation." J. ACM Transactions on Graphics: (3)368--376, 2004
- [85] J.G. Leu. On indexing the periodicity of image textures. Image and Vision Computing, 19(13):987-1000, 2001
- [86] S.R. Jan and Y.C. Hsueh. "Window-size determination for granulometrical structural texture classification." Pattern Recognition Letters, 19(5-6): 439-446, 1998.
- [87] Malik, J. and Belongie, S. and Shi, J. and Leung, T., "Textons, contours and regions: Cue integration in image segmentation" ICCV, 918, 1999.
- [88] Zhu, S.C. and Guo, C. and Wang, Y. and Xu, Z., "What are textons?" International Journal of Computer Vision 62(1):121-143,2005
- [89] Hong, B.W. and Soatto, S. and Ni, K. and Chan, T., "The scale of a texture and its application to segmentation." CVPR, 1-8, 2008
- [90] Grigorescu, SE and Pektov, JMF. "Texture analysis using Renyi's generalized entropies." ICIP, 241-244. ,2003
- [91] Ahuja, N and Todorovic, S., "Extracting texels in 2.1 D natural textures." ICCV, 1-8, 2007
- [92] J Hays, J.; Leordeanu, M.; Efros, A. A. & Liu, Y., "Discovering Texture Regularity as a Higher-Order Correspondence Problem", In *9th European Conference on Computer Vision*, May 2006., Springer, , pp. 522-535 .
- [93] Ardizzone, E. and Bruno, A. and Mazzola, G., "Visual Saliency by Keypoints Distribution Analysis", Image Analysis and Processing ICIAP 2011 pages (691--699) , 2011
- [94] Constantinidis, C., and Steinmetz, M. A.: Posterior parietal cortex automatically encodes the location of salient stimuli. *The Journal of Neuroscience* 25(1), 233-238 (2005)

- [95] Desolneux A., Moisan L. Morel J.-M, "Computational Gestalts and Perception Thresholds", *Journal of Physiology-Paris*, March 2003, 97(2): 311-324(14), 2003
- [96] S.Bileschi, L.Wolf, "Image representations beyond histograms of gradients: The role of Gestalt descriptors", *CVPR*, 2007.
- [97] Ngau, C.W.H. and Ang, L.M. and Seng, K.P. "Comparison of Colour Spaces for Visual Saliency", *International Conference on Intelligent Human-Machine Systems and Cybernetics*, 2009. IHMSC'09. 278—281. 2009
- [98] Stéphane Paris, "Color- and texture-based salient map hierarchy", *International Conference on Image Processing*, 2205--2208 .2010
- [99] Achanta, R. and Susstrunk, S., "Saliency detection using maximum symmetric surround". *International Conference on Image Processing*. 2653—2656. 2010
- [100] Wang, Z. and Li, B., "A two-stage approach to saliency detection in images.", *Acoustics, IEEE International Conference on Speech and Signal Processing*, 265—268. 2008.
- [101] Koch, C., and Ullman, S.: Shifts in selective visual attention: towards the underlying neural circuitry. *Human Neurobiology* 4, 219-227 (1985)
- [102] Itti, L., Koch, C., and Niebur, E.: A model of saliency-based visual attention for rapid scene analysis, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1998, 20, (11), pp.1254-1259.
- [103] J. Harel, C. Koch, and P. Perona. Graph-based visual saliency. In *Advances in Neural Information Processing Systems 19*, pages 545–552. MIT Press, 2007
- [104] J. Luo. Subject content-based intelligent cropping of digital photos. In *IEEE International Conference on Multimedia and Expo*, 2007.
- [105] V. Sundstedt, A. Chalmers, K. Cater, and K. Debattista. Topdown visual attention for efficient rendering of task related scenes. In *Vision, Modeling and Visualization*, pages 209–216, 2004.
- [106] L. Itti and C. Koch. Computational modeling of visual attention. *Nature Reviews Neuroscience*, 2(3), 2001.
- [107] L.-Q. Chen, X. Xie, X. Fan, W.-Y. Ma, H.-J. Zhang, and H.-Q. Zhou. A visual attention model for adapting images on small displays. *ACM Multimedia Systems Journal*, 9(4), 2003.
- [108] Judd, Y. and Ehinger, K. and Durand, F. and Torralba, A. Learning to predict where humans look, *IEEE 12th International Conference on Computer Vision*, pages 2106-2133, 2009

- [109] <http://people.csail.mit.edu/tjudd/WherePeopleLook/index.html>
- [110] <http://www.saliencytoolbox.net>
- [111] Sencar, T. and Memon, N., "Overview of the State-of-the-art in Digital Image Forensics", World Scientific Press, 2008
- [112] Farid, H., "Image Forgery Detection.", Signal Processing Magazine, IEEE, 26(2):16—25, 2009
- [113] J. Fridrich, B. D. Soukal, A. J. Lukáš. Detection of copy-move forgery in digital images. in Proceedings of Digital Forensic Research Workshop, 2003
- [114] C. Popescu, H. Farid. Exposing digital forgeries by detecting duplicated image regions. Dept. Comput. Sci., Dartmouth College, Tech. Rep., 2004
- [115] G. Li, Q. Wu, D. Tu, S. Sun. A sorted neighborhood approach for detecting duplicated regions in image forgeries based on DWT and SVD. IEEE International Conference on Multimedia and Expo, 2007
- [116] W. Luo, J. Huang, G. Qiu. Robust Detection of region-duplication in digital image. Pattern Recognition 4:746-749, 2006
- [117] H. Huang, W. Guo, Y. Zhang. Detection of Copy-move Forgery in digital image using SIFT Algorithm. IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application, 2008, 272-276.
- [118] E. Ardizzone, A. Bruno, G. Mazzola. Detecting multiple copies in tampered images. International Conference on Image Processing, 2010, 2117-2120.
- [119] Amerini, L. Ballan, R. Caldelli, A. Del Bimbo, G. Serra. Geometric tampering estimation by means of a SIFT-based forensic analysis. IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP), 2010, 1702-1705.
- [120] X. Pan, S. Lyu. Region duplication detection using image feature matching. Information Forensics and Security, IEEE Transactions on, 2010, 857-867.
- [121] <http://www.cs.ubc.ca/~lowe/keypoints/>
- [122] <http://surf.co.tv/>
- [123] <http://www.csse.uwa.edu.au/~pk/research/matlabfns>

- [124] Dyer R., Zhang H., Möller T.. "A survey of Delaunay structures for surface representation. 2009
- [125] Zimmer H. "Voronoi and Delaunay Techniques" Lecture Notes, Computer Science VIII, RWTH Aachen, 2005.